.UBAfiuba
FACULTAD DE INGENIERÍA

Radar impulsivo Ultra-Wideband: plataforma digital y algoritmos para monitoreo y clasificación de blancos dinámicos a corta distancia

Ing. Edgardo José Marchi

Doctorado de la Universidad de Buenos Aires, área Ingeniería

Directora:
Dra. Cecilia Galarza

Co-directores:
Dr. Andrés Altieri
Dr. Mario Hueda

Jurados:
Dr. Juan Pablo Pascual, Instituto Balseiro
Dr. Francisco Grings, FCEN - Universidad de Buenos Aires
Dra. Luciana De Micco, Universidad Nacional de Mar del Plata
Dr. Javier Areta, Universidad Nacional de Río Negro

Ciudad de Buenos Aires, 2024

Dedicado a todos mis seres queridos.
Este trabajo es producto de su acompañamiento y apoyo incondicional.
Los logros obtenidos son tanto míos como suyos.

Resumen

"En todo hay una porción de todo"

Anaxágoras.

Los radares de banda ultra ancha o *Ultra Wide-Band* (UWB) han cobrado gran interés en los últimos años a medida que nuevas tecnologías permitieron el surgimiento de numerosas aplicaciones de sensado remoto. Específicamente, dado que las señales UWB poseen ciertas características particulares, es posible obtener información sobre los blancos iluminados que no es posible extraer al trabajar con radares clásicos. Adicionalmente, la resolución en rango de los radares UWB es más alta que la de los radares tradicionales. Sin embargo, la implementación de un radar UWB no es una tarea sencilla debido a que requiere de un gran ancho de banda proporcional y de una alta tasa de muestreo.

En este contexto tecnológico el Instituto Nacional de Tecnología Industrial (INTI) y el Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET) firmaron un acuerdo para el desarrollo de la tecnología. En el marco del mismo, el Centro de Simulación Computacional (CSC-CONICET) conjuntamente con el Departamento de Comunicaciones (DDC-INTI) trabajaron en la investigación y el desarrollo de varias plataformas y algoritmos para UWB y sus aplicaciones.

Esta tesis, realizada en dicho marco, presenta por un lado, el desarrollo de la arquitectura digital de una plataforma experimental para UWB basada en tecnología *System on Chip* (SoC) sobre hardware de código abierto. El diseño de la misma fue llevado a cabo enteramente con énfasis en la replicabilidad y configurabilidad, con el objetivo de poder oficiar tanto de plataforma de adquisición de señales como también de banco de pruebas para el desarrollo de algoritmos de procesamiento de señales en tiempo real. Principalmente el trabajo aborda el desarrollo de la cadena de procesamiento digital de señales. Esto incluye el diseño y la implementación sobre hardware programable (FPGA) del esquema de muestreo, el pre-procesamiento de las muestras, el armado de tramas y el procesamiento de las mismas en tiempo real. Su transmisión a una PC a través de Gigabit Ethernet (GbE) es manejada por un sistema operativo de tiempo real corriendo en el procesador (PS) del mismo SoC. Tanto el diseño del método de muestreo como el desarrollo e implementación de la placa receptora también se cubren en este trabajo. El hardware fue completamente desarrollado e implementado utilizando componentes *off-the-shelf*, lo que permite que la plataforma sea fácilmente replicable y adaptable a diferentes aplicaciones. Adicionalmente, la cantidad de recursos utilizados en la implementación resultó ser considerablemente baja debido a la utilización de lenguajes de bajo nivel (VHDL y C respectivamente) tanto en la descripción del hardware programable como en el software que corre en el procesador. Dado que el diseño de la plataforma fue realizado con un alto grado de configurabilidad en mente, el prototipo construido demostró ser una solución

flexible y adecuada para la investigación y el desarrollo de aplicaciones UWB.

El segundo aspecto presentado en el trabajo se refiere al desarrollo de un algoritmo tiempo-frecuencia (TF) con el objetivo de extraer características no estacionarias de las señales del radar relacionadas a blancos dinámicos. El objetivo en dicho desarrollo fue obtener un esquema de procesamiento de alta tasa de datos para ser implementado sobre una plataforma digital de recursos acotados, o para procesar un gran volumen de información en un tiempo menor que por métodos convencionales. Se puso especial énfasis en obtener un algoritmo capaz de proporcionar un incremento local de la resolución sin aumentar la cantidad de puntos de análisis ni la carga computacional que ello conlleva. La solución propuesta se basa en la transformada *Synchrosqueezing* (SST) y adaptativamente aumenta la resolución TF en función de la distribución de energía de la señal. Adicionalmente, permite la utilización de varios métodos de reasignación tiempo-frecuencia (TFR), lo que da la posibilidad al usuario de configurar el algoritmo de acuerdo a las características particulares de la señal bajo análisis y/o aplicación. La implementación se llevó a cabo en Python como lenguaje principal, lo que dio origen a un *toolbox* de código abierto empaquetado para su fácil instalación y uso. La programación incluyó técnicas de paralelismo, vectorización y optimización de código, lo que permitió obtener un algoritmo de alto rendimiento. También fue diseñado para servir como herramienta en diferentes aplicaciones, por lo que proporciona al usuario final una amplia gama de opciones de configuración. Estas opciones no se limitan solo a los parámetros del algoritmo sino que también permiten la selección del *backend* sobre el cual se ejecuta, adaptándose a la plataforma computacional disponible. Se presentan resultados de simulación y experimentales que muestran las ventajas del algoritmo propuesto en comparación con otros métodos de análisis tiempo-frecuencia.

El lector encontrará finalmente en este trabajo la aplicación de los algoritmos desarrollados a señales experimentales tanto de blancos estáticos como dinámicos de varias fuentes. Particularmente, la plataforma diseñada fue utilizada para clasificar blancos estáticos con diferente contenido de humedad, en un entorno experimental. Los datos recibidos fueron procesados con una cadena de procesamiento de señales que incluyó la eliminación de ruido, la extracción de frecuencias naturales, la transformación del dominio para mejorar la separabilidad y la clasificación. Los resultados muestran el potencial de la tecnología UWB para la clasificación de contenido de humedad de manera no invasiva. Adicionalmente, este trabajo presenta la aplicación del algoritmo TF desarrollado como una herramienta potencial para el sensado remoto de signos vitales, como la respiración, la frecuencia cardíaca y el movimiento cardíaco en el rango de frecuencias del electrocardiograma (ECG). El algoritmo fue probado con un conjunto de datos o *dataset* de señales de radar UWB que contenían señales de referencia de respiración y ECG. También fue evaluada su aplicación en tiempo real utilizando un software experimental desarrollado. Los resultados obtenidos demostraron el buen rendimiento del algoritmo desarrollado tanto en términos de resolución TF como de eficiencia computacional. Los resultados obtenidos en ambas áreas muestran el potencial de nuevas aplicaciones que la tecnología UWB tiene para ofrecer.

Abstract

Ultra-Wideband Impulse Radar: Digital Platform and Algorithms for Monitoring and Classification of Dynamic Targets at Short Range

Ultra Wide-Band (UWB) radars have gained significant interest in recent years as new technologies have enabled the emergence of a wide range of remote sensing applications. Specifically, since UWB signals possess particular characteristics, obtaining certain information from illuminated targets that traditionally could not be extracted by typical radars is possible. Moreover, the distance resolution of UWB radars is much higher than the resolution of traditional ones. However, implementing a UWB radar is not straightforward as it requires a large proportional bandwidth and a high sampling rate.

In this technological context, the National Institute of Industrial Technology (INTI) and the National Scientific and Technical Research Council (CONICET) agreed to develop this technology. Within this agreement, the Computational Simulation Center (CSC-CONICET) and the Communications Department (DDC-INTI) worked jointly on the research and development of several hardware platforms and algorithms for UWB technology and future applications.

This thesis was performed within the aforementioned framework. The initial focus was to develop the digital architecture of an experimental platform for UWB, based on System-on-Chip (SoC) technology over open-source hardware. The platform design was entirely accomplished with particular attention to replicability and ease of configuration. It is intended to serve as a signal acquisition platform and a test bench for developing real-time signal processing algorithms. This thesis mainly addresses the development of the digital signal processing chain. This includes the design and implementation on a programmable hardware (FPGA) of the sampling scheme, the pre-processing of the samples, the assembly of frames, and their real-time processing. Their transmission to a PC via Gigabit Ethernet (GbE) is handled by a real-time operating system running on the processing system (PS) of the same SoC. The design of the sampling method as well as the development and implementation of the mezzanine receiver board is also covered in this work. The hardware was entirely developed and implemented using *off-the-shelf* components, which enables the platform to be easily replicated and adapted to different applications. Additionally, the number of resources used in the implementation was considerably low due to the utilization of low-level languages (VHDL and C, respectively) to describe the

programmable hardware and the software running on the processor. Since the platform design was performed with a high degree of configurability in mind, the built prototype proved to be a flexible and suitable solution for the research and development of UWB applications.

The second aspect presented in this work is the development of a time-frequency (TF) algorithm, aimed at extracting non-stationary features from the radar signals related to dynamic targets. The goal of this development was to obtain a high data rate processing scheme to be implemented on a digital platform with limited resources, or to process a large volume of data in less time than conventional methods. Special emphasis was placed on the development of a method capable of providing a local improvement in resolution without increasing the number of analysis points or the computational load that this entails. The proposed method is based on the Synchrosqueezing Transform (SST) and increases adaptively the TF resolution of certain regions of the TF plane based on the signal energy distribution. It also allows the use of several time-frequency reassignment (TFR) methods, which enables the user to tune the algorithm to the particular characteristics of the signal under analysis and/or application. The implementation was carried out in Python as the main language, which gave rise to an open-source toolbox packaged for easy installation and use. The toolbox programming included parallelism techniques, vectorization, and code optimization, obtaining a high-performance algorithm. It was also designed to serve as a tool in different applications; hence, it provides the end user with a wide range of configuration options. These options are not limited only to algorithm parameters but also allow the selection of the back-end on which it runs, adapting to the available computational platform. Simulation and experimental results are presented and show the advantages of the proposed algorithm compared to other time-frequency analysis methods.

The reader will finally find in this work the application of the developed algorithms to experimental signals obtained from both static and dynamic targets and various sources. Particularly, the designed platform was used to classify static targets with different moisture content, within an experimental setup. The received data was processed with a signal processing pipeline including decluttering, natural frequencies extraction, and transformation of the domain to improve separability and classification. The results show the potential of the UWB technology for non-invasive moisture content classification. Additionally, this work presents the application of the developed TF algorithm as a potential tool for remote sensing of vital signs, such as respiration, heart rate, and heart movement in the Electrocardiogram (ECG) frequency range. The algorithm was tested with both a dataset of UWB radar signals containing reference respiration and ECG signals, and in real-time using a developed experimental software. The obtained results proved the good performance of the developed algorithm both in terms of TF resolution and computational efficiency. The obtained results in both areas show the potential for new applications that the UWB technology has to offer.

Lugar de trabajo:

Departamento de Comunicaciones - Instituto Nacional de Tecnología Industrial.
Laboratorio de Procesamiento de Señales y Comunicaciones - Facultad de Ingeniería - Universidad de Buenos Aires.

Financiamiento:

Este trabajo fue parcialmente financiado por el PICT: 2016-Nº1925:
“Desarrollo de técnicas de sensado no-invasivo mediante señales UWB y su implementación en una plataforma experimental.”
Directora: Cecilia Galarza

Índice general

Índice de figuras

Índice de Tablas

Lista de abreviaciones

INTI	Instituto Nacional de Tecnología Industrial
DDC	Departamento De Comunicaciones
CONICET	Consejo Nacional de Investigaciones Científicas y Técnicas
CSC	Centro de Simulación Computacional
ANPCyT	Agencia Nacional de Promoción Científica y Tecnológica
IEEE	*Institute of Electrical and Electronics Engineers*
ICTP	*International Centre for Theoretical Physics*
MLAB	*Multidisciplinary Laboratory*
FCC	*Federal Communications Commission*
ITU	*International Telecommunication Union*
TIC	tecnologías de la información y la comunicación
EDA	diseño electrónico automatizado o *Electronic Design Automation*
PC	computadora personal
HDL	lenguaje descritor de hardware
VHDL	*VHSIC Hardware Description Language*
USB	*Universal Serial Bus*
PRF	frecuencia de repetición de pulsos o *Pulse Repetition Frequency*
FPS	*frames* por segundo
PCB	circuito impreso o *Printed Circuit Board*
AFE	*Analog Front-End*
DSS	*Downconverter Sampler-Stage*
UWB	banda ultra ancha o *Ultra Wide-Band*
SAR	radar de apertura sintética o*Synthetic Aperture Radar*

IR	radio/radar impulsivo o *Impulse Radio/Radar*
CW	onda continua o *Continuous Wave*
UWB-IR	*Ultra Wide-Band Impulse Radar*
RF	Radio Frecuencia
IF	Frecuencia Intermedia
PPM	modulación por posición de pulso o *Pulse Position Modulation*
ToF	tiempo de vuelo o *Time of Flight*
FPGA	*field-programmable gate array*
SoC	*system-on-chip*
DCI	Impedancia Controlada Digitalmente o *Digitally Controlled Impedance*
CIAA-ACC	Computadora Industrial Abierta Argentina para aplicaciones de Alto Costo Computacional
ADC	conversor analógico digital o *Analog-to-Digital Converter*
DAC	conversor digital analógico o *Digital-to-Analog Converter*
GbE	Gigabit Ethernet
SPI	*Serial Peripheral Interface*
UDP	*User Datagram Protocol*
MTU	*Maximum Transmission Unit*
WBAN	*wireless body area network*
LNA	amplificador de bajo ruido o *low noise amplifier*
PS	*Processing System*
PL	*Programmable Logic*
ISA	*Instruction Set Architecture*
MMU	Unidad de gestión de memoria o *Memory Management Unit*
DMA	acceso directo a memoria o *Direct Memory Access*
FSBL	*First Stage Boot Loader*
RTOS	Sistema Operativo de Tiempo Real o *Real-Time Operating System*
API	interfaz de programación de aplicaciones o *Application Programming Interface*

ABI	interfaz binaria de aplicaciones o *Application Binary Interface*
SIMD	*Single Instruction Multiple Data*
TE	tiempo equivalente
IMT	tipo Modo Intrínseco o *Intrinsic-Mode Type*
ETS	muestreo en tiempo equivalente o *Equivalent Time Sampling*
TF	tiempo frecuencia o *Time-Frequency*
IF	frecuencia instantánea o *Instantaneous Frequency*
FFT	transformada rápida de Fourier o *Fast Fourier Transform*
STFT	*Short-Time Fourier Transform*
WT	transformada wavelet o *Wavelet Transform*
CWT	transformada wavelet continua o *Continuous Wavelet Transform*
SST	*Synchrosqueezing Transform*
ASST	*Adaptive Synchrosqueezing Transform*
B-ASST	*Batched Adaptive Synchrosqueezing Transform*
TFR	*Time-Frequency Reassignment*
SET	*Synchro-Extracting Transform*
RM	*Reassign Method*
TSST	*Time-reassigned Synchrosqueezing Transform*
OTL	fuera del ciclo o *Off-The-Loop*
ITL	dentro del ciclo o *In-The-Loop*
PCA	análisis por componentes principales
SEM	método de expansión de singularidades o *Singularity Expansion Method*
ECG	electrocardiograma
PCG	fonocardiograma
CSA	cámara semi-anecoica
RCS	sección eficaz de radar o *Radar Cross Section*
MSE	error cuadrático medio o *Mean Square Error*
SNR	relación señal a ruido o *Signal to Noise Ratio*
SVD	descomposición en valores singulares o *singular value decomposition*

Capítulo 1

Introducción

RESUMEN: En este capítulo introductorio se presenta la motivación del presente trabajo. Se plantean los objetivos del mismo y se describe el marco general del proyecto. Adicionalmente, se detallan las contribuciones realizadas en el marco de este doctorado, y se presenta la organización del manuscrito.

1.1. Motivación

Las últimas décadas han presentado un crecimiento exponencial en la cantidad de dispositivos electrónicos asociados a tecnologías de la información. Las tecnologías actuales de ”5G”, y las de próxima generación ”6G” proponen una integración total de la localización y el sensado a la comunicación de datos [1]. En este contexto, las señales de banda ultra ancha o *Ultra Wide-Band* (UWB) surgen como una potencial solución para la mencionada integración de estas tecnologías. Las mismas son señales que ocupan un ancho de banda mayor a 500 MHz o que tienen una relación entre el ancho de banda y la frecuencia central geométrica mayor a 0.2 [2].

Estas señales poseen la característica particular de que por medio de ellas es posible obtener cierta información de objetivos que tradicionalmente resultaba extremadamente difícil o no podía ser detectada por radares convencionales como, por ejemplo, algunas características de materiales dieléctricos. Específicamente, los materiales reaccionan frente a las mismas devolviendo una respuesta constituida por dos componentes: la componente relacionada a la reflexión de la onda electromagnética sobre la superficie del blanco, y la componente relacionada a la penetración de la onda electromagnética en el material. A las mismas se las conoce como respuesta temprana y respuesta tardía respectivamente [3]. La respuesta temprana se puede utilizar para la localización de blancos, al igual que un radar tradicional. La principal diferencia en este aspecto radica en la precisión de la localización (proporcional al ancho de banda). Por otro lado, la respuesta tardía permite obtener información sobre la composición y/o geometría del blanco, ya que está relacionada a la emisión electromagnética del mismo debido a la formación de ondas estacionarias en su interior.

Lo mencionado anteriormente otorga un gran potencial de aplicación de las señales UWB, sin embargo, la generación, sensado y procesamiento de las mismas es una tarea compleja. Las limitaciones tecnológicas asociadas a la implementación de sistemas de

UWB acotan la difusión de soluciones de hardware disponibles para la generación y sensado de las mismas. Esto mismo reduce la capacidad de experimentación con señales del mundo real para diseñar, implementar y validar nuevas técnicas de procesamiento. Así mismo, la elevada tasa de muestreo requerida y la carga computacional de los algoritmos, van en detrimento de la utilización de esta tecnología en aplicaciones de baja latencia o tiempo real.

Si bien en la década de 1990 la tecnología UWB se comenzó a utilizar en aplicaciones de sensado a través de muros y para localización, la misma estaba impulsada mayormente por programas militares y de defensa [3]. A mediados de los años 2000 la tecnología UWB empezó a despertar el interés de la comunidad científica/tecnológica [4] debido a los avances en dispositivos y materiales electrónicos (y en las herramientas diseño electrónico automatizado o *Electronic Design Automation* (EDA) correspondientes). Sin embargo, recién en el año 2019 se incluyó el término UWB en un producto comercial de consumo, el iPhone 11 de Apple. Su uso comercial se limita a la transferencia de información (sustituyendo a la tecnología Bluetooth) y a la localización de dispositivos activos.

Al momento de escritura de esta tesis la riqueza de información contenida en las señales UWB se encuentra sub-explotada. Concretamente, el uso de la información contenida es un área de investigación y desarrollo en aplicaciones de salud [5][6][7], seguridad [8][9] localización [10] o monitoreo remoto de procesos industriales [11], entre muchas otras.

La literatura se encuentra focalizada por un lado, en el desarrollo de plataformas de hardware orientadas a aplicaciones específicas, generalmente basadas en silicio a medida (y por ende de limitada accesibilidad para el resto de la comunidad científica)[12][13][14], y por otro, a la implementación de algoritmos avanzados de procesamiento de señales mayoritariamente orientados a cómputo *offline*[15][16][17]. Si bien durante el desarrollo del presente doctorado surgieron algunas plataformas experimentales que se presentarán en el Capítulo 5, en el último año han dejado de estar disponibles para la comunidad científica[18], y la falta de una plataforma multipropósito para experimentación aún sigue siendo un problema para la investigación en el área.

La disponibilidad de una plataforma experimental resulta de suma importancia para la evolución de esta tecnología. No solamente para la experimentación de las propiedades físicas de estas señales en diferentes entornos, sino también para la recolección de datos reales que permitan el desarrollo de algoritmos de procesamiento de señales. Adicionalmente, el procesamiento de grandes volúmenes de datos con baja latencia o tiempo real requiere arquitecturas de cómputo heterogéneas y específicas a la aplicación [19][20], por lo que disponer de reconfigurabilidad en el hardware de procesamiento es un requerimiento fundamental. En ese mismo sentido, existe una demanda de algoritmos que exploten el paralelismo y la aceleración proporcionada por las arquitecturas específicas.

1.2. Objetivos

El objetivo general de esta tesis es plantear soluciones novedosas para el problema de monitoreo y clasificación de blancos dieléctricos dinámicos utilizando señales UWB. Particularmente, en un contexto de baja potencia transmitida y entornos de cercanía.

Para ello se plantean los siguientes objetivos específicos:

- Contribuir al diseño e implementación de una plataforma de transmisión, recep-

ción, digitalización y procesamiento de señales UWB, con foco en la versatilidad. Particularmente, en el desarrollo de una plataforma digital. Entendemos como plataforma digital a toda la infraestructura comprendida entre el sistema de cómputo que interactúa con el usuario final y la generación/muestreo de señales analíticas y/o de banda base. Dicha definición excluye de los objetivos del presente trabajo a la investigación y desarrollo de elementos de radiofrecuencias, aunque se incluyen y se integran en la solución propuesta.

El objetivo es disponer de una herramienta de generación y adquisición que adicionalmente permita descargar potencia de cómputo sobre el hardware, de forma de poder evaluar la aplicabilidad de algoritmos a problemas concretos y diversos. El vasto conjunto de posibles aplicaciones de la tecnología, imponen requerimientos de configurabilidad amplios sobre todos los parámetros del sistema que deberán ser atendidos.

- Estudiar el problema de estimación de parámetros de blancos dinámicos y proponer una solución algorítmica a problemas con requerimientos de baja latencia o tiempo real. La complejidad que se introduce al abandonar los modelos estáticos presenta desafíos no triviales a la hora de extraer características o *features* de un blanco.

  En particular, se plantea el desarrollo de un algoritmo tiempo frecuencia o *Time-Frequency* (TF) capaz de hacer un seguimiento a lo largo del tiempo de componentes oscilatorias dentro de las señales reflejadas de un blanco. El mismo deberá ser computacionalmente viable para aplicaciones con requerimientos de tiempo y suficientemente versátil como para poder aplicarse a problemas diversos.

1.3. Marco general del proyecto

El presente trabajo se enmarca en el Convenio de Investigación y Desarrollo entre el Instituto Nacional de Tecnología Industrial (INTI) y Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET) (Expte. Nº6932/16) para el desarrollo de capacidades técnicas conjuntas en relación a sistemas y tecnologías UWB. Dicho convenio incluyó otros proyectos de investigación y desarrollo por parte tanto de personal del INTI como de CONICET. Dichos trabajos son complementarios al presente y se mencionan en las secciones correspondientes.

El tesista forma parte de la planta permanente del INTI dentro del Departamento De Comunicaciones (DDC) (ex laboratorio de Radio-Comunicaciones) dependiente de la Sub-gerencia Operativa de Electrónica y Energía (ex Centro de Electrónica e Informática) del instituto.

Adicionalmente, parte del trabajo ha sido financiado por el proyecto PICT 2016-1925 de la Agencia Nacional de Promoción Científica y Tecnológica (ANPCyT) titulado ”Desarrollo de técnicas de sensado no-invasivo mediante señales UWB y su implementación en una plataforma experimental”. Y los proyectos UBACyT 20020150200248BA: ”Análisis, diseño e implementación de una plataforma inalámbrica de ancho de banda ultra grande (UWB) con aplicación a radares” y UBACyT 20020170200283BA ”Desarrollo de una plataforma inalámbrica UWB y de algoritmos de inferencia estadística para aplicaciones de radar”.

1.4. Contribuciones

En el marco mencionado anteriormente, este trabajo contribuyó concretamente en el desarrollo de tecnologías y algoritmos para aplicaciones de señales UWB al sensado remoto de blancos dieléctricos dinámicos. En particular el tesista desarrolló e implementó un esquema de muestreo y un hardware de procesamiento digital reconfigurable basado en tecnología *system-on-chip* (SoC) para la adquisición y procesamiento de señales UWB. A diferencia de otras soluciones, la plataforma presentada en el Capítulo 6 está basada íntegramente en componentes *off-the-shelf* por lo que es posible su replicación y/o adaptación. Así mismo fue desarrollada con el objetivo de ser una herramienta de investigación, por lo que se puso especial énfasis en la versatilidad de la misma incluyendo hardware reconfigurable (basado en tecnología SoC/FPGA) para otorgarle poder de procesamiento y capacidad de aceleración de cómputo.

Adicionalmente, el tesista creó un algoritmo adaptativo de análisis tiempo frecuencia para el monitoreo de blancos dinámicos orientado a aplicaciones de tiempo real presentado en el Capítulo 4. El mismo fue implementado en Python, es de código abierto y está disponible para su instalación como paquete en [21].

Se pueden mencionar dos ejes importantes en los aportes de este trabajo:

- Plataforma experimental:
  El método de digitalización presentado en el Capítulo 6 permite la obtención de tasas de muestreo elevadas y configurables en base a componentes *off-the-shelf* mediante un esquema de muestreo en tiempo equivalente híbrido basado en frecuencias fraccionales. El grado de configurabilidad es aportado por un SoC cómo núcleo de la plataforma. El mismo posee un área de hardware reconfigurable y un procesador. El hardware reconfigurable aporta tanto al esquema de muestreo propuesto como al `denoising` de la señal y ofrece la suficiente disponibilidad de recursos como para implementar aceleración de algoritmos por hardware. El procesador de doble núcleo también ofrece poder de procesamiento ocioso para aplicaciones de campo y/o tiempo real.

- Algoritmos:
  El algoritmo desarrollado y publicado en [22] presenta una arquitectura novedosa para el análisis TF basado en la transformada *Synchrosqueezing Transform* (SST) pero con un enfoque *data-driven* que redistribuye recursos computacionales en base a la información contenida en la señal. El mismo está disponible para su instalación y uso en [21] como fue mencionado.

Publicaciones

- E. Marchi, M. Cervetto and C. G, Galarza, “Adaptive synchrosqueezing wavelet transform for real-time applications,” in Digital Signal Processing, vol. 140, 2023, 104133, Elsevier, ISSN 1051-2004, doi: 10.1016/j.dsp.2023.104133.

- M. Cervetto, E. Marchi and C. G. Galarza, “A Fully Configurable SoC-Based IR-UWB Platform for Data Acquisition and Algorithm Testing,” in IEEE Embedded Systems Letters, vol. 13, no. 2, pp. 53-56, June 2021, doi: 10.1109/LES.2020.2997660.

- E. Marchi; M. Cervetto; P. Gámez, "Plataforma Digital de Bajo Costo para Procesamiento de Señales Ultra-Wideband," in SPL2019 - Southern Conference on programmable logic - ISBN 978-950-658-478-8

- P. Gámez, E. Marchi, M. Cervetto, C. Giuffrida, G. Perez, A. Altieri and C. G. Galarza, "A low-cost ultra-wideband test-bed for dielectric target detection," 2017 XVII Workshop on Information Processing and Control (RPIC), Mar del Plata, Argentina, 2017, pp. 1-6, IEEE, doi: 10.23919/RPIC.2017.8214363.

- A. Altieri; P. Gámez; E. Marchi; M. Cervetto, M. Bouza and C. G. Galarza, "A Low-Cost Ultra-Wideband Test-Bed," in XXXV Simpósio Brasileiro de Telecomunicações e Processamento de Sinais (SBrT 2017), 2017, Sao Pedro, Brasil. ISBN:978-85-66836-18-9.

Adicionalmente como trabajo relacionado al desarrollo sobre tecnologías SoC involucrado en este doctorado, se publicó en conjunto con el equipo del *Multidisciplinary Laboratory* (MLAB) del *International Centre for Theoretical Physics* (ICTP):

- W. Florian, B. Valinotti, L. García, M. Cervetto, E. Marchi, M. L. Crespo, S. Carrato and A. Cicuttin, "An Open-Source Hardware/Software Architecture for Remote Control of SoC-FPGA Based Systems," in Applications in Electronics Pervading Industry, Environment and Society. ApplePies 2021. Lecture Notes in Electrical Engineering, vol 866, Springer, doi: 10.1007/978-3-030-95498-7_10

1.5. Organización de la tesis

El presente trabajo se organiza según el siguiente esquema:

- Luego de este capítulo introductorio, en el Capítulo 2 se presenta la tecnología UWB. En particular, se introduce como una evolución tecnológica de los radares tradicionales a partir de la cual surge un conjunto de nuevas potenciales aplicaciones. Adicionalmente, se presentan las características de las señales UWB asociadas a las respuestas de los blancos iluminados y su entorno, y en ese contexto se presentan algunas técnicas comunes de procesamiento utilizadas.

- En el Capítulo 3 se analiza el problema particular de estimación de la distribución TF de una señal determinada. Se presentan los algoritmos de análisis TF más utilizados en la literatura y se analizan sus ventajas y desventajas.

- El Capítulo 4 introduce el algoritmo *Adaptive Synchrosqueezing Transform* (ASST) diseñado por el tesista y publicado en [22] junto con la implementación orientada a procesamiento de baja latencia disponible como *toolbox* en [21].

- En cuanto al avance de la tecnología UWB para adquisición y/o experimentación, el Capítulo 5 presenta un estado del arte de las plataformas de hardware disponibles para el desarrollo de aplicaciones. Se discuten las ventajas y desventajas de las mismas como así también la motivación para la investigación y desarrollo de una plataforma propia.

- En el Capítulo 6 se documenta el diseño detallado de la plataforma de hardware digital creada en el marco de este trabajo, desde los requerimientos propuestos hasta la implementación final.
- Finalmente, en el Capítulo 7 se muestran los resultados experimentales obtenidos tanto del análisis de blancos estáticos como dinámicos, en diferentes entornos y en aplicaciones particulares: sensado de humedad y monitoreo de signos vitales sin contacto. Además, se presenta una implementación de software que incluye la aplicación de uso del algoritmo ASST en tiempo real.

Por último el trabajo concluye poniendo en relevancia las posibles líneas de trabajo futuro y los aportes realizados.

Capítulo 2

Sistemas Ultra Wide-Band

RESUMEN: El presente capítulo introduce la tecnología UWB y sus características. Particularmente se la presenta como una evolución tecnológica de los radares tradicionales y se discute como la expansión de los parámetros de funcionamiento respecto del radar tradicional le otorga nuevas capacidades. Adicionalmente, se introduce terminología y notación que se utilizará a lo largo del documento. Habiendo presentado los conceptos básicos de Radar, se abordarán los procesos más comunes que se aplican sobre las señales recibidas. Particularmente se presentarán tanto las respuestas de los blancos como el clutter del entorno, y se introducirán técnicas para clasificación de los mismos y eliminación de clutter.

2.1. Introducción

Existe una concepción general que posiciona a la tecnología UWB como actual o "moderna" a partir de la cual se generó el desarrollo de aplicaciones para resolver problemas que anteriormente no disponían de solución. Sin embargo, es importante aclarar, citando a Ghavami et al. [23], que UWB es una nueva *tecnología de la Ingeniería* en la cual no se descubrieron nuevas propiedades físicas. Es por ello que resulta interesante introducir la tecnología UWB en el contexto de Radar, simplemente aclarando qué parámetros fueron extendidos y qué ventajas y/o dificultades conlleva dicha extensión.

Retomando las definiciones presentadas en el Capítulo 1 de la *International Telecommunication Union* (ITU) y la *Federal Communications Commission* (FCC), los sistemas UWB se caracterizan por poseer un ancho de banda mayor a 500 MHz o un ancho de banda fraccional (o relativo) B_{frac} mayor al 20 % de la frecuencia central. Si f_L y f_H son la frecuencia inferior y superior respectivamente a $-10\,dB$ respecto de la potencia nominal en la banda central entonces,

$$B_{frac} = \frac{2(f_H - f_L)}{f_H + f_L}. \tag{2.1}$$

Al cumplir con los requerimientos planteados, un sistema UWB se caracteriza por contar con una alta resolución espacial debido a su gran ancho de banda. Adicionalmente, dado que el mencionado ancho de banda es relativo a la portadora, las frecuencias

inferiores pueden ser consideradas "bajas" lo que permite penetrar obstáculos como árboles o incluso ciertas capas del suelo. El desarrollo de esta tecnología vino dado en sus comienzos por programas militares, y las primeras denominaciones que tuvo fueron *radar en banda base*, *tecnología sin portadora* o *tecnología impulsiva*. Adicionalmente, la tecnología UWB presenta las siguientes ventajas que la hacen atractiva para muchas aplicaciones en todos los ámbitos:

- Alta tasa de datos: la capacidad del canal se puede calcular como

$$C = B\log_2(1+\frac{S}{N}), \tag{2.2}$$

  siendo B el ancho de banda, S la potencia de la señal y N la potencia del ruido (Shannon). Dado que B para un sistema UWB es muy grande, la capacidad del canal puede resultar potencialmente muy alta.

- Coexistencia con otros sistemas: la densidad de potencia espectral de un sistema UWB es muy baja debido al bajo ciclo de trabajo de las transmisiones impulsivas, lo que permite que coexista con otros sistemas de comunicaciones sin interferir con ellos.

- Inmunidad a interferencia por múltiples caminos: en relación con el ítem anterior, el corto ancho temporal de los pulsos permite una separación más sencilla de las señales reflejadas a través de diferentes caminos de propagación.

- Capacidad de localización y transferencia de datos a la vez: la alta resolución temporal de las señales permite que los retardos en los pulsos recibidos se puedan utilizar para determinar la distancia al blanco/receptor mientras los mismos se modulan para transmisión de información.

- Alta capacidad de análisis de la firma de los blancos: como otra ventaja de la alta resolución temporal de las señales, se puede diferenciar más fácilmente la respuesta temprana debido a la reflexión de la señal respecto de la tardía relacionada a las ondas estacionarias formadas en el blanco.

En las secciones siguientes se introducirán los conceptos necesarios para formalizar estas características de los sistemas UWB.

2.2. Sistemas de Radar

Un esquema ilustrativo de un sistema de radar se muestra en la Figura 2.1. A este tipo de radar se lo conoce como monoestático, ya que se compone de una única antena transmisora/receptora, o una antena transmisora y una antena receptora muy cercanas entre sí, asociadas al mismo sistema, que mantiene una posición fija. Particularmente, si d_t es la distancia del blanco a la antena transmisora y d_r a la receptora, en condiciones normales $d_t \simeq d_r = d$ y la velocidad de propagación es aproximadamente la velocidad de la luz en el vacío (c). Entonces,

$$d \simeq \frac{c}{2} t_{of}, \tag{2.3}$$

dónde t_{of} se lo conoce como tiempo de vuelo o *Time of Flight* (ToF) y es el tiempo que tarda la señal en recorrer el camino más directo entre la antena transmisora y la receptora pasando por el blanco. Si t_0 es el instante de transmisión de la señal y t_r el de recepción de la misma, entonces $t_{of} = t_r - t_0$.

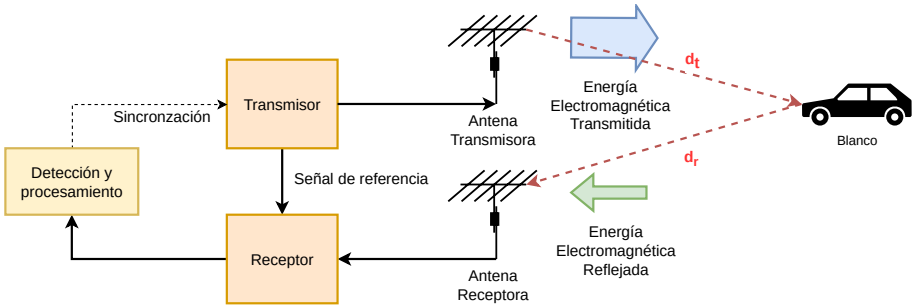


Figura 2.1: Esquema de un sistema de radar.

La ecuación de rango (distancia) para un radar monoestático con ancho de banda B es [24]:

$$\hat{R} = \frac{c}{4\pi}\left[\frac{\phi(f) - \phi_0(f) - \phi(f-B) + \phi_0(f-B)}{B} + \frac{\varepsilon(f) - \varepsilon(f-B)}{B}\right], \tag{2.4}$$

dónde $\phi(\cdot)$ representa la fase de "$\cdot$", ϕ_0 es la fase inicial de la señal (o de referencia) y $\varepsilon(\cdot)$ es el error asociado a la determinación de la fase de "$\cdot$".

Observando el término del error, se puede ver claramente que la precisión de la medición de distancia depende inversamente del ancho de banda de la señal transmitida, introduciendo así la primera ventaja de los sistemas UWB.

Si $s(t) = s_1(t) + s_2(t)$ es la señal a transmitir con

$$s_1(t) = Asin(\omega_1 t + \varphi_1),$$
$$s_2(t) = Asin(\omega_2 t + \varphi_2),$$

la señal trasmitida será $s(t_0)$ y la recibida $s(t_r)$. Despreciando los términos de error, la ecuación (2.4) resulta en

$$\hat{R} = \frac{c}{2}\left[\frac{\omega_2 t_r + \varphi_2 - (\omega_2 t_0 + \varphi_2) - (\omega_1 t_r + \varphi_1) + \omega_1 t_0 + \varphi_1}{\omega_2 - \omega_1}\right] \tag{2.5a}$$

y simplificando se obtiene

$$\hat{R} = \frac{c}{2}(t_r - t_0). \tag{2.5b}$$

Ambas versiones de (2.5) sugieren dos formas generales de implementación de un radar: (2.5a) onda continua o *Continuous Wave* (CW) y (2.5b) radio/radar impulsivo o *Impulse Radio/Radar* (IR).

2.2.1. Radares de onda continua

Siguiendo la ecuación (2.5a), un esquema de la arquitectura simplificada de un radar de onda continua se muestra en la Figura 2.2.

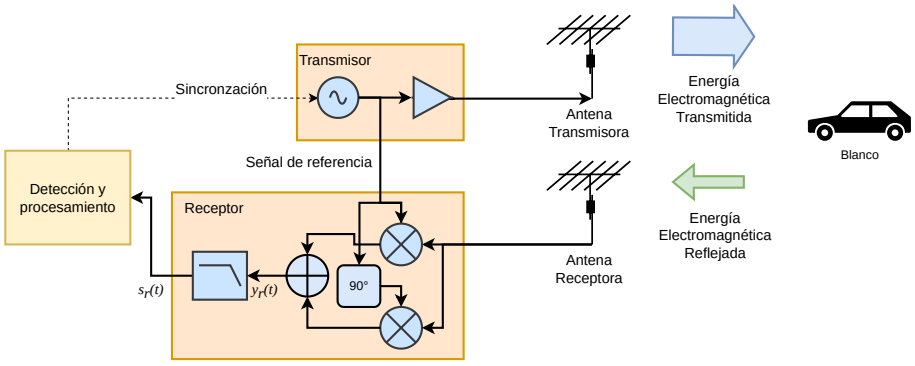


Figura 2.2: Esquema de un radar de onda continua.

Sea $\phi_0 = \omega t + \varphi$ la fase de la señal transmitida y $\phi = \omega(t + t_{of}) + \varphi$ la fase de la señal recibida con t_{of} el ToF. Entonces, la señal recibida luego del sumador es

$$\begin{aligned} y_r(t) &= A_r \cos(\omega(t + t_{of}) + \varphi)[\sin(\omega t + \varphi) + \cos(\omega t + \varphi)] \\ &= \frac{A_r}{2}[\sin(2\omega t + \omega t_{of} + 2\varphi) - \sin(\omega t_{of}) + \cos(\omega t_{of}) + \cos(2\omega t + \omega t_{of} + 2\varphi)] \\ &= \frac{A_r}{2}[\sin(2\omega t + \omega t_{of} + 2\varphi) + \cos(2\omega t + \omega t_{of} + 2\varphi)] + \frac{A_r}{2}[\cos(\omega t_{of}) - \sin(\omega t_{of})] \end{aligned} \tag{2.6}$$

donde A_r es la amplitud de la señal recibida. Aplicando un filtro pasabajos a la salida del mezclador resulta

$$\begin{aligned} s_r(t) &= \frac{A_r}{2}[cos(\omega t_{of}) - sin(\omega t_{of})] \\ &= \frac{A_r}{\sqrt{2}} cos(\omega t_{of} + \pi/4). \end{aligned} \tag{2.7}$$

El tiempo de vuelo (y la distancia al blanco) puede ser obtenido a partir de la amplitud de la señal recibida.

Se puede observar entonces que resulta una arquitectura sencilla desde su concepción. Otra ventaja de este tipo de sistemas es que la potencia transmitida es constante y típicamente mayor que en sistemas impulsivos, lo que permite una mejor SNR en la señal recibida. Estos radares se pueden aplicar en UWB mediante el uso de una señal *chirp* en el oscilador local cuyas frecuencias inicial y final cumplan con los requerimientos de ancho de banda. Sin embargo, instantáneamente siguen siendo sistemas inherentemente de banda angosta.

2.2.2. Radares impulsivos

Si las señales a transmitir son de naturaleza impulsiva, es posible utilizar la ecuación de rango en su variante dependiente de los tiempos (2.5b). La arquitectura simplificada del IR se muestra en la Figura 2.3.

En este caso, aplicando el mismo desarrollo que en (2.6) la señal resultante luego del filtrado queda

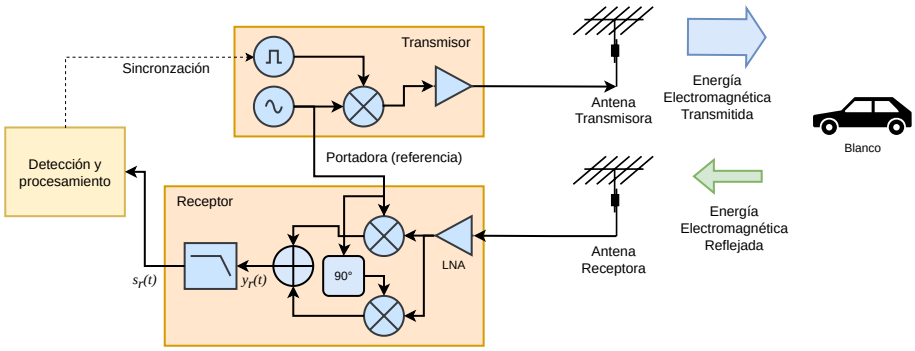


Figura 2.3: Esquema de un radar impulsivo.

$$\begin{aligned} s_r(t) &= A'_r p(t-t_{of}) \left[\cos(\omega t_{of}) - \sin(\omega t_{of})\right] \\ &= A'_r p(t-t_{of}) cos(\omega t_{of} + \pi/4) \end{aligned} \tag{2.8}$$

Dónde (2.8) es el equivalente de (2.7) para radares impulsivos.

Claramente la arquitectura de un radar impulsivo es más compleja, no sólo en su topología (que comparte cierta estructura con el radar CW) sino que también el diseño de cada bloque en particular presenta desafíos adicionales. Particularmente, se debe considerar la respuesta transitoria de todas las transferencias, a diferencia de un radar de onda continua donde se trabaja en estado mayormente estacionario. Adicionalmente, dado que típicamente la implementación involucra la transmisión de pulsos, el ciclo de trabajo de la señal transmitida es bajo, reduciendo la potencia promedio transmitida y comprometiendo la SNR en el receptor.

Para cumplir con los requerimientos de UWB en este caso basta con que el pulso transmitido sea de duración suficientemente corta. Si se cumple dicha condición, la señal es verdaderamente de UWB teniendo en todo instante todas las componentes de frecuencia presentes obteniéndose una respuesta más rica en información.

Por este motivo, en este trabajo se hará foco en este tipo de radares, sin embargo, dada la naturaleza común de las señales, el Capítulo 7 presenta la aplicación del algoritmo desarrollado en el Capítulo 4 a ambos tipos de radares.

2.3. Señales para sistemas UWB Impulsivos

Una señal UWB impulsiva debe cumplir con los requerimientos de ancho de banda definidos en la Sección 2.1 como fue mencionado. Típicamente se utilizan los pulsos presentados a continuación.

Pulso rectangular

La señal más simple que se puede utilizar en un sistema UWB es un pulso rectangular modulado. Se define al pulso rectangular como

$$p_T(t) = \begin{cases} 0, & \text{si} |t| > \frac{T}{2} \\ \frac{1}{2}, & \text{si} |t| = \frac{T}{2} \\ 1, & \text{si} |t| < \frac{T}{2} \end{cases} \tag{2.9}$$

y su transformada de Fourier es

$$\hat{p}(f) = \int_{-\infty}^{\infty} p(t) \cdot e^{-i2\pi ft} dt = 2\frac{\sin(2\pi fT/2)}{2\pi f} = T\,\text{sinc}(\pi fT) \tag{2.10}$$

donde aproximadamente el 90 % de la energía de la señal se encuentra en el intervalo $[-\frac{1}{T}, \frac{1}{T}]$. Si se requiere un ancho de banda mínimo de 500 MHz entonces $T \leq 2\ ns$.

En este caso el espectro se encuentra centrado en $f = 0$. Para centrarlo en la banda de interés basta con multiplicar el pulso por una exponencial compleja de la forma $e^{j2\pi f_0 t}$ (o equivalentemente convolucionar su espectro con una delta de Dirac centrada en la frecuencia de interés). La señal queda entonces definida como

$$s(t) = \sum_{i=0}^{\infty} p_T(t - iT_p) \cdot e^{j2\pi f_0 t} \tag{2.11}$$

siendo T_p el período de repetición de los pulsos. Las condiciones para UWB entonces resultan:

$$T \leqslant 2\ ns \tag{2.12a}$$

o equivalentemente

$$0{,}2 \cdot \frac{1}{T} \geqslant f_0. \tag{2.12b}$$

Pulso Gaussiano

Para mitigar la energía fuera de banda introducida por el pulso rectangular, se introdujeron pulsos Gaussianos que poseen una respuesta en frecuencia más plana dentro de la banda de interés y decae más rápidamente fuera de ella. Adicionalmente, esta familia de pulsos introduce la ventaja de poder centrar el espectro en una frecuencia distinta de $0\,Hz$, eliminando la necesidad introducir una modulación en frecuencia en la cadena. La expresión general para pulsos Gaussianos es:

$$p_{g_n} = \frac{\tau_c^n \left(\frac{n}{2}\right)!}{n!} \frac{\mathrm{d}^n}{\mathrm{d}t^n} \mathrm{e}^{-\frac{t^2}{\tau_c^2}} \tag{2.13}$$

donde n es el orden de la derivada y τ_c es un parámetro asociado al ancho de pulso.

Los tres pulsos Gaussianos más utilizados en UWB, correspondientes a $n = 0, 1, 2$ se presentan en la Tabla 2.1, donde también se puede observar el ancho de banda fraccional de los mismos, mediante la relación entre las dos últimas columnas. Dichas relaciones cumplen ampliamente con los requerimientos para UWB siendo $B_{frac} = 1$ y $B_{frac} = 1{,}155$ para los pulsos monociclo y biciclo respectivamente; un valor muy por encima del 0,2 requerido.

Pulso	Expresión temporal	Transformada de Fourier	Frecuencia central	Ancho de banda
Base	$p_{g_1}(t) = K_1 e^{-(t/\tau_c)^2}$ $K_1 = \sqrt{\frac{E_1}{\tau_c\sqrt{\pi/2}}}$	$\hat{p}_{g_1}(f) = K_1\tau_c\sqrt{\pi}e^{-(\pi\tau_c f)^2}$	0	$0{,}8623\frac{\sqrt{2}}{2\pi}\frac{1}{\tau_c}$
Monociclo	$p_{g_2}(t) = -2K_2\frac{t}{\tau_c^2}e^{-(t/\tau_c)^2}$ $K_2 = \sqrt{\frac{\tau_c E_2}{\sqrt{\pi/2}}}$	$\hat{p}_{g_2}(f) = K_2\tau_c\sqrt{\pi}(j2\pi f)e^{-(\pi\tau_c f)^2}$	$f_0 = \frac{\sqrt{2}}{2\pi\tau_c}$	$\frac{\sqrt{2}}{2\pi}\frac{1}{\tau_c}$
Biciclo	$p_{g_3}(t) = -2K_3\frac{1}{\tau_c^2}\left(1-\frac{2t^2}{\tau_c^2}\right)e^{-(t/\tau_c)^2}$ $K_3 = \tau_c\sqrt{\frac{\tau_c E_3}{3\sqrt{\pi/2}}}$	$\hat{p}_{g_3}(f) = K_3\tau_c\sqrt{\pi}(j2\pi f)^2 e^{-(\pi\tau_c f)^2}$	$f_0 = \frac{2}{2\pi\tau_c}$	$1{,}155\frac{\sqrt{2}}{2\pi}\frac{1}{\tau_c}$

Tabla 2.1: Pulsos Gaussianos más utilizados y sus parámetros.

2.4. Determinación del rango sin ambigüedad

Hasta el momento se consideraron ciertas simplificaciones en la determinación de la distancia al blanco o rango. Particularmente, se asumió que la fase de la señal recibida no supera una diferencia de 2π respecto de la transmitida. Adicionalmente, se asumió que se transmite un solo pulso, cuando en la práctica la trasmisión se realiza mediante un tren de pulsos a una frecuencia de repetición de pulsos o *Pulse Repetition Frequency* (PRF) determinada.

2.4.1. Ambigüedad de fase

Tanto (2.7) cómo (2.8) proveen una forma de determinar t_{of}, sin embargo la solución no es única ya que $e^{-j\phi} = e^{-j(\phi+2\pi)}$.

$$\begin{aligned} \omega_0 t_{of} < 2\pi \Rightarrow \omega_0\frac{2d_{max}}{c} &< 2\pi \\ 2\pi f_0\frac{2d_{max}}{c} &< 2\pi \\ d_{max} &< \frac{1}{f_0}\frac{c}{2} \end{aligned} \tag{2.14}$$

con f_0 la frecuencia de portadora. La primera restricción que surge, es entonces que la distancia máxima a medir queda restringida por la frecuencia central de portadora. Sin embargo, cuando se utilizan más de una frecuencia, como por ejemplo en (2.5a), dicha restricción se limita a la diferencia de frecuencias $B = \omega_2 - \omega_1$, ampliando el rango máximo. Esto resulta en otra ventaja de los radares UWB.

2.4.2. Ambigüedad de tiempo

Para sistemas impulsivos que transmiten un tren de pulsos a una PRF determinada, un pulso recibido después del tiempo de separación de pulsos no puede ser asociado a un pulso transmitido en particular por lo que partiendo de (2.3) el rango máximo sin ambigüedad de medición resulta

$$d_{max} = \frac{c}{2 \cdot \text{PRF}}\,, \tag{2.15}$$

por lo que la determinación de PRF debe considerar la distancia máxima a medir en la aplicación particular.

2.5. Respuesta de un blanco a una señal UWB

La medida de la potencia de la señal dispersada en unidades de ángulo sólido por un blanco respecto de la potencia incidente se denomina sección eficaz de radar o *Radar Cross Section* (RCS). En otras palabras, es una medida de la superficie equivalente que un blanco presenta a un radar en cierta dirección. Planteado en términos sistémicos, es posible obtener el campo electromagnético dispersado (E_s) a partir del campo incidente (E_i) y la función de transferencia del blanco (H). En el dominio de Laplace,

$$E_s(s) = H(s) \cdot E_i(s) \tag{2.16}$$

siendo s la variable de Laplace.

De esta forma, si analizamos las frecuencias sobre el eje $j\omega$ la RCS (σ_{CS}) queda definida como[24]

$$\sigma_{CS}(j\omega) = 4\pi \lim_{r\to\infty} r^2 \frac{|E_s(j\omega)|}{|E_i(j\omega)|} \tag{2.17}$$

dónde r es la distancia entre el radar y el blanco. Generalmente, la dependencia de la frecuencia de σ_{CS} puede ser enmarcada dentro de tres regiones:

- Región de Rayleigh: donde la longitud de onda es mucho mayor que el tamaño del blanco L (i.e. $L/\lambda \ll 1$).
- Región de resonancia: donde la longitud de onda es comparable al tamaño del blanco L ($0{,}5 < L/\lambda < 10$).
- Región óptica: donde la longitud de onda es mucho menor que el tamaño del blanco L ($L/\lambda > 10$).

En el caso de UWB, según la aplicación particular, podemos encontrar que la RCS de un blanco puede contener componentes asociadas a más de una región. En el caso de la región de Rayleigh, el campo electromagnético incide con aproximadamente la misma fase sobre el blanco generando una polarización de la carga sobre el mismo y por ende creando un momento dipolar. En la región de resonancia, la fase del campo incidente es sustancialmente distinta en diferentes puntos del blanco, induciendo corrientes sobre el mismo que generan ondas estacionarias. En este caso el campo dispersado tendrá un comportamiento oscilatorio o resonante. Por último, en la región óptica, el campo dispersado tiene su origen en pequeñas regiones del blanco que se comportan como puntos especulares, generando una amplia variación del campo dispersado en función del ángulo de incidencia.

2.5.1. Método de expansión de singularidades

Es un tema ampliamente estudiado que la transferencia $H(s)$ de un blanco puede ser expresada en términos de una serie de singularidades en el plano complejo[25], i.e.

$$H(\mathbf{r},s) = \sum_{i=0}^{\infty} \eta_i(s,\mathbf{p}) M_i(\mathbf{r})(s-s_i)^{-m_i} + W(\mathbf{r},s,\mathbf{p}) \tag{2.18}$$

donde:

- s_i es la singularidad i o frecuencia natural ubicada en el semi-plano negativo del plano s. La misma depende de la geometría y composición del blanco.
- m_i es el orden de la singularidad i.
- $\mathbf{r}$ es el vector posición.
- $\mathbf{p}$ es la polarización de la onda incidente.
- $\eta_i(s,\mathbf{p})$ es el coeficiente de acoplamiento que depende de singularidad s_i y la polarización de la onda incidente $\mathbf{p}$.
- $M_i(\mathbf{r})$ Es el modo natural i depende de la posición en la estructura y los parámetros del objeto.
- $W(\mathbf{r},s,\mathbf{p})$ es una función entera necesaria para asegurar la convergencia de la serie infinita.

Si bien (2.18) provee una forma explícita para las singularidades, al tratar de extraerlas a partir de la respuesta recibida, se observó que era posible hacerlo luego que la señal incidente atravesara el blanco completamente[26]. Esto determinó dos regiones diferentes en la respuesta temporal de la señal dispersada por un objeto: la *respuesta temprana* y la *respuesta tardía*. La respuesta general en el dominio del tiempo puede ser expresada entonces como

$$R(\mathbf{r},t) = \underbrace{u(t-t_0)\sum_i A_i(\mathbf{r})e^{s_i t}}_{\text{Respuesta tardía}} + \underbrace{W(\mathbf{r},t)[u(t)-u(t-t_0)]}_{\text{Respuesta temprana}} \tag{2.19}$$

con:

- $u(t)$ la función escalón unitario.
- t_0 el doble de tiempo que la señal tarda en atravesar el blanco.
- $A_i(\mathbf{r})$ el residuo que surge de combinar η_i con $M_i(\mathbf{r})$, constante en el tiempo.

Como ya fue mencionado anteriormente, estas resonancias que se encuentran en la respuesta tardía están asociadas a la forma y estructura del blanco y a su composición, pero no a su posición relativa respecto del radar. Aplicando este modelo para la respuesta de un blanco, es posible utilizar las singularidades para clasificarlo. Muchas técnicas de extracción de singularidades ampliamente estudiadas en la literatura pueden ser aplicadas, por ejemplo el método del *Matrix Pencil*[27] o el método de *ESPRIT*[28]. Particularmente el método *ESPRIT* se utilizó en el Capítulo 7.2 para la clasificación de contenido de humedad en un blanco.

2.6. Clutter

Hasta ahora se han analizado las características de las señales asociadas exclusivamente a las dispersiones introducidas por los blancos a analizar. Sin embargo, en aplicaciones reales, a la señal dispersada por un blanco se le suman las dispersiones presentes debido al entorno. Esto es particularmente importante en aplicaciones en interiores dónde el entorno puede resultar extremadamente complejo.

Se puede definir entonces a la señal recibida como

$$r(t) = s(t) + c(t) + n(t)$$

o en forma vectorial, considerando cada señal como un vector de muestras,

$$\mathbf{r} = \mathbf{s} + \mathbf{c} + \mathbf{n} \tag{2.20}$$

Siendo **r** la señal recibida, **s** la señal dispersada por el blanco, **c** la componente del clutter y **n** el ruido.

Si consideramos todos los efectos descorrelacionados entre sí, podemos plantear la matriz de covarianza de la señal recibida como

$$\begin{aligned} R_r &= \mathbb{E}\{\mathbf{r}\mathbf{r}^H\} \\ &= \mathbb{E}\{(\mathbf{s}+\mathbf{c}+\mathbf{n})(\mathbf{s}+\mathbf{c}+\mathbf{n})^H\} \\ &= \mathbb{E}\{\mathbf{s}\mathbf{s}^H\} + \mathbb{E}\{\mathbf{c}\mathbf{c}^H\} + \mathbb{E}\{\mathbf{n}\mathbf{n}^H\} \\ &= R_s + R_c + R_n \end{aligned} \tag{2.21}$$

donde R_s, R_c y R_n son las matrices de covarianza de la señal dispersada, el clutter y el ruido respectivamente.

Si se aplica una descomposición en valores singulares o *singular value decomposition* (SVD),

$$\begin{aligned} R_r &= U_r \Sigma_r V_r^H \\ &= U_s \Sigma_s V_s^H + U_c \Sigma_c V_c^H + U_n \Sigma_n V_n^H \end{aligned} \tag{2.22}$$

con Σ la matriz de valores singulares y U y V las matrices de vectores singulares izquierdos y derechos respectivamente. Particularmente Σ_s, Σ_c y Σ_n contienen los valores singulares asociados a la señal dispersada, el clutter y el ruido respectivamente.

Esta descomposición permite plantear la separación del clutter de la señal dispersada por el blanco en base a la selección de los valores singulares en Σ_r correspondientes.

Si la SVD se realiza ordenando los valores singulares de mayor a menor, los valores singulares asociados a diferentes efectos pueden estar mezclados entre sí y la correcta elección depende de la aplicación particular.

De todas formas, este planteo permite utilizar herramientas estadísticas para la mencionada selección.

2.6.1. Análisis de componentes principales

El análisis por componentes principales (PCA) es una técnica de reducción de dimensionalidad que permite encontrar las direcciones de máxima varianza en un conjunto de

datos. En términos generales se puede definir como una transformación tal que $Y = SX$ donde X es la matriz de datos, S es la matriz de transformación e Y es la matriz de datos transformada que particularmente cumple la condición de que sus componentes están descorrelacionadas [29]. Esta última condición implica una matriz de covarianza diagonal. Retomando la SVD donde $X = U\Sigma V^H$, se puede observar que U representa los autovectores de la matriz XX^H (que coincide con la matriz de covarianza de X). Considerando esto podemos plantear la transformación $Y = SX$ como

$$\Sigma V^H = U^H X \tag{2.23}$$

siendo $Y = \Sigma V^H$ y $S = U^H$. Dado que U es ortonormal $U^{-1} = U^H$, por lo que seleccionando algunas componentes de Y podemos obtener los datos originales asociados a las componentes elegidas, i.e.

$$\tilde{U}Y = \tilde{U}\Sigma V^H = \tilde{X} \tag{2.24}$$

donde $\tilde{U}$ es una sub-matriz de U conteniendo sólo las componentes que se quieren analizar.

En base a la expresión del clutter de (2.22), PCA surge como una herramienta ampliamente utilizada para la eliminación del mismo de la señal de radar recibida. Qué componentes se utilizan depende de la aplicación particular. Por ejemplo, más adelante en el presente trabajo (Sección 7.3), se utilizará la primera componente de PCA como señal de clutter debido a que las mediciones experimentales se realizaron en un entorno interior donde el clutter resulta predominante.

2.7. Resumen

En este Capítulo se introdujeron los conceptos básicos de Radar y se presentó la tecnología UWB como una evolución tecnológica del mismo. Se definieron los conceptos de t_{of} y *rango*. A partir de las ecuaciones de éste último se mostró que UWB permite mejorar la precisión de la medición de distancia, y adicionalmente que surgen dos tecnologías de implementación: CW e IR. Se presentaron las señales más comunes utilizadas en sistemas IR-UWB y sus parámetros principales y se discutieron las consideraciones al momento de determinar los parámetros de transmisión como la frecuencia de portadora, ancho de banda y PRF para evitar ambigüedades en la medición de distancia. Se modeló la respuesta de un blanco a una señal UWB y se presentó el método método de expansión de singularidades o *Singularity Expansion Method* (SEM) como camino para clasificación del mismo. Finalmente, se introdujo el problema del clutter y se presentó la técnica de PCA ampliamente utilizada para la mitigación del mismo en la señal recibida.

Capítulo 3

Algoritmos de análisis tiempo-frecuencia para blancos dinámicos

RESUMEN: Este capítulo oficia de introducción a los algoritmos TF y sus enfoques actuales. La Sección 3.3 introduce la notación a utilizar y las condiciones de contorno asociadas al problema a resolver. Adicionalmente, presenta la transformada wavelet o *Wavelet Transform* (WT) y la SST y la entropía de Rènyi como medida de la esparcidad de la representación. En la Sección 3.4 se presentan trabajos actuales y relevantes relacionados con el análisis TF y se aborda la discusión de sus ventajas y desventajas.

3.1. Introducción

El análisis TF es un área central y madura en el procesamiento de señales no estacionarias. Como se mostrará en la Sección 3.2 resulta una herramienta indispensable para el análisis de las señales UWB en escenarios dinámicos. Es una técnica que se utiliza vastamente en una amplia gama de aplicaciones que van desde la medicina [30], [6], hasta las geociencias [31], incluyendo la interferometría [32] y la maquinaria industrial [33].

A diferencia de un análisis espectral estático, el análisis TF permite estudiar la evolución de la distribución en frecuencia de la energía de una señal a lo largo del tiempo. La representación bidimensional obtenida de la energía de la señal en el plano TF permite identificar y/o extraer características de la señal analizada. Particularmente, en este trabajo se focalizará el estudio en la WT, debido a su generalidad y versatilidad y será la base de los algoritmos propuestos.

En la Sección 3.3 se introduce la notación utilizada como así también definiciones y conceptos básicos de la transformada Wavelet. Daubechies en 2011 formalizó el método la SST[34] que es una técnica de realocación en el plano TF capaz de mejorar la *nitidez* de la WT. Este método sirve como base para el algoritmo desarrollado en esta tesis, por lo que se introducirá en la Sección 3.3.3. En base a lo introducido, en la Sección 3.4 se presenta una selección de trabajos actuales y relevantes relacionados con el análisis TF que permite, por un lado analizar el estado del arte en el tema, y por otro oficia como moti-

vación para el desarrollo del algoritmo presentado en este trabajo (Capítulo 4) analizando las ventajas y desventajas de los métodos estudiados.

3.2. Motivación

Integrando lo presentado hasta el momento, resulta interesante analizar la estructura y naturaleza de las señales recibidas en un radar UWB a partir del planteo de un escenario de aplicación concreto. La Figura 3.1 plantea una configuración de ejemplo en un entorno interior que involucra una persona estática y un objeto en movimiento.

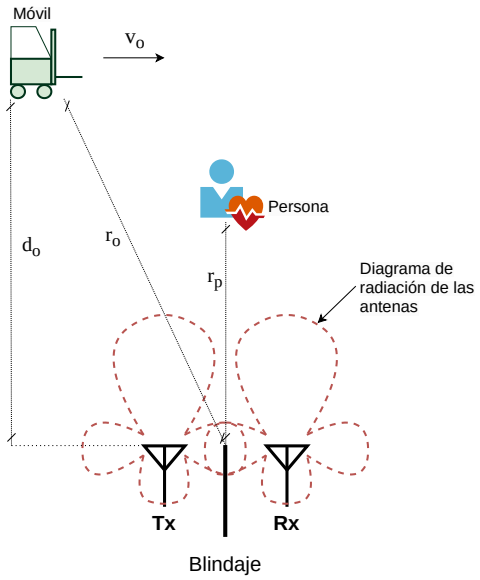


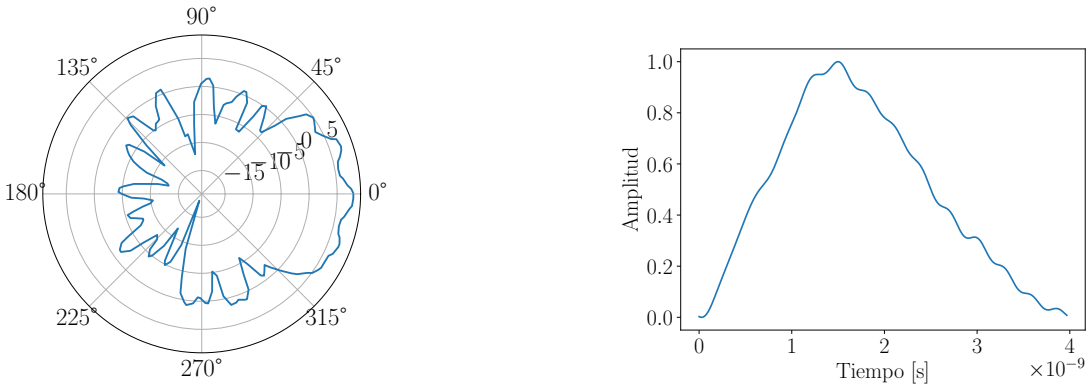
Figura 3.1: Escenario de ejemplo para aplicación de un radar UWB.

A partir de este escenario se puede construir un modelo de simulación acotado para obtener datos aproximados. Los parámetros generales de simulación se muestran en la Tabla 3.1, y fueron seleccionados a partir de valores típicos que se utilizarán en la plataforma desarrollada en el Capítulo 6.

Parámetro	Valor	Descripción
f_s	5,04	Frecuencia de muestreo [GHz]
PRF	7,875	Frecuencia de repetición de pulso [MHz]
f_c	3,78	Frecuencia de la portadora [GHz]
`scanLen`	640	Longitud en muestras de cada scan
`simTime`	10,0	Tiempo de simulación [s]
`nScansSeg`	30	Cantidad de scans por segundo
`aOverSamp`	10,0	Sobre-muestreo analógico
`antShieldGain`	0,01	Coeficiente de atenuación del blindaje entre antenas
SNR	0,128	Relación señal a ruido [veces]

Tabla 3.1: Parámetros de simulación.

Adicionalmente, se utilizaron datos de ganancia de una antena UWB construida[1] y muestras de un pulso en banda base generado en laboratorio mediante FPGA. Dichos datos se muestran en la Figura 3.2.



(a) Ganancia de la antena en dB.

(b) Pulso en banda base.

Figura 3.2: Datos de antena y pulso.

Los parámetros geométricos para un entorno interior se muestran en la Tabla 3.2.

Parámetro	Valor	Descripción
r_p	3,0	Distancia inicial a la persona [m]
d_o	6,0	Distancia a la trayectoria lineal del objeto, en dirección a la persona [m]
v_o	0,5	Velocidad de movimiento del objeto [m/s]
d_{Tx-Rx}	0,3	Distancia entre antenas [m]

Tabla 3.2: Parámetros geométricos.

Con dichos parámetros, la evolución de la distancia de los blancos a la antena se muestra en la Figura 3.3.

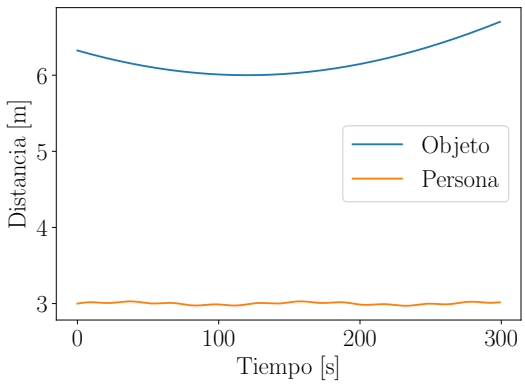


Figura 3.3: Distancia de los blancos a la antena en función del tiempo.

[1]La descripción de la misma se puede encontrar en la Sección 6.2.1.

Para los fines de esta simulación, la persona fue modelada como un blanco estático dónde su posición es muy levemente modulada por dos señales senoidales que representan el movimiento torácico debido a la respiración y al movimiento cardíaco respectivamente. La Tabla 3.3 lista los parámetros físicos asociados a la persona.

Parámetro	Valor	Descripción
m_h	0,003	Amplitud de la señal cardíaca [m]
m_b	0,01	Amplitud de la señal respiratoria [m]
f_h	1,0	Frecuencia cardíaca [Hz]
f_b	0,23	Frecuencia respiratoria [Hz]
$\sigma_{CS,P}$	1	RCS de la persona [m^2]
$\sigma_{CS,M}$	2	RCS del móvil [m^2]

Tabla 3.3: Parámetros físicos.

A partir de estas condiciones, se simuló la propagación de la señal considerando las atenuaciones de espacio libre, el acoplamiento directo de los pulsos y la SNR. El pulso en banda base es modulado a la frecuencia de portadora f_c antes de ser transmitido y luego es demodulado en la antena receptora obteniéndose los canales I y Q. Cada segmento o *frame* de la señal recibida consta de 640 muestras tomadas a f_s a partir del instante en que cada pulso es transmitido. La Figura 3.4 muestra los pulsos recibidos en la antena de dos *frames* para el escenario planteado.

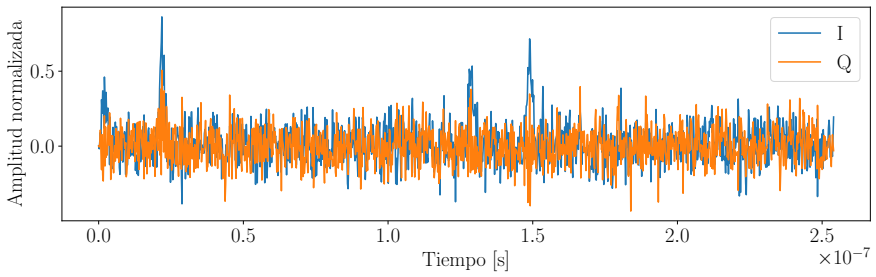


Figura 3.4: Pulsos recibidos en la antena.

Ubicando cada *frame* en una columna de una matriz, se obtiene el radagrama de la Figura 3.5[2] donde se pueden observar las crestas correspondientes al pulso acoplado directamente (a través del blindaje), el pulso reflejado de la persona y el pulso reflejado del objeto móvil para cada tiempo.

Si se muestrea cada frame sobre la cresta correspondiente a la persona y se analiza la fase, tal como fue planteado en el Capítulo 2, se obtiene la señal de la Figura 3.6a. Si se aplica la transformada de Fourier a dicha señal, se puede observar en la Figura 3.6b que contiene información sobre la frecuencia cardíaca y la respiración de la persona (1 y 0,23 Hz respectivamente).

Este ejemplo introductorio resulta de utilidad para analizar cierta información contenida en las señales UWB y su estructura. La simulación simplificada permite observar

[2]En la versión electrónica de este documento la imagen es vectorial permitiendo aumentar el *zoom* sin pérdida de detalles.

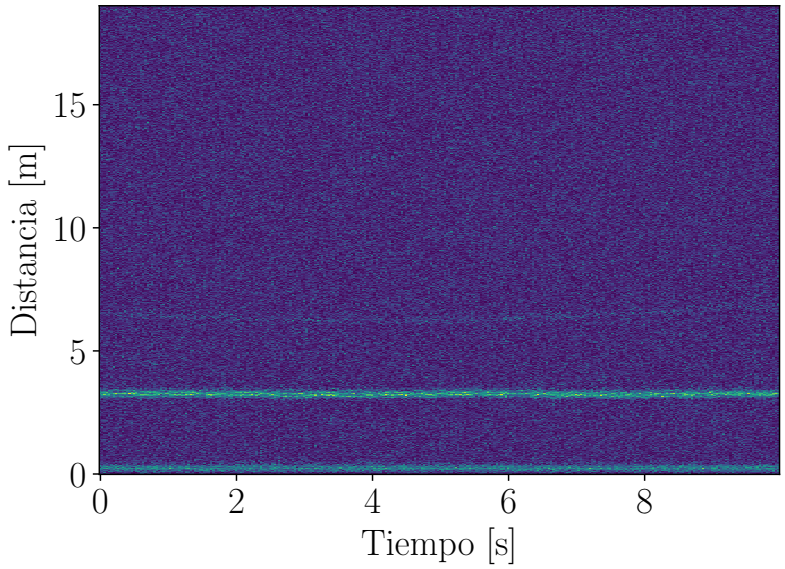
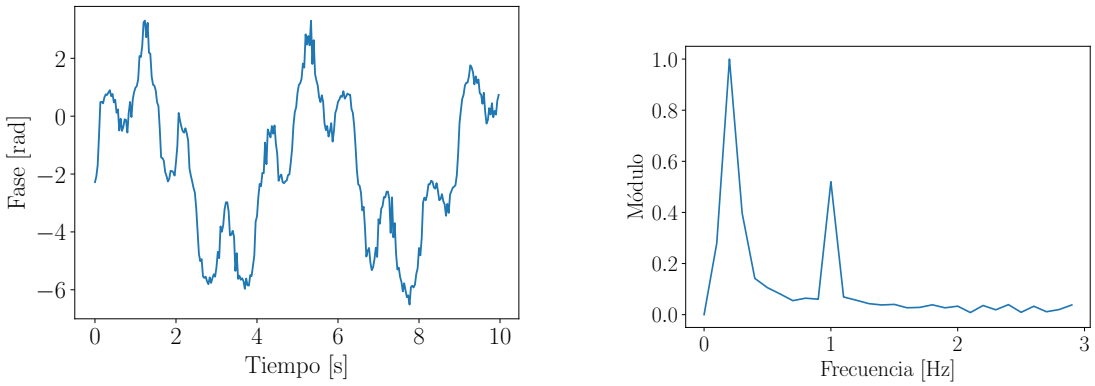


Figura 3.5: Radagrama de la señal recibida.



(a) Fase del pulso recibido para la persona.

(b) Detalle de la FFT de la señal de fase.

Figura 3.6: Análisis de la señal reflejada de la persona.

que bajo condiciones típicas de un radar UWB es posible recuperar información dinámica de los blancos. Sin embargo, en este caso las frecuencias embebidas en el blanco son constantes en el tiempo. Para aplicaciones de la vida real dónde, por ejemplo, se desea hacer un seguimiento de la frecuencia cardíaca y/o respiratoria de una persona es necesario realizar un análisis TF de la señal recibida.

3.3. Notación y contexto

En esta sección se introduce la notación utilizada como así también las condiciones de contorno asociadas al problema a resolver. En este sentido, se define el conjunto de señales admisibles y se presentan la WT y la SST.

3.3.1. Señales admisibles

Siguiendo la estructura de [22] y en concordancia con [34], definimos una señal admisible $s(t)$ como una superposición de M funciones de tipo Modo Intrínseco o *Intrinsic-Mode Type* (IMT) *bien separadas* con fase instantánea $\phi(t)$, parámetro de suavidad $\varepsilon > 0$ y separación $d > 0$ de la siguiente manera:

$$s(t) = \sum_{j=1}^{M} s_j(t) = \sum_{j=1}^{M} A_j(t) e^{i\phi_j(t)} \tag{3.1}$$

donde $A_j \in L^\infty \cap C^1$, $\quad \phi_j \in C^2$, $\quad \phi_j', \phi_j'' \in L^\infty$, y $\phi_j'(t) = \frac{d\phi_j(t)}{dt} = \omega_j(t)$.
Además, $\inf_t \phi_j'(t) > 0$, y $\forall t,\ |A_j'(t)| \leq \varepsilon |\phi_j'(t)|$, $|\phi_j''(t)| \leq \varepsilon |\phi_j'(t)|$, y finalmente

$$\frac{\phi_j'(t) - \phi_{j-1}'(t)}{\phi_j'(t) + \phi_{j-1}'(t)} \geq d. \tag{3.2}$$

Interpretando estos requerimientos se puede observar que las señales admisibles están compuestas por una suma finita de tonos modulados en amplitud y fase, cuyas amplitudes $A_j(t)$ varían más lentamente que las fases $\phi_j(t)$. Particularmente, la relación entre dichas variaciones está acotada por ε. Además, la separación relativa instantánea entre los tonos para cualquier tiempo t está limitada inferiormente por d. A esta última condición se hace referencia en lenguaje coloquial como componentes *bien separadas*.

Este tipo de señales pueden ser encontradas en una enorme cantidad de aplicaciones y/o problemas de la vida real [35][36][37][38]. Las condiciones sobre las señales de entrada pueden ser relajadas a expensas de una complejización de los algoritmos que se discutirá en las secciones siguientes.

3.3.2. Wavelet Transform

Sea $\psi(t)$ una Wavelet madre cuya energía está concentrada en el entorno de una frecuencia positiva ω_0^ψ y decae rápidamente a valores despreciables en otros puntos, es decir, $\psi(t)$ es una función analítica.

La transformada wavelet continua o *Continuous Wavelet Transform* (CWT) de $s(t)$ es

$$W_s(a,b) = \int_{-\infty}^{\infty} s(t) a^{-1/2} \overline{\psi\left(\frac{t-b}{a}\right)} dt, \tag{3.3}$$

donde $\overline{\psi(t)}$ es el complejo conjugado de $\psi(t)$, y $\{a,b\}$ son las variables de escala y de tiempo respectivamente. Típicamente, la WT es expresada en términos de la escala a y el tiempo b. Por cuestiones de claridad y coherencia con la implementación, en este trabajo típicamente se expresará en función de la frecuencia ω sin pérdida de generalidad. La relación entre escala y frecuencia es: $a(\omega) = \omega_0^\psi / \omega$. $W_s(a,b)$ indica la concentración de la energía de la señal en el plano escala-tiempo [39], o equivalentemente $W_s(\omega,b)$ en el plano frecuencia-tiempo.

La implementación de la transformada en una plataforma de cómputo implica considerar un conjunto discreto de escalas $\{a_m\}$ y de tiempos $b_n = nT_s$ con $m,n \in \mathbb{N}$. Luego, la contraparte discreta de (3.3) es

$$W_s(a_m,b_n) = \sum_{l=-\infty}^{\infty} s(l) a_m^{-1/2} \overline{\psi\left(\frac{l-b_n}{a_m}\right)}. \tag{3.4}$$

3.3.3. *Synchrosqueezing Transform*

Si bien la WT es una herramienta muy versátil para el análisis de señales no estacionarias, claramente la representación TF de la señal analizada es afectada por la concentración de energía de la wavelet madre elegida. Para superar esta limitación, la SST utiliza información *a priori* de la señal a analizar. Particularmente, asume que la señal se enmarca en las condiciones planteadas en la Sección 3.3.1. De esta manera, por ejemplo si $s(t) = Acos(\omega t)$, utilizando el teorema de Plancherel se puede escribir

$$\begin{aligned} W_s(a,b) &= \frac{1}{2\pi}\int \hat{s}(\xi) a^{1/2} \overline{\hat{\psi}(a\xi)} e^{ib\xi} \mathrm{d}\xi \\ &= \frac{A}{4\pi}\int [\delta(\xi-\omega)+\delta(\xi+\omega)] a^{1/2} \overline{\hat{\psi}(a\xi)} e^{ib\xi} \mathrm{d}\xi \\ &= \frac{A}{4\pi} a^{1/2} \overline{\hat{\psi}(a\omega)} e^{ib\omega} \end{aligned} \tag{3.5}$$

donde $\hat{s}(\xi)$ es la transformada de Fourier de $s(t)$, y $\delta(\xi)$ es la función delta de Dirac.

Si se considera la siguiente función de estimación de la frecuencia instantánea local

$$\omega_s(a,b) = \Re\left\{ -i(W_s(a,b))^{-1} \frac{\partial}{\partial b} W_s(a,b) \right\}, \tag{3.6}$$

donde $\Re\{.\}$ es la parte real de un número complejo, para el ejemplo anterior se obtiene $\omega_s(a,b) = \omega$.

Luego, utilizando (3.6) la SST se define como

$$S_s^{\delta}(w,b) := \int_{A_{\tilde{\varepsilon},s}(b)} W_s(a,b) a^{-3/2} \delta(\omega_s(a,b)-w) da \tag{3.7}$$

$$A_{\tilde{\varepsilon},s}(b) := \{a \in \mathbb{R}_+ ; |W_s(a,b)| > \tilde{\varepsilon}\}$$

$A_{\tilde{\varepsilon},s}(b)$ es el conjunto de puntos del plano TF donde la energía de $W_s(a,b)$ es mayor a un umbral $\tilde{\varepsilon}$ para un tiempo dado[3] b.

Realizar la integral de (3.7) sobre $A_{\tilde{\varepsilon},s}(b)$ para un tiempo dado b permite evitar situaciones donde el problema resulta mal condicionado en (3.6).

Esta transformación puede ser interpretada como el remapeo de las contribuciones de un determinado tono sobre diferentes escalas, hacia la escala correspondiente a la frecuencia del tono. En otras palabras, la energía de un tono que es dispersada por la propia Wavelet sobre el eje de escalas es concentrada nuevamente mediante la SST.

Similarmente, se puede definir la versión discreta de (3.7) para un conjunto finito de frecuencias de análisis $\{w_0, \cdots w_{K-1}\}$,

$$S_s(w_k,b_n) := \frac{1}{(\Delta w)_k} \sum_{\forall a_m \in A_{\tilde{\varepsilon},s}(b_n)} \mathbb{1}_{\Delta_k}(\omega_s(a_m,b_n)-w_k) W_s(a_m,b_n) a_m^{-3/2} (\Delta a)_m \tag{3.8}$$

[3] En el teorema 3.3 de [34], se puede observar la relación entre ε y $\tilde{\varepsilon}$, siendo $\tilde{\varepsilon} = \varepsilon^{1/3}$

donde $(\Delta a)_m = (a_{m+1} - a_m)/2$, $(\Delta w)_k = (w_{k+1} - w_k)/2$, y $\mathbb{1}_{\Delta_k}$ es la función indicatriz del intervalo $\Delta_k = [-\frac{(\Delta w)_k}{2}, \frac{(\Delta w)_{k+1}}{2}]$.

Aquí la suma es realizada sobre el siguiente conjunto de escalas

$$A_{\tilde{\varepsilon},s}(b_n) := \{a_m : |W_s(a_m, b_n)| \geq \tilde{\varepsilon}\}.$$

Dado que $\|W_s(a_m, b_n)\| \geqslant \tilde{\varepsilon} > 0$ para $a_m \in A_{\tilde{\varepsilon},s}(b_n)$, (3.9) está bien definida. En general, se considerará que los intervalos Δ_k no son simétricos respecto del origen. Esta consideración se debe a que un conjunto de frecuencias que no son equidistantes generará intervalos no simétricos, condición que encontraremos en el desarrollo del algoritmo del Capítulo 4.

Para computar $\omega_s(a_m, b_n)$, en el caso discreto se debe reemplazar la derivada en (3.6) por una diferencia finita. Por lo tanto, si se analiza una señal de la forma $s(t) = Ae^{i\omega t}$ con $s(n) = s(nT_s)$, se obtiene[40][4],

$$\begin{aligned}
\frac{\partial_t W_s(a,b)}{\mathrm{i}W_s(a,b)} &\approx \frac{W_s(a_m, b_{n+1}) - W_s(a_m, b_n)}{\mathrm{i}W_s(a_m, b_n)} \\
&= \frac{\sum_{\ell=-\infty}^{\infty} A\left(\mathrm{e}^{\mathrm{i}\omega(\ell+n+1)T_s} - \mathrm{e}^{\mathrm{i}\omega(\ell+n)T_s}\right) a^{-1/2}\overline{\psi\left(\frac{\ell-b_n}{a_m}\right)}}{\mathrm{i}T_s \sum_{\ell=-\infty}^{\infty} A\mathrm{e}^{\mathrm{i}\omega(\ell+n)T_s} a^{-1/2}\overline{\psi\left(\frac{\ell-b_n}{a_m}\right)}} \\
&= \frac{\mathrm{e}^{\mathrm{i}\omega T_s} - 1}{\mathrm{i}T_s} = \omega \cdot \mathrm{e}^{\mathrm{i}\frac{\omega}{2}T_s} \operatorname{sinc}(\frac{\omega}{2}T_s).
\end{aligned}$$

De este último resultado se puede observar que usando la función de reasignación (3.6) en su forma discreta introduce un sesgo en la estimación de ω, ya que para este caso debería resultar $\omega_s(a_m, b_n) = \omega$. Para evitar este sesgo, se puede adoptar la fórmula de reasignación usada en [41] que surge de la derivada continua de la fase del espectro como se propone en [40]:

$$\begin{aligned}
\partial_t \arg\left[W_s(a,b)\right] &\approx \frac{\arg\left[W_s(a_m, b_{n+1})\right] - \arg\left[W_s(a_m, b_n)\right]}{T_s} \\
&= \frac{1}{T_s}\arg\left(\frac{\sum_{\ell=-\infty}^{\infty} A\mathrm{e}^{\mathrm{i}\omega(\ell+n+1)T_s} a^{-1/2}\overline{\psi\left(\frac{\ell-b_n}{a_m}\right)}}{\sum_{\ell=-\infty}^{\infty} A\mathrm{e}^{\mathrm{i}\omega(\ell+n)T_s} a^{-1/2}\overline{\psi\left(\frac{\ell-b_n}{a_m}\right)}}\right) = \frac{\arg\left(\mathrm{e}^{\mathrm{i}\omega T_s}\right)}{T_s} = \omega
\end{aligned}$$

donde $arg(x)$ devuelve el argumento de x para cualquier $x \in \mathbb{C}$. Por lo tanto, se redefinirá $\omega_s(a_m, b_n)$ como

$$\omega_s(a_m, b_n) = T_s^{-1} arg\left(\frac{W_s(a_m, b_{n+1})}{W_s(a_m, b_n)}\right), \tag{3.9}$$

utilizando esta función de reasignación desde ahora en adelante salvo que se aclare lo contrario.

Se puede observar que este enfoque produce la frecuencia exacta (sin sesgo) cuando se utilizan diferencias discretas en lugar de derivadas continuas. Resulta interesante también

[4]En este caso por claridad se hará un abuso de anotación usando ∂_t equivalentemente a $\frac{\partial}{\partial t}$.

notar que este operador $\omega_s(a_m, b_n)$ podría proveer una buena estimación de la frecuencia instantánea cuando el análisis TF es realizado mediante $S_s(w_k, b_n)$ o $W_s(a_m, b_n)$, y los máximos del espectro de la wavelet son conocidos para cada tiempo b. Conociendo dichos máximos (o crestas) y accediendo a las coordenadas correspondientes de la matriz $\omega_s(a_m, b_n)$ se puede extraer el valor de frecuencia instantánea. Esto podría no ser factible cuando solamente la representación TF es posteriormente procesada, por ejemplo, para extraer características como en [42], y no se dispone de información sobre $\omega_s(a_m, b_n)$. En el *toolbox* implementado en este trabajo, se provee al usuario de esta matriz para ampliar la versatilidad del mismo.

A modo ilustrativo la Figura 3.7 muestra la representación TF para una señal *chirp* cuadrática dual (dos componentes) con frecuencias iniciales de 8 y 2 Hz y frecuencias vértice de 5 y 4 Hz para sus componentes respectivamente. Los parámetros utilizados son $K = 64$ frecuencias de análisis, una frecuencia de muestreo $f_s = \frac{1}{T_s} = 400Hz$, una duración de $12s$ y una WT en base a una Wavelet madre tipo Morlet con $\omega_0^{\Psi} = \frac{2\pi}{T_s}$, con un análisis entre 0,1 y 11 Hz.

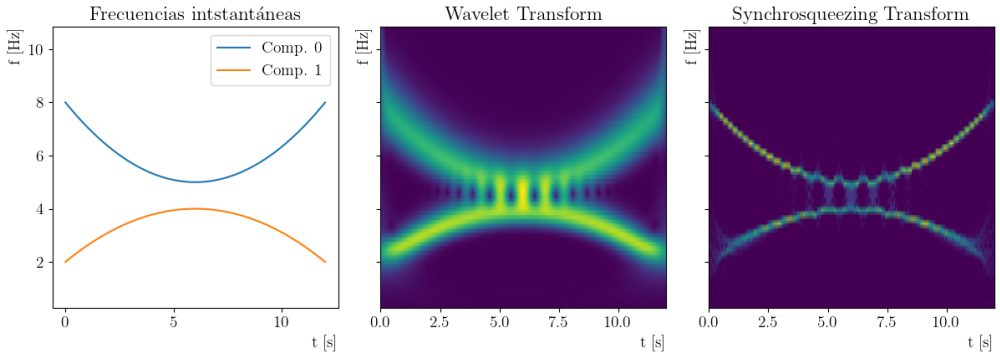


Figura 3.7: Representación TF para una señal *chirp* dual. Frecuencias instantáneas reales (izquierda), WT (centro) y SST (derecha).

De dicha figura se puede observar que si bien la WT es capaz de identificar las dos componentes individuales de la señal, cuando ambas se acercan la SST permite una mejor localización y discriminación de las mismas en el plano TF.

Adicionalmente, si se realiza el análisis para la misma señal que el caso anterior pero variando la cantidad de frecuencias K para la SST, se puede observar un aumento de la *nitidez* a medida que K aumenta. Por ejemplo, para $K = \{16, 32, 64, 128\}$ el resultado se puede apreciar en la Figura 3.8

3.3.4. Entropía de Rènyi

Los ejemplos presentados en la sección precedente introducen de manera visual la mejora en la resolución TF que provee la SST en comparación con la WT y a su vez como un aumento de K genera representaciones más *nítidas*. Sin embargo es necesario contar con una métrica que permita cuantificar la concentración de energía en el plano TF de una señal. Tradicionalmente, la entropía de Shannon es la métrica esperada para obtener esta medida. Sea $D_s(\omega, t)$ una distribución genérica (normalizada) en el plano TF de una señal $s(t)$, entonces la entropía de Shannon se calcula como

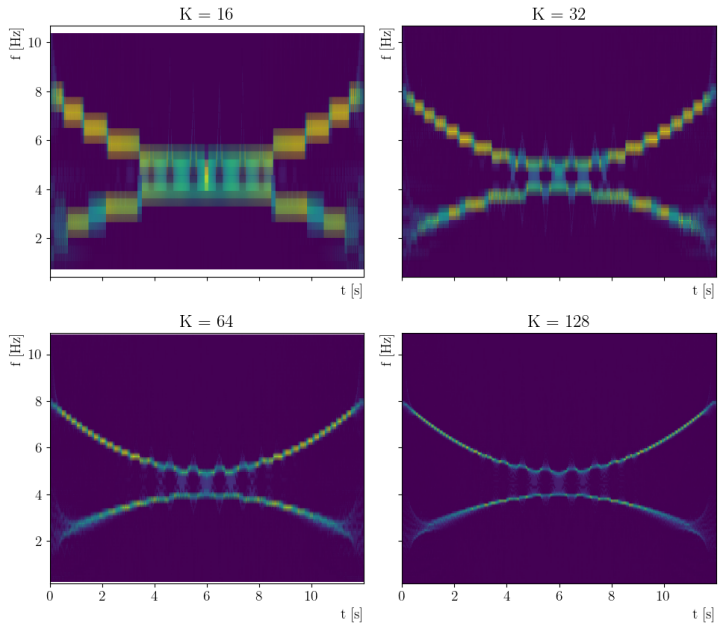


Figura 3.8: Representación TF de la SST para la misma señal de la Figura 3.7 con diferentes K.

$$H_s(D_s) := -\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} D_s(\omega,t)\log_2(D_s(\omega,t))dtd\omega. \tag{3.10}$$

En la práctica, la distribución D_s puede ser negativa en algunos puntos del plano (ω,t) [43], por lo que el término que implica log_2 puede no ser computable. Una solución a este problema es la utilización de la entropía de Rènyi generalizada que se calcula como

$$H_\alpha(D_s) := \frac{1}{1-\alpha}\log_2\left(\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}[D_s(\omega,t)]^\alpha dtd\omega\right). \tag{3.11}$$

Típicamente se utiliza como métrica para la concentración de la energía de una señal en el plano TF la entropía de Rènyi de orden 3 [40][44][45][5].

En el caso discreto y con distribuciones no normalizadas, la entropía de Rènyi se calcula como

$$H_\alpha(D_s) := \frac{1}{1-\alpha}\log_2\left(\sum_{n=0}^{N-1}\sum_{k=0}^{K-1}\left[\frac{|D_s(\omega_k,t_n)|^2}{||D_s||}\right]^\alpha\right) \tag{3.12}$$

con

$$||D_s|| = \sum_{n=0}^{N-1}\sum_{k=0}^{K-1}|D_s(\omega_k,t_n)|^2.$$

Cuando se analizan los ejemplos presentados se obtiene, para el análisis de la Figura 3.7 los valores de la Tabla 3.4.

Un valor menor en la entropía implica un menor número de frecuencias en las cuales se distribuye la energía, es decir, una representación más concentrada. En este ejemplo en particular, claramente la SST presenta una mejora respecto de la WT.

[5]Notar que para Shannon $\alpha \to 1$

Método	Entropía de Rènyi
WT	16,960
SST	13,028

Tabla 3.4: Entropía de Rènyi para el análisis de la Figura 3.7.

Para el caso de estudio de la Figura 3.8 se obtienen los valores de la Tabla 3.5.

K	Entropía de Rènyi
16	13,0989
32	13,0569
64	13,0284
128	13,0561

Tabla 3.5: Entropía de Rènyi para el análisis de la Figura 3.8.

De la mencionada Tabla 3.5 se pueden observar varias situaciones. Por un lado, un aumento de K genera una disminución en la entropía de Rènyi en los primeros tres casos, aunque en magnitudes mucho más modestas que ante un cambio de transformada. Por otro lado, en el caso de $K = 128$ la entropía vuelve a aumentar. Esto se debe a que en este último caso se superó el límite teórico de resolución asociado a la wavelet seleccionada (ver Sección 4.3.1). En estas condiciones el agregado de frecuencias de análisis adicionales no proveen un aumento en la concentración de energía y por lo tanto simplemente se obtiene un número mayor de frecuencias con contenido de energía (numerador de (3.12)), anulando el aumento del número total de puntos del plano TF (denominador de (3.12)). Sin embargo, el aumento de K puede resultar en una disminución del error de estimación de la frecuencia instantánea aunque no se refleje en una mejora en la entropía de Rènyi. Esto último se debe a que la frecuencia instantánea típicamente se estima a partir de las crestas de la representación TF, asociándola a la frecuencia de análisis w_k correspondiente. Para cuantificar estas mejoras no consideradas por la Entropía de Rènyi se utilizará adicionalmente el MSE de la estimación de las frecuencias instantáneas como medida.

El problema de seleccionar un K adecuado resulta entonces un problema nada trivial. Un aumento de este valor puede proveer mejoras en la resolución y capacidad de discriminar componentes individuales a expensas de una complejidad computacional mayor. Adicionalmente desde otro enfoque complementario es posible analizar que al superar el límite de resolución teórico, si bien no se obtiene una mayor concentración de la energía en la representación obtenida, la redundancia de información puede producir mejoras en el resultado final si la misma va a alimentar a otro proceso. Este caso puede darse, por ejemplo, al alimentar una red neuronal convolucional con la representación TF calculada [42].

3.4. Trabajos relevantes

En los años recientes se ha hecho un gran esfuerzo para aumentar la resolución del análisis TF, a través del uso de información *a priori* sobre la estructura de las señales de entrada. Además, se ha extendido la técnica de *Synchrosqueezing* a la Transformada de

Fourier de Tiempo Corto [46]. Esta última extensión del algoritmo se puede pensar como un caso particular de la WT donde las Wavelets están definidas por la ventana utilizada y se centran en las frecuencias armónicas de Fourier. Dado que el objetivo del algoritmo desarrollado es adaptar la grilla de frecuencias de análisis, como fue mencionado anteriormente nos centraremos en el enfoque Wavelet que resulta más general y abarcativo.

3.4.1. SST de orden superior

Una versión mejorada se obtuvo originalmente en [47] y luego en [48] donde se calcula una función de mapeo de frecuencia a escala diferente. La idea se centra en definir una función de remapeo $\omega_s^{[G]}(a,b)$ de orden (G) que relaje las restricciones sobre $\phi(t)$. Para el caso donde $G=2$, de la misma manera que en (3.6), se define la función de remapeo para tiempos como

$$t_s(a,b) = b - \frac{\partial_a W_s(a,b)}{\mathrm{i}2\pi W_s(a,b)} \tag{3.13}$$

para luego definir el operador de modulación de segundo orden para frecuencias complejas como

$$\tilde{q}_s(a,b) = \frac{\partial_b \omega_s(a,b)}{\partial_b t_s(a,b)} = \frac{\partial_b \left(\frac{\partial_b W_s(a,b)}{W_s(a,b)} \right)}{2\pi\mathrm{i} - \partial_b \left(\frac{\partial_a W_s(a,b)}{W_s(a,b)} \right)} \tag{3.14}$$

y la función de remapeo de segundo orden

$$\omega_s^{[2]}(a,b) = \begin{cases} \omega_s(a,b) + \tilde{q}_s(a,b)\,(b - t_s(a,b)) & \text{si } \partial_b t_s(a,b) \neq 0 \\ \omega_s(a,b) & \text{de otra forma} \end{cases} \tag{3.15}$$

finalmente, $\omega_s^{[2]} = \Re\{\omega_s^{[2]}\}$. Con esta nueva función de remapeo se prueba en [49] que $\omega_s^{[2]}(a,b) = \omega(t)$ cuando $s(t)$ tiene una forma más compleja como ser una *chirp* modulada por una Gaussiana. La SST de segundo orden se calcula simplemente reemplazando $\omega_s(a,b)$ por $\omega_s^{[2]}(a,b)$ en (3.7).

Este enfoque resulta mucho más preciso en su remapeo de frecuencias, especialmente para señales que no cumplen con las condiciones enunciadas en la Sección 3.3.1. Sin embargo, introduce mucha más carga computacional: requiere el cálculo adicional de la diferenciación de $W_s(a,b)$ respecto de a, luego la diferenciación respecto del tiempo b del operador $\omega_s(a,b)$ y de $t_s(a,b)$, respectivamente. Adicionalmente, el cociente introducido en (3.14) puede introducir problemas numéricos en condiciones adicionales.

3.4.2. Diferentes métodos de realocación en tiempo-frecuencia

Otras técnicas relacionadas han sido introducidas últimamente. Las mismas se basan en diferentes formas de remapear el plano (a,b) al plano (ω,t).

3.4.2.1. Transformada Synchrosqueezing reasignada en tiempo

La Transformada Synchrosqueezing reasignada en tiempo o *Time-reassigned Synchrosqueezing Transform* (TSST) aprieta o *squeezes* en el dominio del tiempo, en lugar

del de la frecuencia, lo que la hace más adecuada para analizar señales impulsivas [50]. Utilizando la misma idea de que varios tiempos pueden contribuir al mismo instante, debido a la dispersión temporal introducida por las wavelets, la operación de reasignación se realiza sobre el eje b en lugar de a, dando origen a la siguiente función de remapeo:

$$\tau_s(a,b) = \begin{cases} \Re\left\{\frac{i\partial_a W_s(a,b)}{W_s(a,b)}\right\}, & |W_s(a,b)| > \tilde{\varepsilon} \\ 0 \quad , & |W_s(a,b)| \leqslant \tilde{\varepsilon} \end{cases} \tag{3.16}$$

donde $\tilde{\varepsilon}$ cumple el mismo parámetro de umbral que en (3.7). Luego, la función de acumulación equivalente a (3.7) se define como

$$TSST_s(a,t) = \int_{B_{\tilde{\varepsilon},s}(a)} W_s(a,b)\delta(t-\tau_s(a,b))db \tag{3.17}$$

con

$$B_{\tilde{\varepsilon},s}(a) := \{a; |W_s(a,b)| > \tilde{\varepsilon}\}$$

Claramente este método proporciona una mejor resolución temporal para señales de naturaleza impulsiva, en detrimento de resolución en el eje de frecuencias.

3.4.2.2. Método de reasignación

Tanto SST como TSST son casos particulares del Método de Reasignación o *Reassign Method* (RM) que opera en ambos dominios[50], frecuencia y tiempo. Este método busca reasignar el punto de energía (a,b) a su centro de gravedad correspondiente $(\omega_s(a,b), \tau_s(a,b))$, por lo tanto su función de acumulación resulta:

$$RM_s(t,\omega) = \int_{-\infty}^{+\infty}\int_{0}^{+\infty} |W_s(a,b)|^2 \delta(\omega-\omega_s(a,b))\delta(t-\tau_s(a,b))da\,db \tag{3.18}$$

con $\omega_s(a,b)$ y $\tau_s(a,b)$ como en (3.6) y (3.16) respectivamente.

3.4.2.3. *Synchro-Extracting Transform* (SET)

Con el objetivo de proporcionar una mejor inmunidad al ruido, la SET reemplaza la acumulación de las frecuencias remapeadas por una extracción. Esto es, en lugar de acumular la energía dispersa en el plano TF correspondiente a un modo en su partición del espectro correspondiente, sólo mantiene la energía dentro de su propio segmento del espectro e ignora el resto [51][52]. En este caso, la función de remapeo se mantiene como en (3.9) y la función de acumulación se reemplaza por una función llamada de "extracción". La transformada se define entonces como

$$SET_s(\omega,b) = W_s(a,b).\delta(\omega_s(a,b)-\omega) \tag{3.19}$$

Puede notarse que las técnicas presentadas en esta sub-sección (3.4.2) involucran solamente variantes a la función de remapeo del plano TF. La variante elegida (SST, TSST, RM o SET) dependerá de la naturaleza de la señal y de los objetivos del análisis. Salvo por la ausencia de la acumulación en el método SET, la carga computacional de estos métodos es similar. En el Capítulo 4 se presentará un algoritmo que permite utilizar cualquiera de estos métodos de reasignación en conjunto con un esquema adaptativo, basado en el concepto de multirresolución abordado a continuación.

3.4.3. Análisis multirresolución

Otro camino de mejora ampliamente explorado para estas técnicas es el análisis de multirresolución a través de la adaptación de datos. De esta manera, se pueden utilizar grillas TF no uniformes que resultan en representaciones más nítidas [53] de zonas relevantes en el plano TF.

3.4.3.1. División adaptativa del plano TF y la ventana (*Short-Time Fourier Transform* (STFT))

En [40], el plano se divide en súper-bloques o *súper-tiles*. Un lote de datos se analiza con W STFT correspondientes a W ventanas diferentes. Luego, para cada *tile*, se calcula la entropía de Rènyi para cada ventana y se selecciona la que tenga la mejor resolución para ese bloque, descartando cualquier otra. El algoritmo básico se resume en el Algoritmo 1 con la notación utilizada en el presente trabajo.

Algoritmo 1 Algoritmo adaptativo presentado en [40]

1: Seleccionar las W ventanas a utilizar denominadas h^w. Seleccionar los puntos de muestreo del plano TF dados por el tamaño del salto entre ventanas H (en nuestro caso 1) y L^w el tamaño de la transformada rápida de Fourier o *Fast Fourier Transform* (FFT) (en nuestro caso equivalente K).
2: Para cada h^w calcular $STFT_s^{h^w}[n,m]$ con H y L^w.
3: Fijar los parámetros $A,B \in \mathbb{N}$ que determinan el tamaño de los super-tiles $\square_{r,s}$ en el plano TF. Con:
$\square_{r,s} = [r\tilde{A},(r+1)\tilde{A}) \times [s\tilde{B},(s+1)\tilde{B})$
donde $\tilde{A} := AH\Delta t$, $\tilde{B} := \frac{B}{(F_{min}\Delta t)}$ y $F_{min} := min_w\{F^w\}$ la mínima frecuencia de análisis.
4: Para cada $\square_{r,s}$ y h^w calcular la energía total como $E_{r,s}^{h^w} = \sum_{(n,m)\in\square_{r,s}} |STFT_s^{h^w}[n,m]|^2$.
5: Para cada $\square_{r,s}$ y h^w calcular la entropía de Rènyi de $STFT_s^{h^w}$ según (3.12).
6: Para cada *super-tile* $\square_{r,s}$ seleccionar la ventana h^w que minimice la entropía de Rènyi.

Una versión mejorada del algoritmo también se introduce en [40] que perturba cada *super-tile* y calcula el promedio de la entropía de Rènyi para cada ventana, seleccionando la que minimiza dicho promedio. Esto le otorga una mayor inmunidad al ruido. Se puede apreciar que este algoritmo pone el foco en mejorar notablemente la resolución TF, sin embargo requiere calcular la STFT para cada ventana y para cada *super-tile*, implicando un esfuerzo computacional considerable, principalmente porque muchos resultados obtenidos se descartan si no minimizan la entropía de Rènyi. Adicionalmente, poniendo el foco sólo en la entropía de Rènyi, no se consideran situaciones dónde el MSE de la estimación de la frecuencia instantánea puede ser más relevante, como fue planteado anteriormente.

3.4.3.2. Adaptación paramétrica de la Wavelet

En un enfoque diferente, los autores de [54][55] proponen adaptar la Wavelet en función de la señal a procesar para aumentar la concentración de energía. Para este caso, los autores también proporcionan una implementación en Matlab[56].

Si $s(t) = \sum_{j=1}^{M} A_j e^{i\phi_j(t)}$ con $\phi_j(t) = 2\pi c_j t$, entonces utilizando el resultado de (3.5) podemos obtener

$$W_s(a_m, b_n) = \sum_{j=1}^{M} A_j \overline{\widehat{\psi}(a_m c_j)} e^{i2\pi b_n c_j}. \tag{3.20}$$

A partir del soporte de ψ, esto es, la región del plano TF donde no se cumple que $\psi \approx 0$, se puede definir una condición de separabilidad de los tonos. Sea

$$\widehat{\psi}_\sigma(\xi) = e^{-2\sigma^2\pi^2(\xi-\mu)^2} - e^{-2\sigma^2\pi^2\left(\xi^2+\mu^2\right)} \tag{3.21}$$

la transformada de Fourier de una wavelet Morlet con σ el parámetro que controla la concentración de energía de la misma en el plano TF. Para analizar la región de soporte basta con analizar la envolvente de la wavelet que se puede definir como una función Gaussiana $g(t) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{t^2}{2}}$. O equivalentemente por su transformada de Fourier como

$$\widehat{g}(\xi) = e^{-2\pi^2\xi^2}. \tag{3.22}$$

Esto permite definir un umbral τ_0 a partir del cual se considera que la energía de la wavelet es despreciable. Si $\widehat{g}(\beta) = \tau_0$, entonces

$$\beta = \frac{1}{2\pi}\sqrt{2\ln(\frac{1}{\tau_0})}. \tag{3.23}$$

Luego, el soporte de la wavelet se define como $[-\beta, \beta]$.

Li et al. proponen modificar el parámetro σ de la wavelet en función del tiempo, es decir $\sigma \to \sigma(b)$, y adaptarlo para cada tiempo b en función de minimizar la entropía de Rènyi definida en (3.12). Esto es,

$$\sigma_u(b) = \arg\min_{\sigma \geqslant \frac{\beta}{\mu}} \{H_\alpha[W_s^\sigma(a,b)]\}. \tag{3.24}$$

La notación introducida $W_s^\sigma(a,b)$ indica que la WT depende del parámetro $\sigma(b)$.

Considerando que la j-ésima componente de la señal $s(t)$ cae en una región del plano TF comprendida entre

$$\frac{\mu - \frac{\beta}{\sigma}}{c_j} \leqslant a \leqslant \frac{\mu + \frac{\beta}{\sigma}}{c_j}$$

es posible definir que dos componentes son separables si sus regiones de soporte no se solapan. Es decir, si

$$\frac{\mu + \frac{\beta}{\sigma}}{c_j} \leqslant \frac{\mu - \frac{\beta}{\sigma}}{c_{j-1}} \tag{3.25}$$

o en forma equivalente

$$\sigma \geqslant \frac{\beta}{\mu}\frac{c_j + c_{j-1}}{c_j - c_{j-1}} \tag{3.26}$$

donde se puede observar la similitud con los requerimientos de las señales admisibles introducidos en la Sección 3.3.1.

Resulta importante notar que esta condición de separabilidad utilizada para ajustar el parámetro de la wavelet depende de la cantidad y frecuencias de los componentes verdaderos de la señal de entrada que son desconocidos. Para ello el algoritmo estima los mismos a partir de un análisis de crestas para cada tiempo b. Los límites entonces se transforman en

$$h_j = \frac{\mu + \frac{\beta}{\sigma}}{\widehat{c}_j}, \qquad g_j = \frac{\mu - \frac{\beta}{\sigma}}{\widehat{c}_j}$$

siendo $\widehat{c}_j$ la estimación de c_j. El conjunto de los intervalos de soporte para $W_s^{\sigma}(a_m, b_n)$ entonces queda

$$\mathbf{S} = \{[g_1, h_1], [g_2, h_2], ..., [g_v, h_v]\} \tag{3.27}$$

Particularmente, los mismos no se solapan si

$$h_j \leqslant g_{j+1}, \qquad \text{para } j = 1, 2, ..., v-1. \tag{3.28}$$

Para el cómputo, las variaciones del parámetro $\sigma_u(b)$ se discretizan uniformemente con un intervalo de muestreo de $\Delta\sigma = \sigma_{j-1} - \sigma_j$ con $j = 1, 2, ..., n$. Finalmente, el método propuesto por Li et al. se presenta en el Algoritmo 2.

Algoritmo 2 Algoritmo adaptativo presentado en [54]

1: Sea b un tiempo determinado. Encontrar σ_u según (3.24) con $\sigma \in \{\sigma_j, j = 1, 2, ..., n\}$.
2: Sea **S** el conjunto definido en (3.27) con $\sigma = \sigma_u$. Sea $z = \sigma_u$.
3: **if** Los intervalos de soporte determinados por σ_u cumplen con la condición de separabilidad en (3.28) **then**
4: Continuar en 8
5: **else**
6: **goto** 15
7: **end if**
8: $\sigma = z - \Delta\sigma$
9: **if** La cantidad de intervalos v determinados por z no cambia, $\sigma \geqslant \sigma_n$ y se mantiene la condición de separabilidad en (3.28) **then**
10: Continuar en 14
11: **else**
12: **goto** 15
13: **end if**
14: Repetir 8 con $z = \sigma$
15: $C(b) = z$. Avanzar el tiempo a b_{m+1} y volver al inicio.
16: Suavizar $C(b)$ con un filtro pasa bajos: $\sigma_{est}(b) = (C * B)(b)$

Como resultado se obtiene una representación a partir de la WT con una wavelet adaptada para cada tiempo en función de la señal de entrada. Nuevamente en este enfoque resulta claro un elevado esfuerzo computacional, siendo un algoritmo pensado para análisis puramente *offline*. Cabe destacar también que la grilla de frecuencias de análisis no es adaptada a la señal, por lo que nuevamente se produce un aumento de la *nitidez* de

la representación TF, pero bajo ciertas condiciones el error de estimación de la frecuencia instantánea puede no disminuir.

3.4.3.3. Recursividad

Otro camino se explora en [57] donde la operación de *synchrosqueezing* se realiza de forma recursiva para mejorar la resolución TF.

El algoritmo realiza V operaciones de *synchrosqueezing* sobre las sucesivas representaciones TF obtenidas. Adicionalmente, los autores muestran que la ganancia del algoritmo en resolución se obtiene de la función de remapeo de orden V, es decir $\omega_s^{[V]}(a,b)$, solamente. Esto permite calcular una única operación de *synchrosqueezing* si se dispone de las V funciones de remapeo. Es de suma importancia aclarar que a diferencia de la función de remapeo de mayor orden presentada en la Sección 3.4.1, en este caso $\omega_s^{[V]}(a,b)$ se calcula como $\omega_s^{[V]}(\omega_s^{[V-1]},b)$ y $\omega_s^{[1]}(a_m,b_n) = \omega_s(a_m,b_n)$ se obtiene como en la versión discreta de (3.6).

El método se presenta en el Algoritmo 3.

Algoritmo 3 Algoritmo recursivo presentado en [57]

Inicialización: Definir la wavelet y el número V de iteraciones a realizar.
Calcular la WT $W_s(a_m,b_n)$ y $w_s(a,b)$ según (3.6).
Calculo de la función de remapeo de orden V:
$\omega_s^{[1]}(a_m,b_n) = \omega_s(a_m,b_n)$
if $V > 1$ **then**
  for $v = 2$ **to** V **do**
    for $n = 1$ **to** N **do**
      for $m = 1$ **to** M **do**
        $\xi \leftarrow \omega_s^{[v-1]}(a_m,b_n)$;
        $\omega_s^{[v]}(a_m,b_n) \leftarrow \omega_s^{[v-1]}(\xi,b_n)$;
      end for
    end for
  end for
end if
Calculo de la SST de orden V:
$SST_s^{[V]}(a_m,\xi) \leftarrow 0$;
for $n = 1$ **to** N **do**
  for $m = 1$ **to** M **do**
    $\xi \leftarrow \omega_s^{[V]}(a_m,b_n)$;
    $SST_s^{[V]}(\xi,b_n) \leftarrow SST_s^{[V]}(\xi,b_n) + W_s(a_m,b_n)$;
  end for
end for
Salida: $SST_s^{[V]}(\xi,b_n)$

El efecto de este método es proporcionar sucesivas concentraciones de energía en las bandas determinadas, mediante el cálculo de V funciones de remapeo. Los sucesivos remapeos permiten lidiar con términos de modulación de mayor orden de la señal de entrada, mejorando la representación TF en situaciones de alto índice de modulación. Este

método resulta computacionalmente más eficiente que los presentados, ya que solamente requiere el cálculo de V funciones de remapeo que no se descartan. La concentración de energía lograda permite un análisis por crestas más sencillo, aunque nuevamente al no adaptar la grilla de análisis el error de estimación de las frecuencias instantáneas de las componentes de una señal puede no disminuir en ciertas condiciones.

3.5. Resumen

La introducción de la SST permitió obtener un incremento sustancial en la resolución obtenida por métodos tradicionales como la WT o la STFT. Muchos algoritmos han surgido para mejorar aún más las representaciones TF y es un tema de investigación activo y de gran interés. Los métodos más importantes introducidos en la Sección 3.4 logran un rendimiento muy bueno en términos de resolución TF. Estos enfoques en general conllevan a un alto esfuerzo computacional y difícilmente se pueden implementar en dispositivos embebidos o nodos de *edge computing*. Sin embargo, algunas aplicaciones del mundo real no requieren tanta *resolución* o *nitidez*, o incluso tampoco necesitan la reconstrucción de los modos. Éste es el caso cuando solo se rastrea un conjunto limitado de frecuencias instantáneas como en [58]. Además, una amplia gama de problemas en campos como la medicina [36], la geología [35] o la mecánica [37][38] requieren tratar con modulaciones de bajo orden o frecuencias que varían lentamente. En este trabajo, se aborda el problema de implementar el análisis TF para estos casos, en particular, para aquellos que requieren procesamiento en tiempo real. Se entiende por "tiempo real" a la capacidad de procesar datos a una velocidad promedio más rápida que la de llegada de la información, manteniendo así un flujo de datos constante a lo largo del tiempo. Del análisis de los métodos presentados se desprenden puntos importantes a considerar y que sirvieron como base para el desarrollo de un algoritmo adaptativo en tiempo real:

- La adaptación de la grilla de frecuencias de análisis puede mejorar tanto la resolución TF y la capacidad de discriminación de componentes individuales como el error de estimación de la frecuencia instantánea de dichas componentes.
- Considerar tanto la entropía de Rènyi como el MSE de la estimación de la frecuencia instantánea en la adaptación del algoritmo puede proporcionar una mayor utilidad de la representación TF obtenida.
- Un enfoque iterativo general puede resultar en una menor carga computacional que iteraciones específicas para cada tiempo o para el cálculo de parámetros específicos mediante cómputos auxiliares.

En el capítulo siguiente se presenta el mencionado algoritmo adaptativo orientado a la implementación en tiempo real, que mejora la resolución y disminuye el MSE de estimación de frecuencias en función de una adaptación de la grilla de análisis.

Capítulo 4

Trasformada *synchrosqueezing* adaptativa para análisis en tiempo real

RESUMEN: A continuación se presenta el desarrollo de un algoritmo adaptativo de análisis TF. El mismo busca obtener tanto un aumento de la resolución como una disminución del error de estimación de las frecuencias instantáneas sin un aumento significativo de la carga computacional. El desarrollo parte del objetivo de obtener una implementación en tiempo real. Las Secciones 4.1 y 4.2 introducen el problema y notación adicional a utilizar. El algoritmo propuesto es presentado en la Sección 4.3, mientras que su implementación es documentada en la Sección 4.4. Finalmente, la Sección 4.5 presenta resultados tanto numéricos como de rendimiento del algoritmo, adicionalmente se lo compara con otro toolbox disponible implementado a partir de un algoritmo presentado en el capítulo anterior.

4.1. Introducción

Por lo general, el análisis TF se realiza *offline* procesando todo el conjunto de datos a la vez como se introdujo en el capítulo anterior. Agregar capacidades de procesamiento en tiempo real permite nuevas aplicaciones, por ejemplo, advertencias críticas de salud, alarmas de mantenimiento preventivo y detección remota, entre otros. Estas aplicaciones del mundo real suelen implementarse en dispositivos embebidos o computadoras industriales con recursos computacionales limitados. Esto último implica que si se desea un aumento de *resolución* en el análisis TF, o una estimación instantánea de frecuencia más precisa, se debe considerar el costo computacional como una métrica a tener en cuenta al momento de seleccionar el algoritmo adecuado.

Adicionalmente, el método a utilizar en estas situaciones no debe requerir conocer toda la señal de antemano, sino que debe ser capaz de ofrecer una adaptación dinámica en función de la porción observable de la señal para un determinado tiempo. Estas restricciones plantean la necesidad de un algoritmo pensado desde el inicio para análisis en tiempo real. En el presente capítulo se presenta un método (y su implementación) que cumple con estas restricciones iniciales, logrando buenas métricas tanto de resolución como de rendimiento computacional. Adicionalmente, se provee su implementación en un toolbox Python de código abierto y fácil instalación [59], de forma de que sea accesible a toda la comunidad científica/técnica.

4.2. Planteo general del problema

Sea $s(t)$ la señal a analizar, T_s el tiempo de muestreo, y $s(n) = s(nT_s)$. Dado que se propone realizar procesamiento en tiempo real de la señal en cuestión, resulta de utilidad definir un segmento N de longitud B muestras de la señal $s(n)$ como

$$s_N(l) = s((N-1)B+l) \quad , \quad l \in [0, B-1],\ N \in \mathbb{Z},\ B \in \mathbb{N}.$$

En adelante se denominará a esta división (y procesamiento) por segmentos como *batching* (loteo) y a cada segmento como *batch* (lote). Luego, la señal original puede ser descripta como la suma de dichos segmentos o *batches*:

$$s(n) = \sum_{N=-\infty}^{\infty} s_N(n-(N-1)B).$$

Al realizar un análisis TF se produce un aumento de la dimensionalidad de la señal ya que se agrega una dimensión asociada a la frecuencia w, que en un análisis teórico es una variable continua. Cuando se quiere realizar un cómputo numérico de este análisis, naturalmente es necesario particionar w en un conjunto de K intervalos centrados en las frecuencias de análisis. Definimos este conjunto de K frecuencias de análisis como

$$\{w_0, \cdots, w_{K-1}\}.$$

Esta definición más general del conjunto de frecuencias de análisis se introduce debido a que, a diferencia de los métodos presentados en el Capítulo 3 donde la partición del eje de frecuencias tenía una distribución uniforme (o logarítmica), en este capítulo adaptaremos dicho conjunto a la señal de entrada y no necesariamente obtendremos particiones uniformes. El eje $[k]$ del gráfico TF de la Figura 4.1 muestra una grilla armada a partir del conjunto de frecuencias de análisis w_k.

Si tenemos una señal con una frecuencia ω, ésta será mapeada con cierto coeficiente en cada intervalo de frecuencia centrado en w_k. Obviamente, cuanto más cercano sea ω a w_k mayor será el coeficiente para ese intervalo. Esto inicialmente lleva a pensar que cuanto más pequeño sea el intervalo de frecuencia, o equivalentemente mayor sea el K, mejor será la resolución en frecuencia del análisis. Sin embargo, se deben considerar dos aspectos: por un lado, como se mencionará en la Sección 4.3 esto no es necesariamente cierto debido a los límites teóricos de las transformadas, y por otro lado aumentar K implica un aumento en el costo computacional. Inversamente, un K muy bajo puede implicar la incapacidad de distinguir diferentes frecuencias o estimar adecuadamente su valor.

Este trabajo plantea utilizar K como un parámetro de entrada del algoritmo, y un conjunto de frecuencias de análisis $\{w_0, ..., w_k, ..., w_{K-1}\}$ adaptado a la señal a analizar. La Figura 4.1 muestra un esquema conceptual del procesamiento de una señal con el algoritmo propuesto, incluyendo el proceso de *batching* y el análisis adaptativo TF.

Intuitivamente se puede observar que el error de estimación de frecuencia será menor con el nuevo conjunto de frecuencias de análisis como se mostrará en las secciones siguientes de este capítulo.

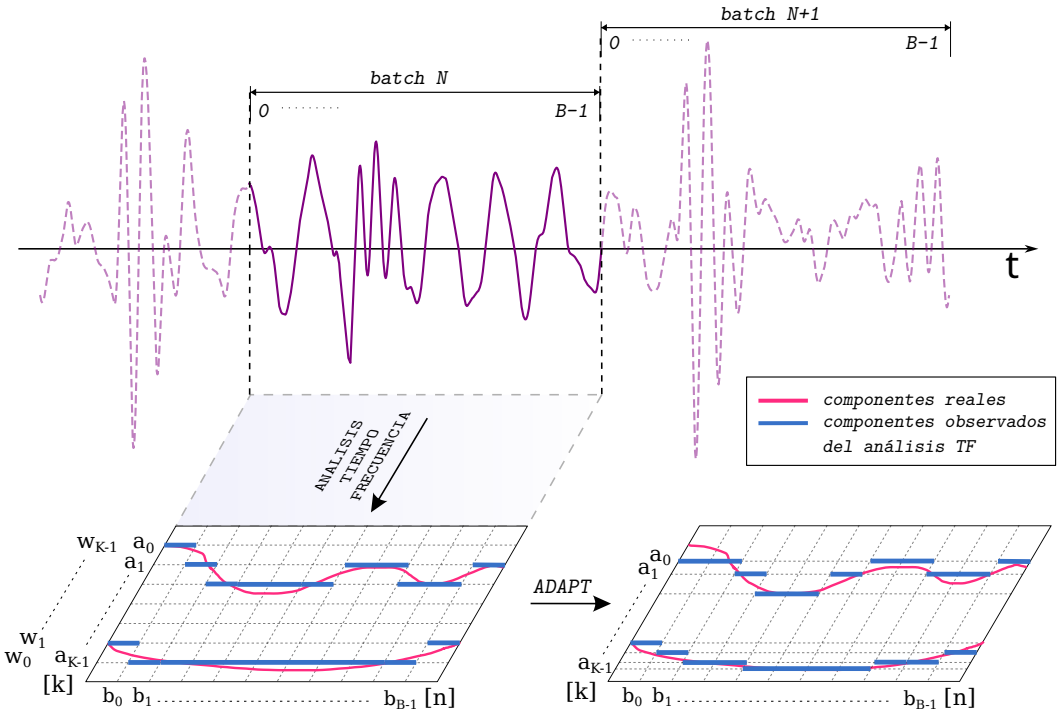


Figura 4.1: Esquema conceptual de procesamiento del algoritmo propuesto

4.3. El algoritmo ASST

4.3.1. Sobre la estimación de frecuencia instantánea

Un estudio sobre la relación entre la resolución en frecuencia, el error de reconstrucción y la discretización de las escalas (o en forma equivalente, frecuencias) puede ser encontrado en [53]. Particularmente se muestra que dos componentes con frecuencias ω y $\omega + \Delta\omega$ pueden ser recuperadas independientemente de la CWT con un error máximo de reconstrucción de ε_r si

$$\Delta\omega(\omega) \geqslant \Delta\omega_{min}(\omega) = \omega\left(\frac{\xi_2(\varepsilon_r)}{\xi_1(\varepsilon_r)} - 1\right) \tag{4.1}$$

donde $\xi_1(\varepsilon_r), \xi_2(\varepsilon_r)$ definen el soporte-ε_r de la wavelet y satisfacen

$$\left|\frac{\int_0^{\xi_1} \hat{\psi}^*(\xi)\frac{d\xi}{\xi}}{\int_0^{\infty} \hat{\psi}^*(\xi)\frac{d\xi}{\xi}}\right| = \varepsilon_r/2, \qquad \left|\frac{\int_{\xi_2}^{\infty} \hat{\psi}^*(\xi)\frac{d\xi}{\xi}}{\int_0^{\infty} \hat{\psi}^*(\xi)\frac{d\xi}{\xi}}\right| = \varepsilon_r/2.$$

Cabe destacar que el intervalo de frecuencia $[\xi_1(\varepsilon_r), \xi_2(\varepsilon_r)]$ contiene la proporción $(1-\varepsilon_r)$ de la integral total de la función wavelet. Claramente $\xi_1(\varepsilon_r)$ y $\xi_2(\varepsilon_r)$ dependen de la wavelet seleccionada, pero también varían con el error admitido ε_r. Éste es el límite teórico impuesto por la CWT. Esta definición de soporte extiende la definición utilizada en [54] y presentada en la Sección 3.4.3.2, donde se considera que $\widehat{\psi}(\xi) \approx 0$ para $|\xi - \omega_0^{\psi}| \geqslant \beta$.

Reducir el error de reconstrucción requiere de frecuencias componentes (o modos) más separadas. Por otro lado, cuando se analizan frecuencias con una separación peque-

ña[1] $\Delta\omega_{min}$, es de esperarse un error de reconstrucción ε_r mayor. Nuevamente, estas consideraciones complejizan la condición de separabilidad planteada en la Sección 3.4.3.2.

Cuando se trabaja con un conjunto discreto de frecuencias de análisis, dos tonos separados por $\Delta\omega$ pueden ser identificados como dos componentes distintas si ambos caen en diferentes segmentos de la partición del eje w. Como fue mencionado en otras oportunidades, el enfoque tradicional particiona el eje de frecuencias con una grilla uniforme o logarítmica. Los autores de [53] muestran que para evitar el incremento del error de reconstrucción debido a la mencionada discretización, dado ε_r uno debe elegir la cantidad de frecuencias de análisis K de forma que

$$K = \left\lceil \frac{10\log 2}{\log \xi_2(\varepsilon_r) - \log \xi_1(\varepsilon_r)} \right\rceil . \tag{4.2}$$

La Tabla 4.1 ejemplifica la variación del número requerido de wavelets para discriminar dos componentes con frecuencias ω y $\omega + \Delta\omega_{min}$ para diferentes ε_r. Si el valor de K seleccionado es menor que el dado por (4.2), dos componentes que satisfacen (4.1) caerán en el mismo segmento de frecuencia, por lo tanto no se los podrá diferenciar a partir de su representación TF. Por otro lado, si K es mayor, no se obtiene ganancia alguna dado que la resolución en frecuencia está limitada por la resolución teórica de la CWT (y de la wavelet seleccionada). Notar que para una wavelet dada, el número de frecuencias de análisis requerido para poder posicionar dos componentes en segmentos de frecuencia separados se incrementa cuando la separación entre dichos componentes ($\Delta\omega_{min}$) se reduce. Para un $\Delta\omega_{min}$ pequeño, el número K de frecuencias de análisis requerido puede implicar que se excedan los recursos computacionales disponibles para un procesamiento en tiempo real o de baja latencia. Por este motivo en este trabajo se considera un método distinto de discretización del eje de frecuencias.

Tabla 4.1: K y $\Delta\omega_{min}(\omega)$ para diferentes valores de ε_r, utilizando $T_s = 1/2000s$, $\omega = 100Hz$, y una Wavelet madre de Morlet con $f_c = 256Hz$ y un ancho de banda de $40Hz$ a $3dB$.

ε_r	$\Delta\omega_{min}$	K
0.05	155.56	8
0.1	120.00	9
0.2	78.26	12
0.3	60.00	15
0.4	46.15	19
0.5	37.04	23

Como fue mencionado, cuando se usa una grilla de frecuencias discretas para el análisis, la frecuencia instantánea de una componente de la señal $s(t)$ es mapeada en un segmento de frecuencia para cada tiempo discreto b_n. Sea w_k la frecuencia de análisis correspondiente a ese bin para un tiempo fijo. Por ejemplo, w_k puede haber sido seleccionada como frecuencia instantánea por un análisis de crestas. Dado que ω es a priori desconocida, para obtener el error de estimación de frecuencia, se la puede modelar como una variable aleatoria uniforme dentro del bin correspondiente. Es decir,

[1] En este caso nos referimos a *pequeña* en términos relativos a las definiciones de esta sección y los requerimientos de las señales admisibles.

$\omega \sim U\left(w_k - \frac{(\Delta w)_k}{2}, w_k + \frac{(\Delta w)_{k+1}}{2}\right)$, donde $(\Delta w)_k$ sigue la definición en (3.8). Usando este modelo, obtenemos el error cuadrático medio de estimación (MSE) para ese bin como

$$MSE_{w_k} = E[(\omega - w_k)^2] = \frac{1}{12}\left(\frac{(\Delta w)_k + (\Delta w)_{k+1}}{2}\right)^2. \tag{4.3}$$

aún utilizando una SST. Es decir, al estimar por un análisis de crestas sobre el plano TF la frecuencia instantánea, la concentración de energía introducida por el proceso de *Synchrosqueezing* no aporta mayor precisión debido a la discretización del eje de frecuencias.

Para reducir el error introducido por la discretización de frecuencias, es necesario disminuir $(\Delta w)_k$ en concordancia con la discusión planteada. El esquema tradicional incrementaría el número total de wavelets. Por ejemplo, para un entero dado J, el número de frecuencias podría aumentarse de K a JK, con el objetivo de disminuir $(\Delta w)_k$ en un factor de J. Sin embargo, esto también incrementaría el número requerido de operaciones de cómputo en J^3 dado que (3.8) involucra operaciones en los ejes de escala, frecuencia y tiempo. Alternativamente, podemos dividir el segmento de frecuencia $[w_k - \frac{(\Delta w)_k}{2}, w_k + \frac{(\Delta w)_{k+1}}{2})$ en J sub-intervalos. Haciendo esto, la grilla de discretización de frecuencias se vuelve localmente más fina y la representación TF más detallada. Con J frecuencias ya asignadas, el algoritmo propuesto distribuye las $K - J$ frecuencias de análisis restantes en regiones fuera del segmento en cuestión (centrado en w_k) de manera de mantener el número total de wavelets constante.

4.3.2. Discretización no uniforme de la grilla de frecuencias de análisis

Dado el conjunto de frecuencias de análisis $\{w_0, ..., w_k, ..., w_{K-1}\}$, el objetivo es encontrar un nuevo conjunto $\{w_0^*, ..., w_k^*, ..., w_{K-1}^*\}$ tal que, para un tiempo fijo b_n y una frecuencia ω,

$$min\|w_k^* - \omega\| \leqslant min\|w_k - \omega\| \tag{4.4}$$

Para condiciones donde $\varepsilon_r \approx 0$, el criterio anterior equivale a minimizar la entropía de Rènyi, ya que implica aumentar la concentración de energía en una/unas frecuencia/s particular/es.

La solución trivial sería seleccionar un conjunto $\{w_0^*, ..., w_k^*, ..., w_{K-1}^*\}$ tal que $w_k^* = \omega$, pero dado que ω es desconocida, resulta necesario establecer una política para resolver (4.4) cerca de las bandas de frecuencia de interés. Para hacer esto, se requiere seleccionar los segmentos de frecuencia (centrados en w_k) que pueden contener la frecuencia instantánea actual y refinarlos. Es sabido que el espectro de la señal es un buen candidato para determinar la densidad de frecuencias de análisis a lo largo del eje dado que se concentra cerca de ω. Luego, analizamos la energía contenida en los coeficientes de la WT en el *batch N*. Usando (3.4) y sumando sobre el N-ésimo *batch*, obtenemos la energía asociada a cada coeficiente como [60]:

$$E_N(a_m) = \sum_{n=B(N-1)}^{BN-1} \frac{|W_s(a_m, b_n)|^2}{a_m}. \tag{4.5}$$

Dado que $E_N = \sum_{\forall m} E_N(a_m)$ es la energía total del *batch*, la distribución normalizada de energía puede ser estimada como:

$$\hat{E}_N(a_m) = \frac{E_N(a_m)}{E_N} \tag{4.6}$$

Una estimación similar se puede obtener utilizando la SST en lugar de la WT, es decir, reemplazando $W_s(a_m, b_n)$ por $S_s(w_k, b_n)$ en (4.5),

$$E_N(w_k) = \sum_{n=B(N-1)}^{BN-1} |S_s(w_k, b_n)|^2 (\Delta\omega)_k. \tag{4.7}$$

Dado que $\sum_{\forall k} E_N(w_k) = E_N$, obtenemos como en (4.6)

$$\hat{E}_N(w_k) = \frac{E_N(w_k)}{E_N}. \tag{4.8}$$

En adelante se hará referencia indistintamente a la estimación de energía como $\hat{E}_N(k)$, diferenciando entre (4.6) y (4.8) cuando sea necesario. Ahora el problema se reduce a seleccionar K frecuencias de análisis de manera que los bins de frecuencia que contienen la frecuencia instantánea actual tengan una grilla más fina que el resto. A partir de una grilla inicial de frecuencias y una estimación de la energía de la señal $\hat{E}_N(k)$, necesitamos realizar un procedimiento de dos pasos. Primero, los segmentos de frecuencia relevantes son seleccionados a partir del perfil de energía. Luego, las K frecuencias de análisis son distribuidas de manera que los segmentos más relevantes tengan una grilla más fina a expensas de los bins menos relevantes. Para lograr esto, proponemos dos métodos distintos: distribución *proporcional* y *con umbral*.

4.3.2.1. Distribución Proporcional

Dado $\hat{E}_N(k)$ asociado al bin de frecuencia $\left[w_k - \frac{(\Delta w)_k}{2};\ w_k + \frac{(\Delta w)_{k+1}}{2}\right)$, asignamos a este bin un número de frecuencias proporcional a $\hat{E}_N(k)$ manteniendo el número total de frecuencias de análisis igual a K. Éste es un problema de distribución entera donde se tienen K categorías y la relevancia o peso de cada categoría es $\hat{E}_N(k)$. Es decir, se deben distribuir un número entero de elementos (frecuencias en nuestro caso) sobre un conjunto finito de categorías (bandas de frecuencia) de acuerdo a una proporción no entera para cada categoría (aquí, distribución de energía). Una solución bien establecida a este problema es la cuota de Hagenbach-Bischoff [61].

Cuando la redistribución inicial de las frecuencias de análisis es completada, el proceso se repite para evaluar la concentración de energía dentro de las nuevas bandas de frecuencia. Este proceso iterativo se repite hasta que la energía normalizada sobre las bandas seleccionadas es uniforme o se alcanza el número máximo de iteraciones. Claramente, si se alcanza la uniformidad en la distribución de energía se obtiene el peor caso para la entropía de Rènyi. Sin embargo, se obtiene un MSE muy bajo en la estimación de frecuencia instantánea, cómo se mostrará en la Sección 4.5 para el caso de un tono puro, dónde el algoritmo adapta la grilla de frecuencias en un entorno muy cercano a la frecuencia de la señal. La posibilidad de controlar la cantidad de iteraciones paramétricamente permite al usuario configurar al algoritmo en función de la aplicación particular a la que se lo aplica.

Este método se resume en el Algoritmo 4.

Algoritmo 4 Método de reasignación de frecuencias *proporcional*

1: $K, \text{Iters}_{max} \leftarrow$ parámetros
2: Generar el conjunto inicial de frecuencias de análisis $\mathbf{w} = \{w_0, w_1, ..., w_{K-1}\}$
3: Computar $W_s(a_m, b_n)$ [o $S_s(w_k, b_n)$] sobre $\mathbf{w}$
4: Computar la energía $E_N(k)$ de acuerdo a (4.5) [o (4.7)]
5: Normalizar $E_N(k)$ como en (4.6) [o (4.8)]
6: Verificar si se alcanzó el número máximo de iteraciones (Iters_{max}) o se encontró equilibrio. **If** $False \mapsto continue$ **else** $goto \mapsto 10$.
7: Computar el número de frecuencias por banda con K y $\hat{E}_N(k)$ como parámetros
8: Generar el nuevo conjunto de frecuencias $\mathbf{w}$
9: goto $\mapsto 3$
10: **fin**

La Figura 4.2 muestra un ejemplo numérico de este proceso de relocación de frecuencias con $s(n) = sin(2\pi 20t) + 0{,}5\,sin(2\pi 10\pi t) + 0{,}2\,sin(2\pi 40t)$, $K = 12$ frecuencias inicialmente distribuidas uniformemente entre 10 y 50 Hz, y una frecuencia de muestreo de 200 Hz. En este caso particular, luego de la iteración 0 el nuevo conjunto $\mathbf{w}^* = \{w_0^*, w_1^*, ..., w_k^*, ..., w_{K-1}^*\}$ producirá $\Delta w_3^* = \frac{\Delta w_3}{4}$ para $w_3 = 19{,}23Hz$ y $\Delta w_7^* = \frac{\Delta w_7}{3}$ para $w_7 = 31{,}53Hz$ según la distribución calculada que se muestra en la Figura 4.2a. Se puede observar también en la Figura 4.2c que después de la iteración 1, las frecuencias se encuentran más agrupadas cerca de las componentes de la señal. Las figuras 4.2b y 4.2d muestran la redistribución de las frecuencias para la segunda iteración.

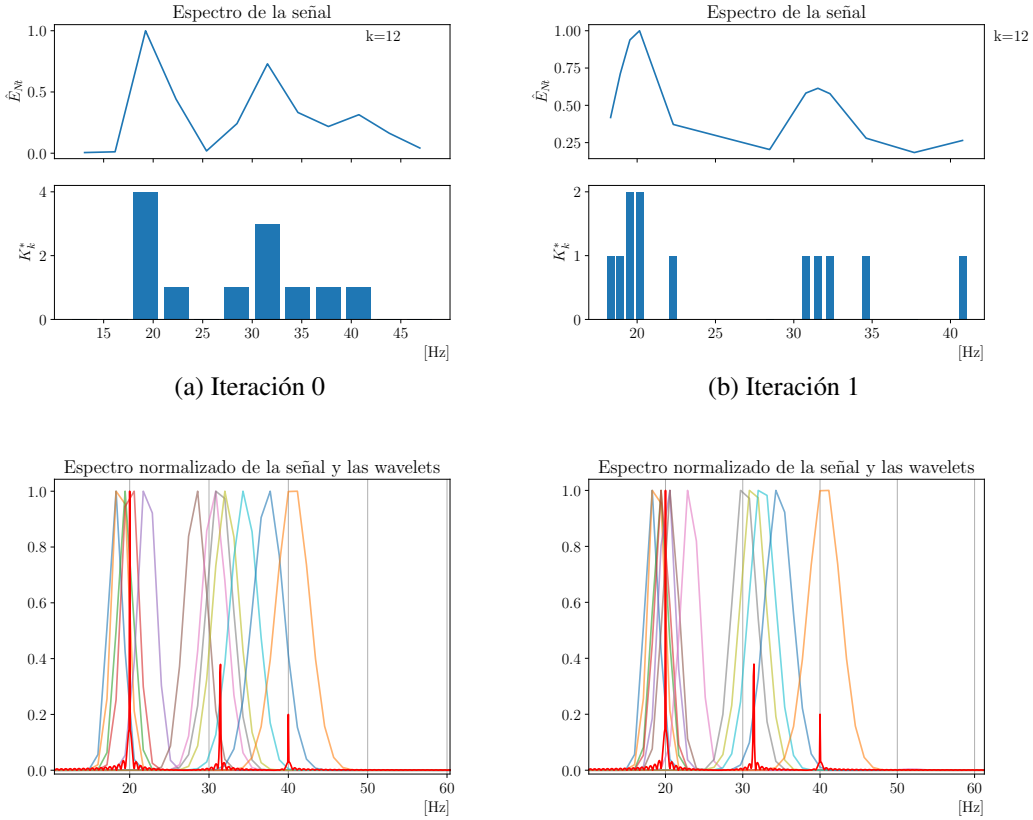
4.3.2.2. Distribución por *umbral*

En este caso, se utiliza un parámetro de umbral E_{th} para determinar la relevancia de un segmento dado. Si la energía normalizada en dicho segmento no excede el umbral definido, es decir, $\hat{E}_N(k) < E_{th}$, la frecuencia de análisis correspondiente w_k es marcada para ser reasignada a una banda más relevante. Las frecuencias marcadas son distribuidas entre las bandas restantes a través de un método de distribución entera como en el caso anterior, pero analizando solamente la energía a través de las bandas sobrevivientes. Cómo en el método de reasignación proporcional, estas operaciones se repiten sobre las nuevas bandas de análisis hasta que no queden más frecuencias por reasignar o se alcance el número máximo de iteraciones. Para seleccionar el valor de E_{th}, el usuario debe considerar la SNR dado que E_{th} está destinado a separar el ruido de la señal.

Los pasos mencionados se listan en el Algoritmo 5. La Figura 4.3 muestra el número de frecuencias asignadas por bin, basado en el mismo ejemplo de señal y los mismos parámetros del método *proporcional*. En este caso, el método de *umbral* finaliza la distribución de frecuencias luego de solamente dos iteraciones con el umbral seleccionado.

4.3.3. Sobre la amplitud de la señal reconstruida

Sea $\hat{s}(t)$ la señal reconstruida a partir de $S_s(w_k, b_n)$. Luego, la señal reconstruida obtenida a partir de $S_s(w_k, b_n)$ es



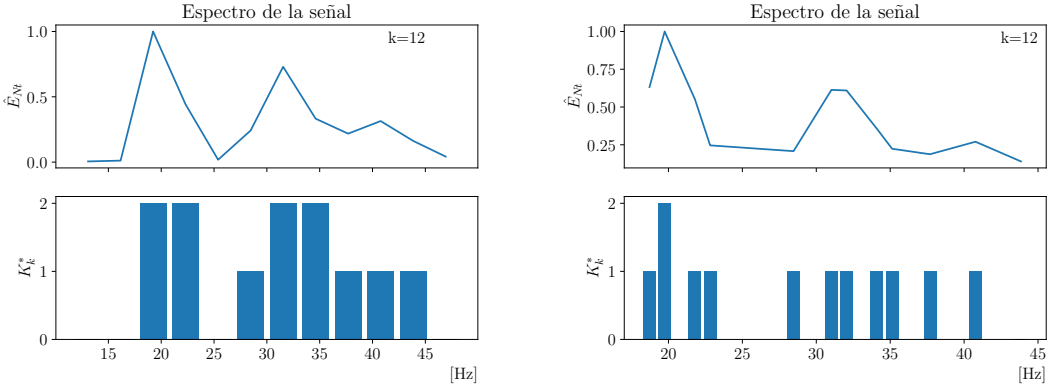
(a) Iteración 0

(b) Iteración 1

(c) Luego de iteración 1

(d) Luego de iteración 2

Figura 4.2: Reasignación de frecuencias *Proporcional*. Fila de arriba: reasignación proporcional por banda. Fila de abajo: espectro de la señal y de las Wavelets.



(a) Iteración 0

(b) Iteración 1

Figura 4.3: Redistribución por el método de *umbral*.

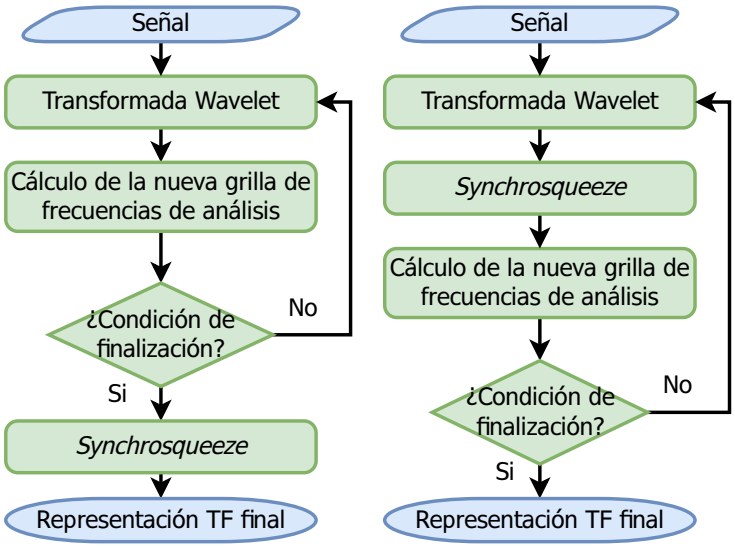
Algoritmo 5 Reasignación de frecuencias por el método de *umbral*

1: $K, \text{Iters}_{max}, E_{th} \leftarrow$ parámetros
2: Generar el conjunto inicial de frecuencias de análisis $\mathbf{w} = \{w_0, w_1, ..., w_{K-1}\}$
3: Computar $W_s(a_m, b_n)$ [o $S_s(w_k, b_n)$] sobre $\mathbf{w}$
4: Calcular la energía $E_N(k)$ de acuerdo a (4.5) [o (4.7)]
5: Normalizar $E_N(k)$ como en (4.6) [o (4.8)]
6: Verificar si el número máximo de iteraciones (Iters_{max}) fue alcanzado o se encontró equilibrio. **If** $False \mapsto continue$ **else** $goto \mapsto 11$.
7: Marcar las bandas donde $E_N(k) < E_{th}$ y contarlas ($\#tag$)
8: Calcular el número de frecuencias por banda con el número de frecuencias a asignar ($\#tag$), el número de bandas para asignarlas ($K - \#tag$), y la relación de proporción $\hat{E}_N(k)$ como parámetros
9: Generar el nuevo conjunto de frecuencias $\mathbf{w}$
10: goto $\mapsto$ 3
11: **finish**

$$\hat{s}(b_n) \approx \Re\mathfrak{e}\left[\frac{1}{C_\psi}\sum_k S_s(w_k, b_n)(\Delta w)_k\right], \tag{4.9}$$

$$C_\psi = \frac{1}{2}\int_0^\infty \overline{\hat{\psi}(\xi)}\frac{d\xi}{\xi}.$$

Dado que $\{\psi(\frac{l-b_n}{a_m})\}$ genera un *frame* en lugar de una base, como es el caso de las wavelets típicas (e.g. Morlet)[62], las wavelets vecinas contribuyen en cierta medida al mismo coeficiente de la WT afectando la reconstrucción, especialmente cuando las componentes de la señal se encuentran cerca de los límites de análisis. Dado el número finito de frecuencias de análisis, la suma sobre el conjunto $A_{\tilde{\varepsilon},s}(b_n)$ se vuelve pequeña cuando w_k se acerca a dichos límites. Como resultado, la energía contenida en $S_s(w_k, b_n)$ decrece a medida que w_k se acerca a los límites. Claramente, existe un error de reconstrucción debido a la computación sobre una grilla finita, el cual es especialmente notable cuando la banda de análisis se estrecha cerca de las componentes individuales. Dado que el algoritmo propuesto ajusta los límites de análisis a través de iteraciones sucesivas, si el proceso de synchrosqueezing se incluye dentro del ciclo (por ejemplo, cuando se utiliza (4.8) para estimar la energía), dichos límites se volverán extremadamente cercanos a algunas componentes de la señal produciendo una estimación de frecuencia instantánea muy precisa a expensas de la reconstrucción de la amplitud. Cuando la amplitud instantánea de la señal reconstruida es importante, la operación de synchrosqueezing puede realizarse fuera del ciclo. En adelante, se denominará implementación fuera del ciclo o *Off-The-Loop* (OTL) cuando la operación de synchrosqueezing se realiza después de la refinación de la grilla, e implementación dentro del ciclo o *In-The-Loop* (ITL) cuando la operación de synchrosqueezing se realiza dentro del ciclo de refinación para computar la estimación de energía. La Figura 4.4 muestra ambas variantes del algoritmo.



(a) OTL (b) ITL

Figura 4.4: Diagramas de flujo de las variantes del algoritmo.

4.4. Implementación del *toolbox* `adaptivesswt`

Tal como fue mencionado en la Sección 4.1, la implementación del algoritmo propuesto fue llevada a cabo con las premisas de aplicación a tiempo-real, extensibilidad, y fácil distribución, de forma que pueda ser aplicado a una amplia variedad de situaciones. En ese sentido, el lenguaje de programación seleccionado para el desarrollo fue Python, ya que:

- Es un lenguaje de programación de alto nivel
  Esto permite que el usuario programador se concentre en la implementación del algoritmo, sin necesidad de atender a problemas complejos de desarrollo como la alocación y manejo de memoria, ciclo de vida de los objetos, seguridad, etc. Su sintaxis posee un alto nivel de expresividad por lo que requiere pocas líneas de código en comparación con otros lenguajes para resolver problemas similares.

- Dispone de un gran ecosistema de bibliotecas de código abierto
  El índice de paquetes de Python *Pypi* [63] lista en enero de 2024 un total de 510229 proyectos. Esto permite que el usuario pueda utilizar la infraestructura disponible para resolver problemas complejos de manera sencilla.

- Es un lenguaje de programación multiparadigma y multiplataforma
  La portabilidad de la implementación resulta de suma importancia para la aplicación del algoritmo en diferentes situaciones y por ende, plataformas. Actualmente se lo utiliza para el desarrollo en plataformas que van desde dispositivos extremadamente sencillos como Raspberry Pi [64], pasando por SoCs como Pynq [65], hasta computación de alto desempeño [66][67][68].

- Tiene una interfaz eficiente con lenguajes de más bajo nivel como C
  Python posee una integración con C "nativa" a través de la biblioteca estándar `ctypes`, la cual provee una interfaz de programación de aplicaciones o *Application*

Programming Interface (API) bien definida[69], e incluso a partir de la versión 3.2 parte de la API posee compatibilidad binaria (interfaz binaria de aplicaciones o *Application Binary Interface* (ABI)) lo que permite que no sea necesario re-compilar código que utilice dicha ABI cuando se produce un cambio de versión de Python. A través de estos mecanismos, la mayoría de las bibliotecas maduras están optimizadas en bajo nivel logrando tiempos de ejecución similares a los de los lenguajes de bajo nivel, incluso utilizando extensiones particulares de los procesadores (por ejemplo, *Single Instruction Multiple Data* (SIMD)). Adicionalmente, permite al usuario hacer uso de la ley de Amdhal para hacer optimizaciones de bajo nivel en puntos críticos de los algoritmos.

- Es el lenguaje más popular[70][71]
  La enorme popularidad de Python, produce por un lado que exista una gran comunidad trabajando en la actualización y optimización del ecosistema, y por otro que exista una gran cantidad de potenciales usuarios. Adicionalmente, es el lenguaje seleccionado como estándar de facto para el desarrollo y ejecución de algoritmos de aprendizaje automático e inteligencia artificial.

- Fácil empaquetamiento, distribución e instalación de paquetes.
  Python ofrece una infraestructura avanzada para la distribución y empaquetado de módulos de software. La misma permite la instalación automática de dependencias, instalación desde repositorios git directamente, creación de comandos y ejecutables integrados con la terminal de forma automática, entre otras facilidades. Esto permite la sencilla distribución del *toolbox* desarrollado y su aplicación directa.

4.4.1. Arquitectura general

Una perspectiva estructural de la arquitectura general de la implementación del algoritmo se muestra en la Figura 4.5. El paquete de Python se encuentra disponible públicamente en [59].

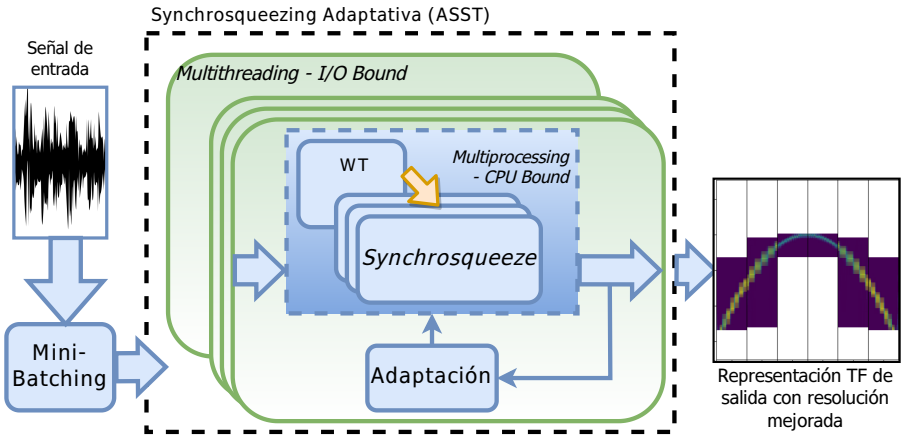


Figura 4.5: Esquema estructural de la arquitectura de software

4.4.1.1. Batching

Dado que la implementación está orientada a aplicaciones de tiempo real o *streaming*, la tasa de datos de entrada está determinada típicamente por la frecuencia del conversor analógico digital o *Analog-to-Digital Converter* (ADC), la velocidad de lectura del disco o dispositivo de almacenamiento, o incluso por la tasa de transferencia de red. Estas fuentes están asociadas a procesos atados a entrada/salida (*I/O Bound*), por lo que para mejorar el rendimiento, el primer paso es dividir la señal de entrada en *batches* que se encolan en diferentes hilos (asociados al mismo procesador). Cada hilo procesa un *batch* $s_q(n)$. Para formar el *batch*, se recolectan B muestras de la señal y se agrega un intervalo de guarda de longitud P muestras, como se explica más adelante. Para evitar los errores introducidos por el cono de influencia de la WT, en este trabajo se proponen dos enfoques de *streaming*: *overlap and add* (OA) asociado con el relleno de ceros y *overlap and save* (OS) asociado a una ventana deslizante.

Para OA, el intervalo de guarda se rellena con ceros de manera que el *batch* $s_q(n)$ se define como

$$s(n) = ..., s(0), s(1), ..., s(B), s(B+1), ..., s(2B), ...$$

$$s_0(n) = \underbrace{s(0), s(1), ..., s(B), \overbrace{0, ..., 0}^{\text{P samples}}}_{\text{batch 0}}$$

$$s_1(n) = s(B+1) + \hat{s}(B+1), ..., s(B+P) + \hat{s}(B+P), s(B+P+1), ..., s(2B), \overbrace{0, ..., 0}^{\text{P samples}}$$

donde $\hat{s}(n)$ es la señal reconstruida según (4.9).

Para OS, la señal se divide de acuerdo a:

$$s(n) = ..., \underbrace{\overbrace{s(0), ..., s(\lfloor P/2 \rfloor)}^{\text{No guardado}}, s(\lfloor P/2 \rfloor + 1), ..., s(\lfloor P/2 \rfloor + B), \quad \overbrace{s(\lfloor P/2 \rfloor + B + 1), ..., s(B+P)}^{\text{no guardado}}}_{\text{batch } 0 := s_0(n)},$$

$$\underbrace{\overbrace{s(B+1), ..., s(B+\lfloor P/2 \rfloor)}^{\text{No guardado}}, s(B+\lfloor P/2 \rfloor + 1), ..., s(2B+\lfloor P/2 \rfloor), \quad \overbrace{s(2B+\lfloor P/2 \rfloor + 1), ..., s(2B+P)}^{\text{no guardado}}}_{\text{batch } 1 := s_1(n)}, \cdots$$

Vale la pena destacar que el método OA requiere procesar el *batch* anterior mientras que OS no.

Estos enfoques por lotes optimizan la utilización de la memoria ya que grandes cantidades de datos se dividen automáticamente en *batches* evitando la saturación de la memoria del sistema de procesamiento, debido a que no se requiere la carga simultánea de todos los datos. Esta metodología de memoria controlada también proporciona beneficios en cuanto a velocidad de computo cuando se utilizan cachés (CPUs) o FIFOs (FPGAs), ya que se explota el principio de localidad.

4.4.1.2. Transformada Wavelet

Este sub-módulo computa $W_{s_q}(a_k, b_n)$, análogamente a (3.4) para un *batch* particular $s_q(n)$ de la señal de entrada. Luego de dividir dicha señal, el primer paso para cada *thread*

es realizar la WT sobre cada *batch*. La WT es un algoritmo bien establecido y existen implementaciones muy eficientes. En este trabajo se utilizó PyWavelets[72], pero puede ser reemplazado por cualquier otro módulo que calcule la WT de manera eficiente. El único requisito para el módulo de cálculo es tener la capacidad de realizar la WT en escalas (frecuencias) definidas por el usuario (no uniformes).

4.4.1.3. *Synchrosqueeze*

Este sub-módulo computa la reasignación de frecuencias de acuerdo a (3.8) (y el remapeo de acuerdo a (3.9)). Adicionalmente, a la SST, también implementa las diferentes estrategias presentadas en la Sección 3.4.2 con el objeto de ampliar la usabilidad del *toolbox* como fue mencionado. Dichas estrategias son:

- SET
  Para la implementación, (3.19) también se puede expresar algorítmicamente de la siguiente manera:

  $$SET_s(\omega, b) = \begin{cases} W_s(a,b), \text{ sii } \omega = \omega_0 \\ 0, \ \omega \neq \omega_0 \end{cases}$$

  obviamente, para el caso discreto la igualdad se reemplaza por la pertenencia a un segmento de frecuencias en torno a cada w_k.

- TSST
  Este método de reasignación realiza la operación de *Synchrosqueeze* sobre el eje de tiempos en lugar del eje de frecuencias cómo se menciona en 3.4.2.1. El remapeo se implementó según (3.16) y la transformación de acuerdo a (3.17).

- RM
  La implementación está basada en (3.18), dónde el proceso de *squeeze* se realiza tanto en tiempo como en frecuencia.

Las estrategias mencionadas son prácticamente idénticas en términos de arquitectura computacional y complejidad. Cómo ya fue mencionado, la única diferencia es la falta de una operación de acumulación dentro de la implementación de SET. Debido a esto, se hará referencia a este módulo como SST de forma general, sin por ello perder detalles de implementación.

Esta etapa es la más exigente computacionalmente, e incluye una diferenciación matricial para calcular $w_s(a_k, b_n)$ (o $t_s(a_k, b_n)$ para TSST). Esta matriz de reasignación $w_s(a_k, b_n)$ se calcula utilizando operaciones vectorizadas de NumPy[73] para todos los casos.

4.4.1.4. Implementaciones del paralelismo

En (3.8) se puede observar que cada elemento de la matriz $S_s(w_k, b_n)$ se puede calcular de forma independiente, involucrando únicamente los valores de $W_s(a_m, b_n)$ y $\omega_s(a_m, b_n)$ correspondientes al tiempo b_n dado. Es posible entonces reescribir (3.8) de forma algorítmica como

$$S_s(w_k, b_n) = \frac{1}{(\Delta w)_k} \sum_m \hat{W}_s(a_m, b_n) \tag{4.10}$$

siendo

$$\hat{W}_s(a_m,b_n) = \begin{cases} W_s(a_m,b_n)a_m^{-3/2}(\Delta a)_m & \text{si } \omega_s(a_m,b_n) \in [w_k, w_{k+1}) \\ 0 & \text{en otro caso} \end{cases}. \tag{4.11}$$

Esta reinterpretación del algoritmo permite distribuir la carga de trabajo en diferentes unidades de procesamiento, ya sea separando el cálculo de cada elemento de la matriz resultante o separando por sub-matrices, según las capacidades de la plataforma a utilizar. Si se divide el cómputo en Γ procesos, el cálculo de $S_{s_q}(w_k,b_n)$ para cada proceso tendrá un tamaño máximo de datos de $K \times \frac{B+P}{\Gamma}$. Nótese que esta implementación reduce la carga computacional porque el tamaño de entrada se reduce en Γ. Si t_{ex} es el tiempo de ejecución de un proceso con complejidad $\mathcal{O}((B+P)^3)$, el método propuesto obtiene $t_{ex}(\Gamma) = \frac{(B+P)^3}{\Gamma^3} = \frac{t_{ex}}{\Gamma^3}$.

Dado que el *toolbox* fue diseñado para ofrecer rendimiento tanto en plataformas sencillas como en plataformas de alto desempeño, se utilizaron diferentes *backends* para la implementación del paralelismo.

Python puro

En el caso que la plataforma sólo presente compatibilidad con Python, se realizó la implementación del paralelismo utilizando la biblioteca `multiprocessing`. La arquitectura de la implementación se muestra en la Figura 4.6.

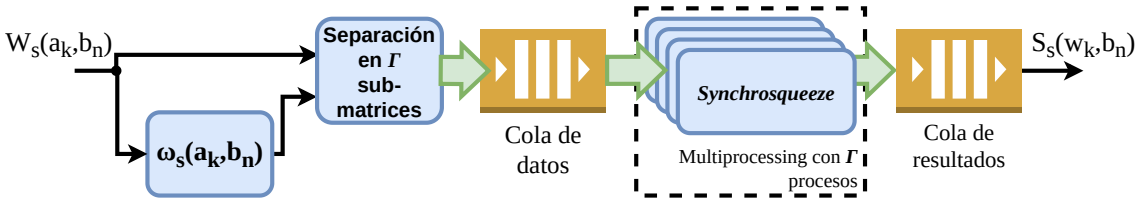


Figura 4.6: Estructura de la implementación de la operación de *Synchrosqueeze* en un esquema multiprocesamiento.

Esta implementación tiene pocos requerimientos en cuanto al stack de software, sin embargo presenta la desventaja que en Python para distribuir el trabajo en diferentes unidades de procesamiento es necesario instanciar un interprete por unidad debido al *Global Interpreter Lock* (GIL), lo que introduce un overhead considerable en la comunicación de datos inter-proceso. Por otro lado, si la distribución de la carga de trabajo se realiza mediante hilos (y no procesos), dichos hilos en Python (puro) no pueden ejecutarse en diferentes núcleos de procesamiento. El despacho de los datos y la obtención del resultado final se realiza mediante colas que permiten la comunicación entre los diferentes procesos y la sincronización de los mismos.

Pre-compilación en código máquina

Como segundo backend, para plataformas con capacidad de paralelismo con una arquitectura homogénea[2] se utilizó la biblioteca *Numba*. Esta biblioteca permite la compilación automática de código python en código máquina descartando las limitaciones del

[2]Es decir, todas las unidades de cómputo tienen la misma arquitectura.

interprete de Python. En este caso, la arquitectura no necesita colas para la transferencia de datos, ya que el paralelismo se realiza mediante el desplegado de los ciclos `for` sobre los diferentes núcleos. La arquitectura resulta mucho más sencilla según se observa en la Figura 4.7, obteniéndose un mejor rendimiento.

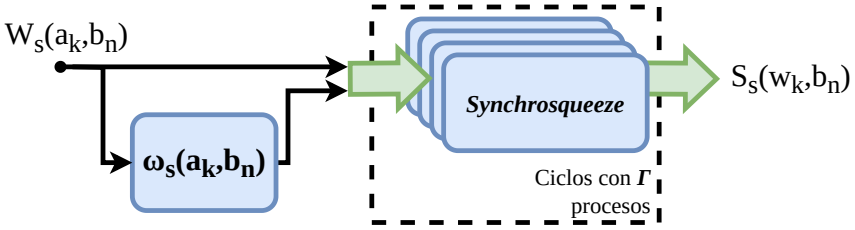


Figura 4.7: Estructura de la implementación utilizando Numba.

El *snippet* de código a continuación muestra a modo ilustrativo la implementación de la función:

```
@njit(parallel=True, fastmath=True)
def _freq_agregate_nb(deltaFreqs: np.ndarray, borderFreqs: np.ndarray,
                      aScale: np.ndarray, wab: np.ndarray, tr_matr: np.ndarray,
                      sst: np.ndarray):
    for b in prange(sst.shape[1]):        # Time
        for w in prange(sst.shape[0]):    # Frequency
            components = np.logical_and(wab[:,b] > borderFreqs[w],
                                        wab[:,b] <= borderFreqs[w+1])
            sst[w,b] = (tr_matr[components,b] * aScale[components]).sum() / deltaFreqs[w]
    return sst
```

dónde los parámetros que recibe la función son:

- `deltaFreqs`: Vector $(\Delta w)_k$ (K elementos).
- `borderFreqs`: Vector w_k ($K+1$ elementos).
- `aScale`: Vector $a_m^{-3/2}$ (K elementos).
- `wab`: Matriz $\omega_s(a_m,b_n)$ ($K \times B$ elementos).
- `tr_matr`: Matriz $W_s(a_m,b_n)$ ($K \times B$ elementos).
- `sst`: Matriz $S_s(w_k,b_n)$ ($K \times B$ elementos) $\leftarrow$ salida.

El rango `prange` indica que el ciclo puede ser paralelizado, por lo que la cantidad Γ de procesos depende de la cantidad de núcleos de procesamiento disponibles para *Numba*.

OpenCL

El último backend utilizado para la implementación fue *OpenCL*. Si bien éste impone la mayor cantidad de requerimientos en cuanto al stack de software, es el que ofrece la mayor capacidad de paralelismo y versatilidad. Para utilizar *OpenCL* es necesario que el hardware de la plataforma esté soportado y que el driver de se encuentre presente. Actualmente *OpenCL* es soportado por la mayoría de las compañías que desarrollan arquitecturas como Intel (y Altera), AMD (y Xilinx), Nvidia, ARM, etc. La mayor ventaja de este backend es que permite trabajar con plataformas heterogéneas, contemplando CPUs,

GPUs, FPGAs, y núcleos de procesamiento especializados. Esto permite lograr un alto grado de paralelismo y optimización en la ejecución de los algoritmos. El diseño está orientado a *kernels* los cuales son unidades atómicas de software, que se distribuyen en las unidades de procesamiento disponibles.

En base a 4.10 es posible entonces definir el siguiente *kernel* de OpenCL en lenguaje C para el cálculo de cada elemento de la matriz $S_s(w_k, b_n)$.

```
#define PYOPENCL_DEFINE_CDOUBLE
#include <pyopencl-complex.h>
__kernel void agregate(
    __global const double *deltaFreqs, __global const double *borderFreqs,
    __global const double *aScale,  __global const double *wab, __global const cdouble_t *tr_matr,
    __global cdouble_t *sst, __global int *width, __global int *height){
        int wd = *width;
        int hg = *height;
        int r_gid = get_global_id(0);
        int c_gid = get_global_id(1);
        int idx = c_gid + wd*r_gid;
        for(int w=0; w<hg; w++){
            if (wab[c_gid+w*wd] >= borderFreqs[r_gid] && wab[c_gid+w*wd] < borderFreqs[r_gid+1]){
                sst[idx] = cdouble_add(sst[idx], cdouble_mulr(tr_matr[c_gid+w*wd],
                                    aScale[w] / deltaFreqs[r_gid]));
                }
            }
        }
```

donde los parámetros que recibe la función son los mismos que la implementación en *Numba*, con la adición de:

- `width, height`: Ancho y alto de la matriz $S_s(w_k, b_n)$.

Con esta implementación, cada instancia del *kernel* recibe un *id* global que se utiliza como coordenada de fila y columna para computar el elemento correspondiente de la matriz $S_s(w_k, b_n)$. Cuando toda la matriz $S_s(w_k, b_n)$ ha sido computada, se finaliza la ejecución de la tarea. *OpenCL* contempla la distribución de los datos hacia distintos nodos de cómputo, por lo que el paralelismo implica copiar los datos a procesar hacia la memoria más cercana a los núcleos de procesamiento, reservar espacio en dicha memoria para los datos de salida, y finalmente cuando la tarea está finalizada volver a copiar los datos de salida (`sst` en este caso) a la memoria principal. En este caso la arquitectura de la implementación se muestra en la Figura 4.8.

La copia de datos entre memorias es un proceso altamente optimizado, típicamente realizado por un bus PCI-Express o AXI. La tasa de transferencia es muy elevada y para disminuir el impacto del *overhead*, se recomienda realizar la copia de datos en bloques de tamaño considerable. En este caso se mueven a la memoria dedicada $(\Delta w)_k$, $a_m^{-3/2}$, w_k, $\omega_s(a_k, b_n)$ y $W_s(a_k, b_n)$, y se copian de vuelta los resultados de $S_s(w_k, b_n)$.

Este *backend* permite correr el algoritmo no sólo sobre CPUs, sino también sobre GPUs, FPGAs, y otros dispositivos de cómputo.

Todas estas implementaciones se replicaron para las diferentes estrategias de reasignación de frecuencias mencionadas en la Sección 4.4.1.3 con los mismos conceptos involucrados.

4.4.1.5. Etapa adaptativa

Ambas variantes del algoritmo propuesto, ITL y OTL, utilizan los resultados previos de la función de *Synchrosqueeze* para analizar la matriz de la SST o de la WT respec-

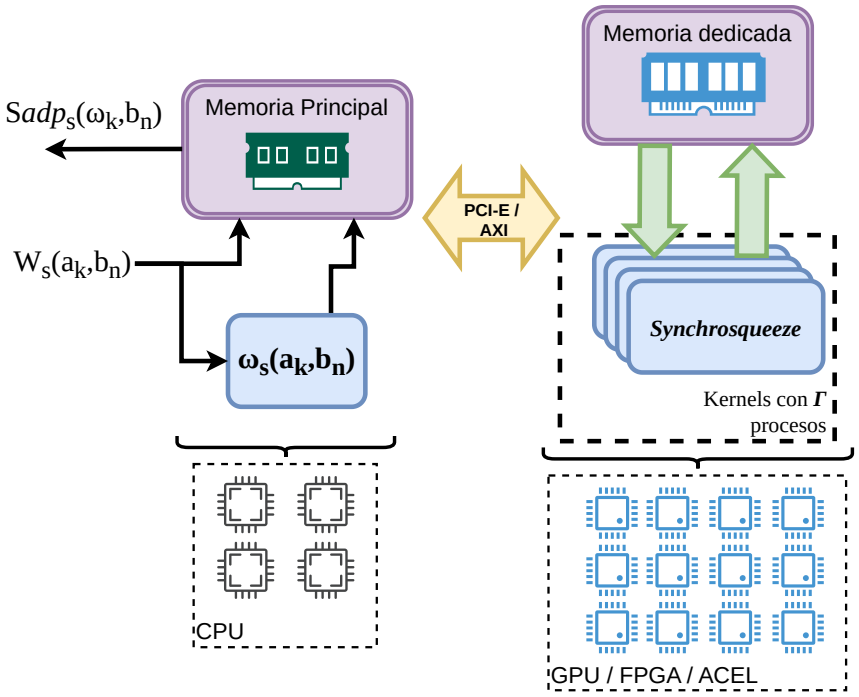


Figura 4.8: Estructura de la implementación de la operación de *Synchrosqueeze* en un esquema de OpenCL.

tivamente. Adicionalmente, los métodos de reasignación por *umbral* y *proporcional* no representan un esfuerzo computacional notable, siendo la operación de *Synchrosqueeze* la más crítica en este aspecto, especialmente porque esta operación iterativa de adaptación se realiza en un lazo. Por lo tanto, se provee al usuario el control del número máximo de iteraciones a realizar para adecuarlo tanto a la resolución como al esfuerzo computacional requerido. La estructura de la etapa de adaptación se muestra en la Figura 4.9.

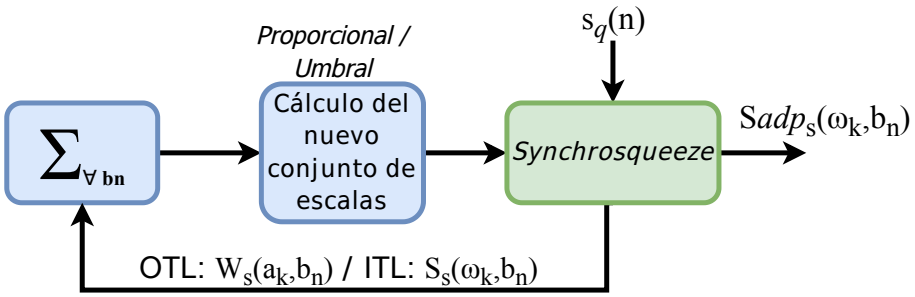


Figura 4.9: Perspectiva estructural de la etapa adaptativa

4.5. Resultados numéricos

En esta sección se presenta un análisis del algoritmo frente a señales sintéticas que permiten caracterizarlo. En la Sección 7.3 se presenta la aplicación del mismo al procesamiento de señales UWB reales. Se realizó la evaluación del algoritmo con las siguientes señales muestreadas a $T_s = 1/400s$ y con una longitud de señal de $t_1 = 12s$.

- Señal senoidal de 30 Hz:
  $s_1(nT_s) = sin(2\pi \cdot 30 \cdot nT_s)$.

- Señal chirp lineal entre 25 Hz y 35 Hz:
  $s_2(nT_s) = sin(2\pi f(nT_s) \cdot nT_s)$ con $f(nT_s) = 25 + (35-25) * n.T_s/t_1$.

- Señal chirp cuadrática entre 28 Hz y 32 Hz:
  $s_3(nT_s) = sin(2\pi f(nT_s) \cdot nT_s)$ con $f(nT_s) = 32 - (32-28).(t_1 - n.T_s)^2/t_1^2$.

- Señal bitono de 30 Hz y 40 Hz:
  $s_4(nT_s) = sin(2\pi \cdot 30 \cdot nT_s) + sin(2\pi \cdot 40 \cdot nT_s)$.

- Dos chirps cuadráticas entre 28 Hz y 32 Hz y entre 42 Hz y 38 Hz respectivamente:
  $s_5(nT_s) = sin(2\pi f_1(nT_s) \cdot nT_s) + sin(2\pi f_2(nT_s) \cdot nT_s)$.

donde $f_1(nT_s)$ y $f_2(nT_s)$ tienen la siguiente expresión:

$$f(n.T_s) = F_1 - (F_1 - F_0).(T_1 - n.T_s)^2/T_1^2,$$

con:

- para f_1 : $F_0 = 28\ Hz;\ F_1 = 32\ Hz;\ T_1 = 12\ s$
- para f_2 : $F_0 = 42\ Hz;\ F_1 = 38\ Hz;\ T_1 = 12\ s$.

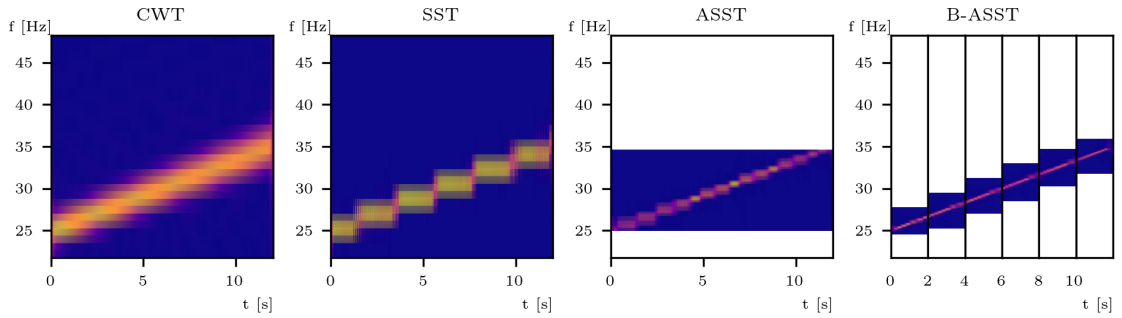
4.5.1. *Resolución* o *nitidez*

La Figura 4.10 muestra la representación TF de la aplicación del algoritmo a las señales chirp lineal (4.10a), bitono (4.10b) y chirp cuadrática dual (4.10c). Se puede observar como el algoritmo propuesto mejora la *resolución* o *nitidez* de la representación TF en comparación con la SST estándar.

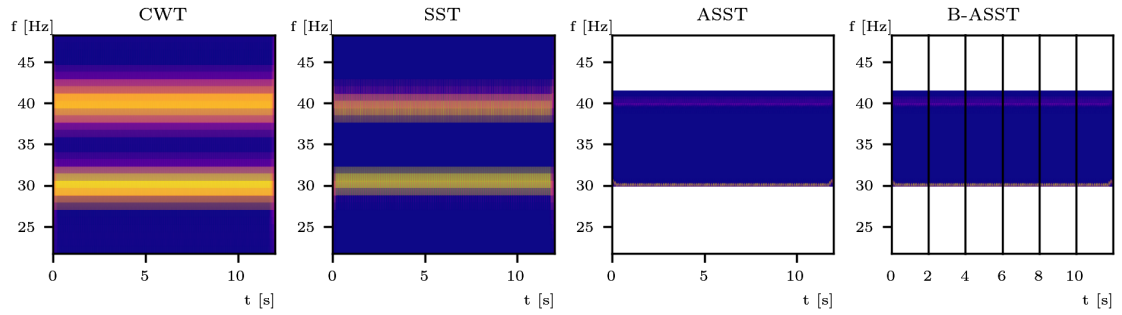
Para cuantificar el aumento en *resolución*, la entropía de Rènyi para cada representación TF de las señales de prueba se presenta en la Tabla 4.2. Los resultados muestran una representación TF más concentrada en la mayoría de las variantes del método propuesto. La excepción es el caso de un tono puro (s_1) donde el rango de análisis se estrecha fuertemente cerca del componente único, y por lo tanto todas las wavelets contribuyen en cierta medida a la entropía, como fue explicado anteriormente. Sin embargo, en esos casos un posterior análisis del MSE de la estimación de frecuencias instantáneas muestra una mejora significativa en la precisión de la estimación.

Tabla 4.2: Entropías de Rènyi para SST estándar y ASST propuesta, con $K = 12$.
Notación: I: ITL, O: OTL, P: Proporcional, U: Umbral y B: versión batched del algoritmo

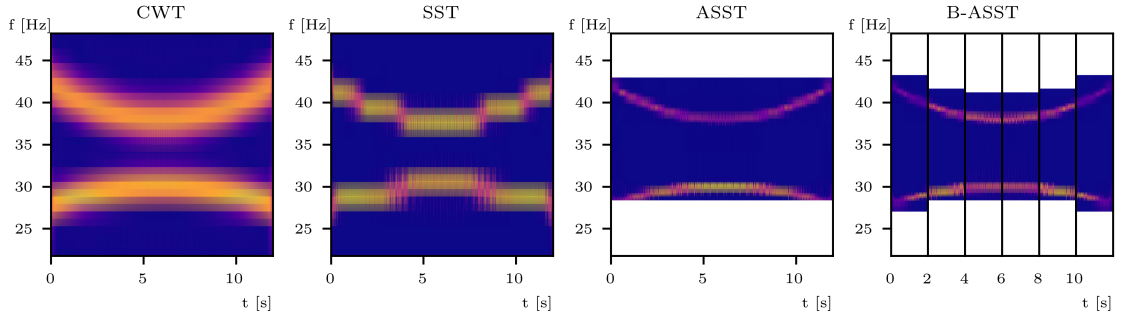
Señal	WT	SST	I-P	I-U	O-P	O-U	B-I-P	B-I-U	B-O-P	B-O-U
s_1	12.9577	12.2166	12.6419	12.6419	12.1937	12.6850	10.3131	10.1938	9.7759	10.0994
s_2	12.7356	12.1905	11.3054	11.6859	11.2075	11.2752	9.3362	9.3740	9.1314	9.2073
s_3	12.6964	12.2015	11.5420	11.3501	11.4429	11.5133	9.4556	9.3854	9.3976	9.3400
s_4	14.0828	13.2157	12.4786	12.4786	13.2920	12.8920	7.9174	9.8301	10.7070	10.3069
s_5	13.8389	13.2034	9.9393	10.1008	12.6061	11.2948	8.0115	7.8116	10.2053	10.0079



(a) Chirp lineal.



(b) Señal bitono.



(c) Dos chirps cuadráticas

Figura 4.10: Representaciones TF obtenidas a partir de diferentes métodos para las señales s_2, s_4 y s_5 con $K = 16$.

4.5.2. MSE de las estimaciones de frecuencias instantáneas

A partir de las representaciones TF obtenidas se puede realizar una estimación de las frecuencias instantáneas mediante un análisis de crestas para cuantificar el error de cada método mediante el cálculo del MSE correspondiente. La Figura 4.11 muestra la estimación obtenida a partir de dicho análisis y el MSE en función del tiempo para las mismas señales de la Figura 4.10. El MSE obtenido se encuentra notablemente por debajo del MSE calculado a partir de la estimación de las frecuencias instantáneas obtenida mediante la SST estándar[3]. Adicionalmente, las Figuras 4.12b y 4.12c muestran los resultados utilizando la variantes OTL e ITL del algoritmo para la señal s_4 respectivamente. Se puede observar que para el método ITL se produce una adaptación mucho más agresiva, que

[3]con excepción en el comienzo y final de la señal dónde los efectos de borde dominan en todos los casos.

para señales con poca variación temporal produce un error mucho menor.

La Tabla 4.3 resume los valores totales de MSE obtenidos para las estimaciones de frecuencia instantánea cuando se realiza un análisis de crestas. El método propuesto supera a la SST estándar para el mismo número de frecuencias de análisis, es decir, el mismo valor de K. Particularmente, bajo la mayoría de las condiciones, la reasignación proporcional ofrece un rendimiento ligeramente mejor que el umbral. Esto se debe principalmente al proceso totalmente automatizado, a diferencia del método de umbral, que requiere un proceso manual de ajuste fino para ajustar el nivel de detección.

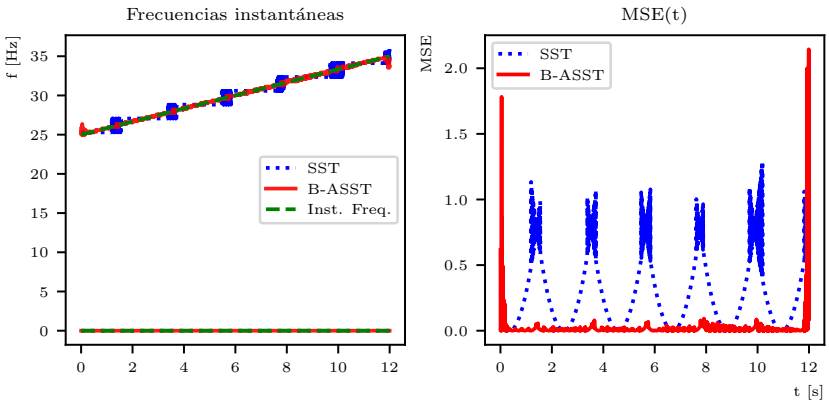
Tabla 4.3: MSE de las estimaciones de frecuencias instantáneas para la SST y la ASST propuesta, con $K = 12$.
Notación: I: ITL, O: OTL, P: Proporcional, U: Umbral y B: versión batched del algoritmo

Señal	WT	SST	I-P	I-U	O-P	O-U	B-I-P	B-I-U	B-O-P	B-O-U
s_1	0.3460	0.3465	0.0010	0.0010	0.0114	0.0117	0.0002	0.0003	0.0114	0.0117
s_2	0.2528	0.2695	0.0545	0.0545	0.0702	0.1656	0.0170	0.0159	0.0168	0.0166
s_3	0.2783	0.2878	0.0250	0.0939	0.0257	0.1491	0.0121	0.0160	0.0196	0.0172
s_4	0.7063	0.7329	0.0293	0.0196	0.0911	0.0658	0.0305	0.0287	0.0911	0.0658
s_5	0.6347	0.6838	1.7230	1.7367	0.1965	0.1042	0.6055	0.7439	0.0951	0.1027

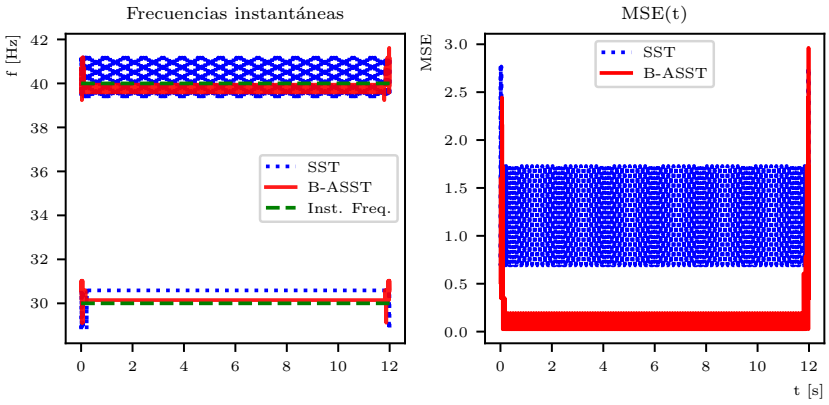
También se puede observar que para un número pequeño de iteraciones, ITL proporciona claramente una mejor resolución ya que puede concentrar las wavelets de una manera más compacta cerca de las bandas de interés. Sin embargo, el error introducido por este método también se puede observar en el caso de la señal conteniendo dos *chirps* cuadráticos. Esto está atribuido al hecho de que el procedimiento de distribución de frecuencias más agresivo de este algoritmo empuja el cono de influencia hacia las bandas de interés, como se discute en la Sección 4.3.3, lo que lleva a descartar erróneamente algunas frecuencias. Los resultados también muestran que la arquitectura *streaming* de la variante *Batched Adaptive Synchrosqueezing Transform* (B-ASST) del algoritmo disminuye este efecto ya que la señal se vuelve relativamente más estacionaria dentro de los *batches* y los límites de análisis no se estrechan en la banda de interés tan rápido como una señal que varía rápidamente. La Figura 4.12b muestra este efecto cuando el número de iteraciones máximas crece para s_5. En contrapartida, la Figura 4.12a muestra que este efecto no está presente para señales con variaciones más lentas como s_2. En general, la Figura 4.12 muestra que el rendimiento aumenta rápidamente, y se obtiene un valor estable con muy pocas iteraciones (1 o 2 según el caso)[4].

Para ilustrar más en profundidad, se expandió el análisis de la señal de *chirp* cuadrática dual s_5. La Figura 4.13 muestra las representaciones TF obtenidas con diferentes métodos. Se puede observar que se obtiene una representación más nítida cuando se utiliza el algoritmo de *streaming* adaptativo propuesto. Para todos los métodos se han utilizado $K = 16$ frecuencias de análisis, con la excepción del espectrograma, donde $K = 64$ para poder graficar la misma cantidad de frecuencias dentro del rango analizado, por lo que el rendimiento del espectrograma es muy limitado y se obviará de la discusión en este trabajo.

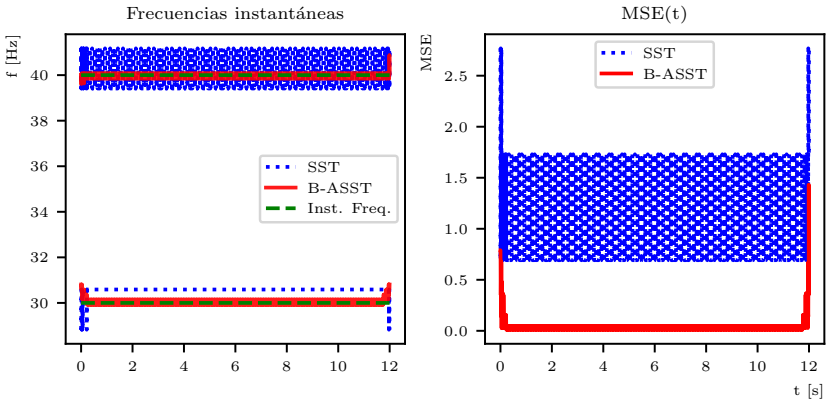
[4]Cabe aclarar que cuando se menciona "1 iteración" se hace referencia a una aplicación del método adaptativo, lo que involucra el cálculo de dos transformadas. El caso dónde no se realiza adaptación se lo referencia como "0 iteraciones".



(a) Chirp lineal (ITL).

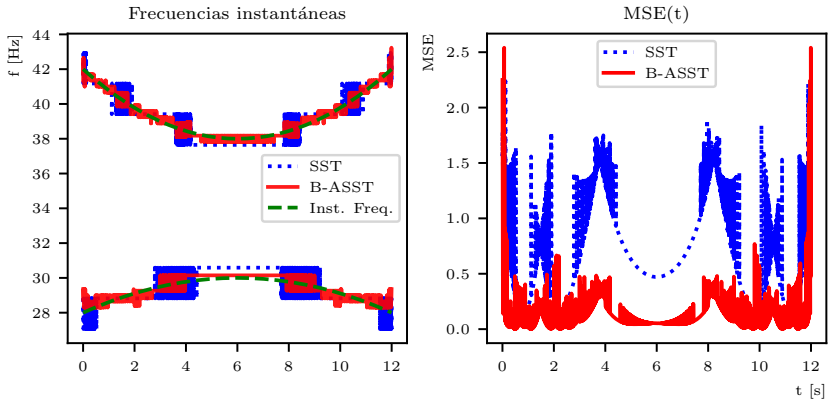


(b) Señal bitono (OTL).



(c) Señal bitono (ITL).

Figura 4.11: Estimación de las frecuencias instantáneas y MSE en función del tiempo para las señales s_2, s_4 y s_5 con $K = 16$.



(d) Dos chirps cuadráticas (OTL)

Figura 4.11: Estimación de las frecuencias instantáneas y MSE en función del tiempo para las señales s_2, s_4 y s_5 con $K = 16$ (cont.).

La Tabla 4.3 muestra el MSE obtenido con WT y SST en comparación con el algoritmo ASST propuesto y la versión *batched* (B-ASST) en todas sus variantes. Los resultados muestran un error de estimación menor para la ASST con la excepción de la combinación ASST-ITL/proporcional debido a la limitación mencionada anteriormente, lo que lleva a descartar erróneamente algunas frecuencias.

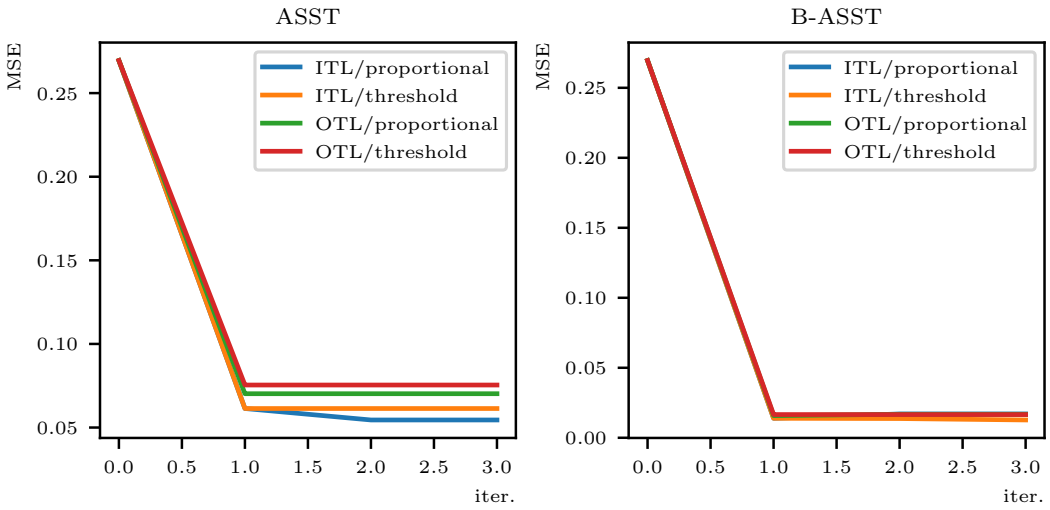
Tabla 4.4: MSE de las estimaciones de frecuencia instantánea usando detección de crestas para diferentes métodos con $s(n) = s_5(n)$.

	ITL/proporcional	ITL/umbral	OTL/proporcional	OTL/umbral
WT	0.6883	0.6883	0.6883	0.6883
SST	0.7029	0.7029	0.7029	0.7029
ASST	2.455	2.4643	0.1978	0.1333
B-ASST	0.4677	0.6054	0.1106	0.1029

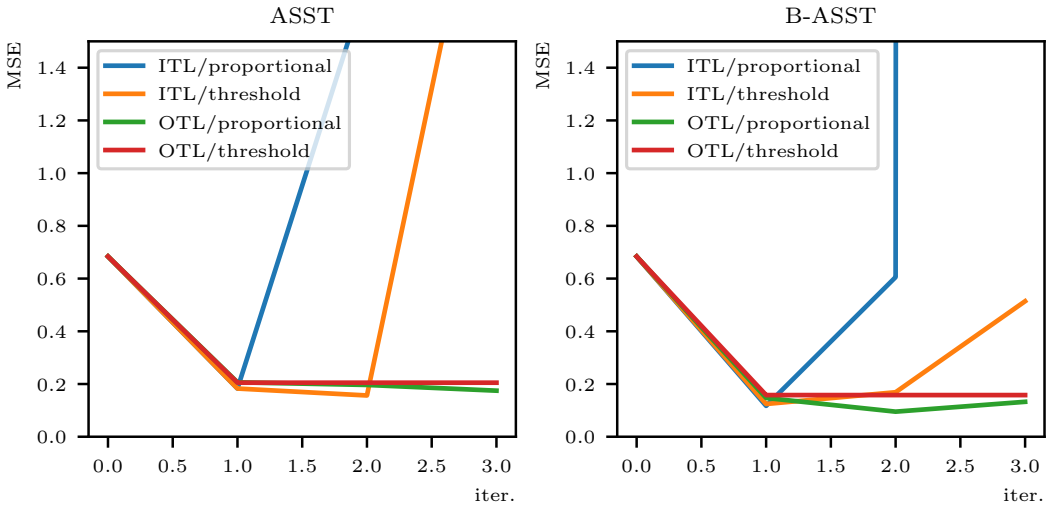
También se ha estudiado el rendimiento del algoritmo propuesto al analizar señales ruidosas. En particular, la Figura 4.14 muestra los resultados para $s_5(nT_s)$, que es la señal de *chirp* cuadrática dual. Se puede observar que el método propuesto obtiene una ganancia notable en comparación con la SST estándar. Particularmente, la variante ITL tiene un rendimiento ligeramente mejor debido a su adaptación más agresiva.

4.5.3. Rendimiento Computacional

El rendimiento computacional depende fuertemente de la arquitectura de hardware y la configuración, así como del *software stack* en ejecución. El objetivo particular es evaluar la aplicabilidad del algoritmo propuesto a problemas en tiempo real, por lo que se ha centrado el análisis en la frecuencia máxima de muestreo manejable por varias plataformas para funcionamiento en dicho modo. Particularmente, se analizó la relación entre el tiempo de ejecución y la duración del segmento de señal (es decir, el tamaño de la entrada).



(a) Chirp lineal



(b) Doble chirp cuadrática

Figura 4.12: MSE vs iteraciones para las variantes ITL y OTL de ASST y *batched*-ASST con $K = 12$

Para observar la relación mencionada, se registró el tiempo de ejecución de los algoritmos utilizando una señal muestreada a $f_s = 2kHz$ con una longitud máxima que varía de 40 a 400s. Para mediciones de tiempo de ejecución, se realizaron 10 llamadas a las mismas funciones con las mismas entradas y el resultado final se obtuvo promediando los tiempos de ejecución. En cada configuración se cronometró la SST simple y la ASST completa (en ejecuciones separadas). Dado que la etapa adaptativa se ejecuta una vez por iteración, y la SST se ejecuta un número de veces igual a $Iteraciones + 1$ para la versión ITL del algoritmo[5], el tiempo de ejecución de la etapa adaptativa se puede obtener como

[5]ITL resulta el peor caso para el rendimiento computacional, por lo que se utiliza dicho método para esta sección salvo que se indique lo contrario.

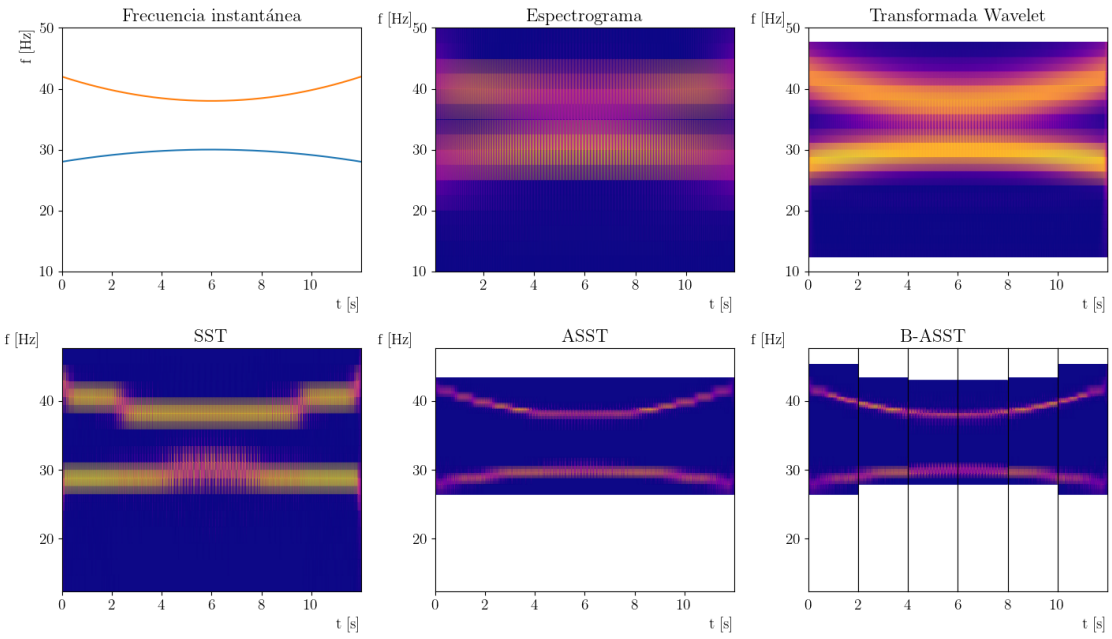


Figura 4.13: Análisis de una *chirp* cuadrática dual para $K = 16$.

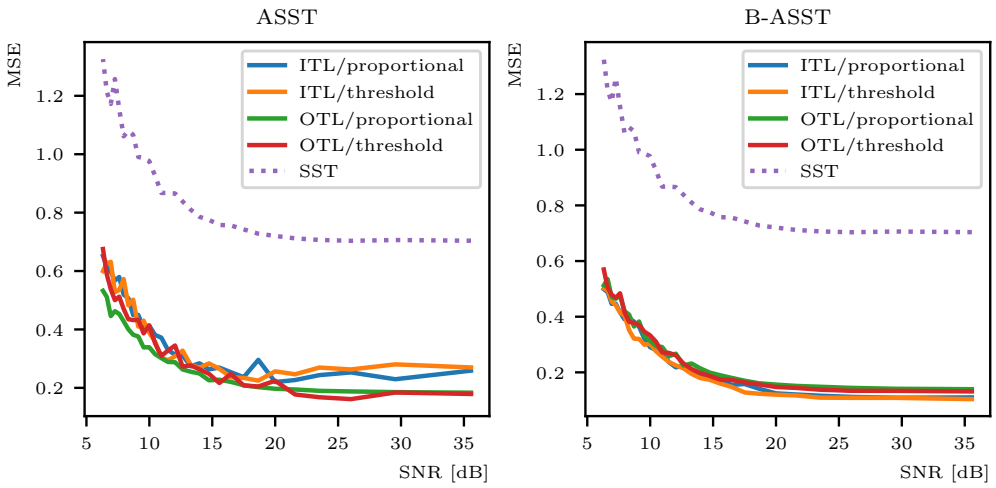


Figura 4.14: MSE vs SNR, para una *chirp* cuadrática dual con $K = 16$

$$\text{Tiempo de ejecución de la etapa adaptativa} = \frac{\text{Tiempo de ejecución de ASST}}{\text{iteraciones}} - \frac{(\text{Tiempo de ejecución de una única SST}) * (\text{iteraciones} + 1)}{\text{iteraciones}}.$$

La Figura 4.15 muestra los resultados del tiempo de ejecución de la ASST completa para distintos valores de iteraciones máximas junto con el tiempo de ejecución de $(Iteraciones + 1)$ veces el tiempo de una sola SST. La plataforma utilizada para estas mediciones fue un procesador AMD Ryzen 7 5700G (8 núcleos - 16 hilos) corriendo un kernel de Linux 6.8.0-41-generic. Cuando la longitud de la señal es demasiado corta, resulta más evidente el *overhead* relacionado al paralelismo, lo que produce dispersión en los resultados de tiempo y, por ejemplo, la ASST completa ocasionalmente se calcula en menos tiempo que (Iter. + 1) SSTs. Siendo ese *overhead* un tiempo típicamente constante,

multiplicar $(Iteraciones + 1)$ veces al tiempo de ejecución de una única SST multiplica de igual manera ese *overhead*, lo que arroja en ciertos casos un tiempo de ejecución mayor que el de la ASST completa.

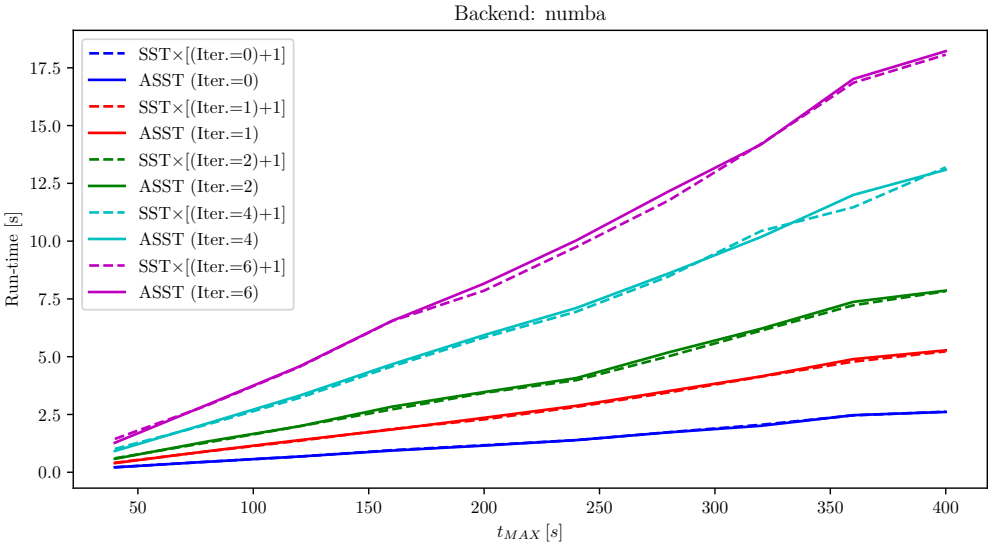


Figura 4.15: Tiempos de ejecución para el algoritmo completo y para $(Iteraciones + 1)$ veces una única SST, como función del tamaño de la entrada.

Se puede observar que el tiempo de ejecución de la etapa adaptativa es despreciable en comparación con el tiempo de ejecución de la SST. También se puede notar que el cuello de botella computacional de la implementación propuesta es el proceso de *synchrosqueezing*, como se explicó en la Sección 4.4.1.3.

Adicionalmente, al evaluar el tiempo de ejecución de la transformada wavelet provista por el paquete `PyWavelets`, se observó que el mayor tiempo se consume en dicha operación. La Figura 4.16 muestra el tiempo de ejecución de la transformada wavelet junto al tiempo de ejecución de la ASST para la misma plataforma[6].

Si se prueba el algoritmo en una plataforma levemente más limitada, como un procesador Intel Core i7-8650U (4 núcleos - 8 hilos), se observa en la Figura 4.17 que el tiempo de ejecución de la transformada wavelet no sufre grandes variaciones. Esto implica una poca optimización de dicho desarrollo en cuanto al soporte de computo o plataforma a utilizar.

Por lo analizado en esta sección, el proceso adaptativo completo está atado en tiempo de ejecución a $I + 1$ veces el tiempo de *synchrosqueezing*, siendo I el número máximo de iteraciones configurado. Para evaluar entonces el rendimiento del algoritmo, se probó la propia implementación de la SST (que incluye la CWT del paquete `PyWavelets`). Para ello, se calculó la frecuencia máxima de muestreo para funcionamiento en tiempo real como:

$$fs_{MAX} = \left[\left(\frac{\text{Tiempo total de ejecución}}{D} \right) \frac{1}{t_{MAX}} \right] \times f_s, \tag{4.12}$$

donde *Tiempo total de ejecución* es el tiempo que toma correr D veces el algoritmo con

[6] AMD Ryzen 7 5700G (8 núcleos - 16 hilos)

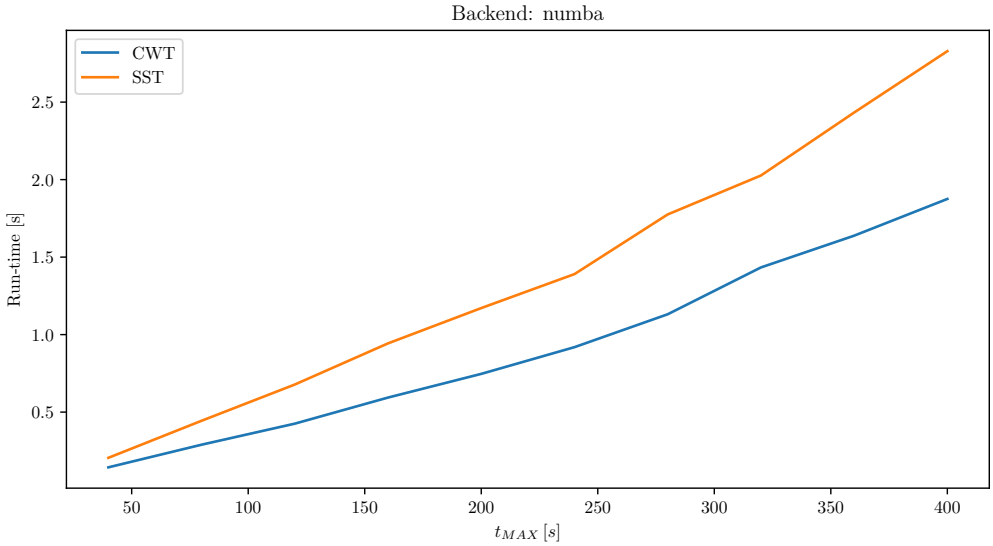


Figura 4.16: Tiempos de ejecución para la transformada wavelet y la ASST para una señal de $400\,s$ muestreada a $2\,kHz$ con $K = 32$.

las mismas entradas, t_{MAX} es la longitud de la señal en segundos y f_s es la frecuencia de muestreo de la misma.

La Figura 4.18 muestra los resultados de iterar sobre diferentes longitudes de la señal y una cantidad creciente de procesos en paralelo.

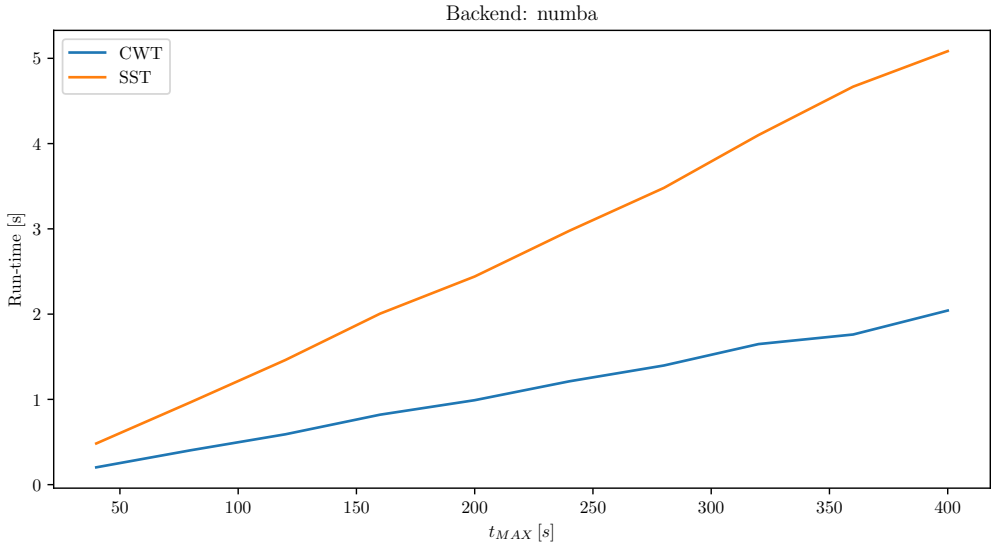
Los resultados muestran que se obtiene un gran aumento de rendimiento al paralelizar con pocos hilos (aproximadamente hasta 5). Con un número más elevado de procesos en paralelo, la ganancia no aumenta proporcionalmente debido a la falta de optimización de la implementación utilizada de la transformada Wavelet. Sin embargo, la arquitectura propuesta presenta flexibilidad, de forma de obtener un buen rendimiento incluso con un bajo número de caminos paralelos y adicionalmente muestra un escalado adecuado de rendimiento frente al aumento de capacidad de cómputo.

4.5.4. Comparación con otros métodos

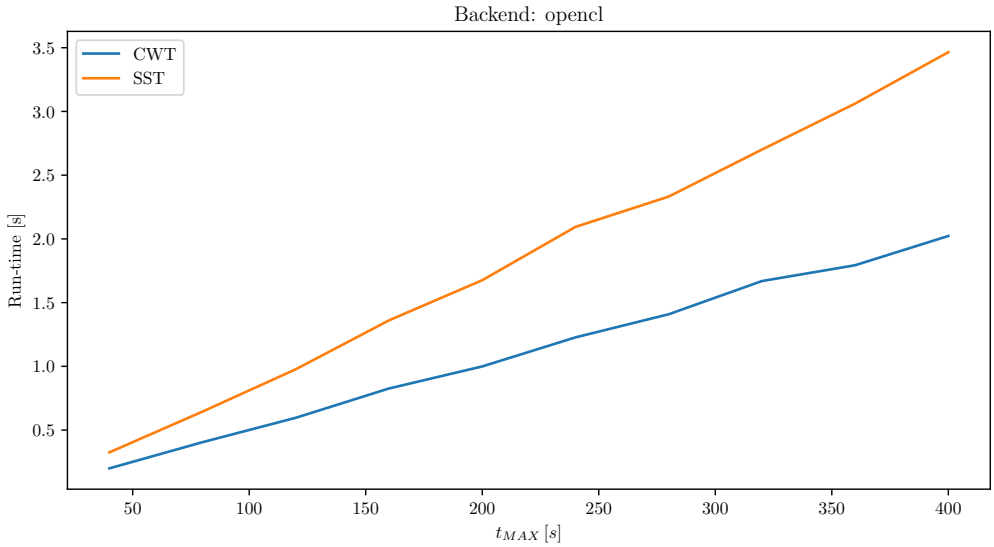
Para comparar el rendimiento del algoritmo propuesto con otros métodos en diferentes casos, se realizaron simulaciones similares con el método propuesto por Li L., Jiang Q. et al [54]. El mismo se encuentra disponible en `http://www.math.umsl.edu/~jiang/Jsoftware_SST.htm`. Este método está implementado en Matlab y cómo fue mencionado en la Sección 3.1, se basa en el refinamiento de los parámetros de la wavelet.

En el primer experimento se realizaron simulaciones con los parámetros y las señales por defecto ($K = 256$ frecuencias de análisis y dos *chirps*) incluidos en el paquete de simulación. Los resultados se muestran en la Figura 4.19.

Se puede observar que se obtiene una representación más nítida con el algoritmo de Li L. et al. Sin embargo, el tiempo de ejecución para ambos métodos resulta de $98{,}179ms$ para el algoritmo propuesto y $52{,}56s$ para el algoritmo de Li L. et al. Adicionalmente cuando el número de frecuencias de análisis se reduce drásticamente ($K = 24$), sólo el algoritmo presentado en este trabajo es capaz de proporcionar una representación nítida de una *chirp* cuadrática dual, como se muestra en la Figura 4.20. Esto se debe a que el



(a) Intel Core i7-8650U.



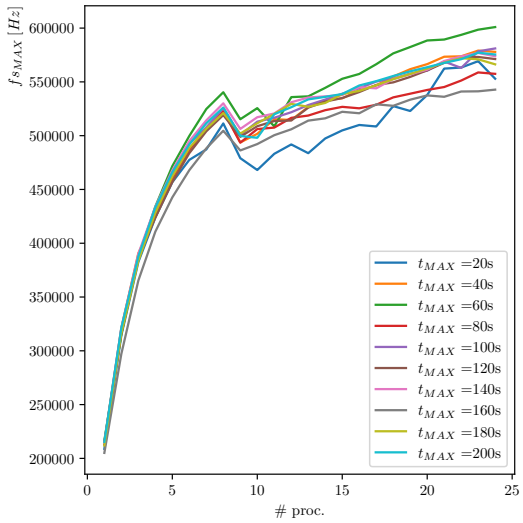
(b) Nvidia Quadro P500.

Figura 4.17: Tiempos de ejecución para la transformada wavelet y la SST para una señal de $400\,s$ muestreada a $2\,kHz$ con $K = 32$.

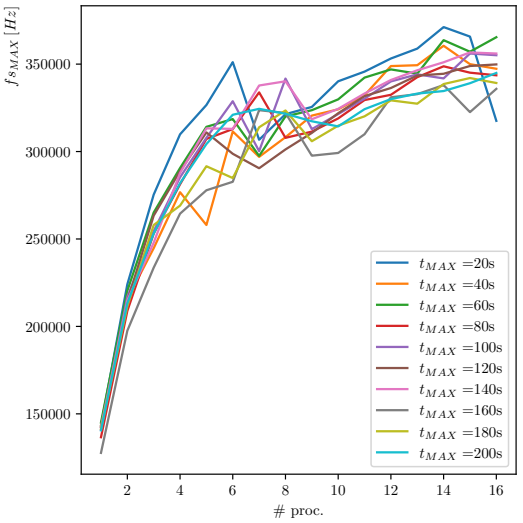
algoritmo desarrollado reasigna la grilla de frecuencias de análisis a las bandas relevantes.

La Tabla 4.5 resume los tiempos de ejecución y las Entropías de Rènyi para ambos algoritmos. Si bien la Entropía de Rènyi es menor para el algoritmo de Li L. et al. en las simulaciones con los parámetros por defecto, particularmente la condición para $K = 24$ representa un caso dónde toda la energía de las componentes cae mayormente en una única frecuencia de análisis, produciendo casi nula resolución. Por otro lado, el algoritmo propuesto en esta tesis adapta la grilla de frecuencias de análisis, ajustando también el rango, produciendo una representación con mayor resolución.

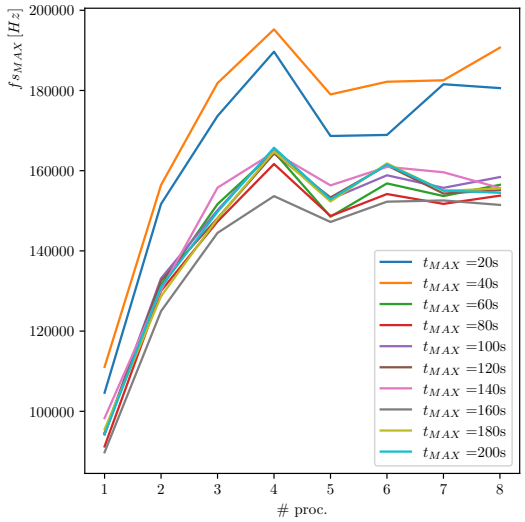
El algoritmo propuesto en [54] tiene una muy buena performance en cuanto a resolución en condiciones de procesamiento *offline* donde se puede utilizar un número elevado



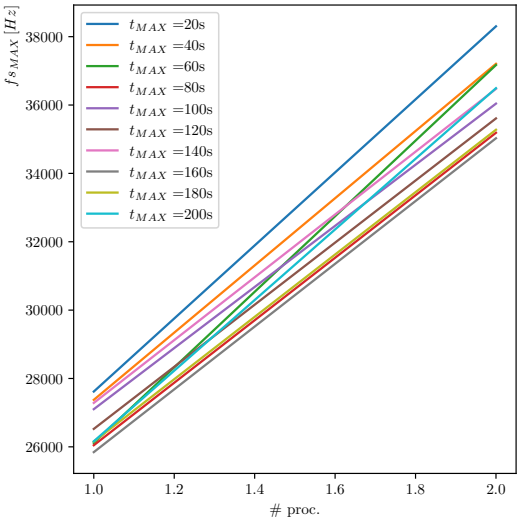
(a) Core i9 12900k (16 núcleos - 24 hilos)



(b) Ryzen 7 5700G (8 núcleos - 16 hilos)



(c) Core i7 8650U (4 núcleos - 8 hilos)



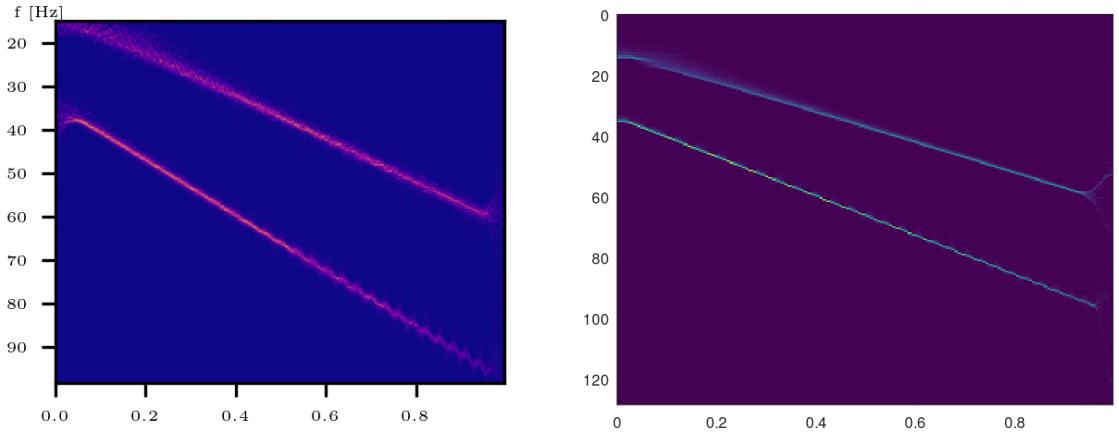
(d) Dual Core T4400 (2 núcleos - 2 hilos)

Figura 4.18: Tiempos de ejecución por muestra de la señal de entrada como función del número de procesos en paralelo.

Señal	f_s $[Hz]$	B	K	Tiempo de ejecución (ASST)	Tiempo de ejecución (Li et al.)	Entropía de Rènyi (ASST)	Entropía de Rènyi (Li et al.)
Chirp lineal dual	256	256	256	98.179 ms	52.56 s	10.266	8.026
Chirp cuadrática dual	256	4096	24	103.14 ms	364.641s	11.916	11.934

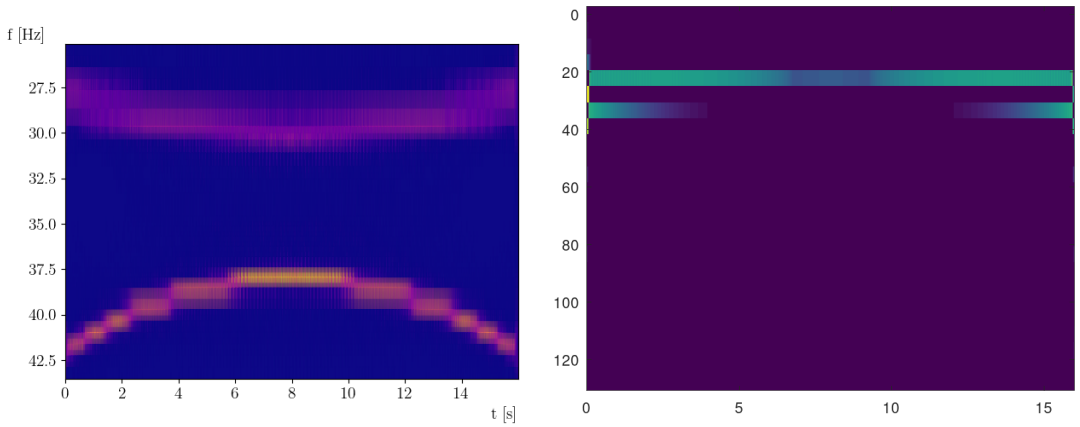
Tabla 4.5: Comparación de tiempos de ejecución y entropías de Rènyi para el algoritmo propuesto y el algoritmo de Li L., Jiang Q. et al.

de frecuencias de análisis. El enfoque hacia tiempo real o procesamiento *online* del algoritmo propuesto en este trabajo, lo ubica en una posición con cierta vacancia en el ecosistema de algoritmos de análisis TF.



(a) ASST (b) Li L., Jiang Q. et al

Figura 4.19: Comparación de la representación de una *chirp* cuadrática dual para $K = 256$.



(a) ASST (b) Li L., Jiang Q. et al

Figura 4.20: Comparación de la representación de una *chirp* cuadrática dual para $K = 24$.

Capítulo 5

Plataformas de hardware para radar UWB

RESUMEN: En este capítulo se presenta el estado del arte del desarrollo de plataformas de hardware para radar UWB. En la Sección 5.2 se hace una revisión de los trabajos de investigación y desarrollo de plataformas UWB. En la Sección 5.3 se enumeran las plataformas comerciales disponibles que surgieron durante el desarrollo de esta tesis y sus características. Finalmente, en la Sección 5.4 se discute el problema asociado a la adquisición de señales experimentales.

5.1. Introducción

El desarrollo de hardware para la generación de señales UWB se puede remontar a los primeros transmisores de chispa de Guillermo Marconi [74], pero la patente presentada por Ross [75], en 1973 puede considerarse como una de las bases para el comienzo del desarrollo del hardware propiamente dicho para esta tecnología. Sin embargo, el desarrollo de una plataforma transmisora-receptora de señales UWB es una tarea compleja debido a, como fue mencionado anteriormente, el gran ancho de banda proporcional que poseen las señales involucradas. Esto conlleva dificultades tecnológicas asociadas a la disponibilidad de componentes que soporten dichos requerimientos. Adicionalmente, las técnicas de diseño tradicionales no aplican a sistemas UWB donde el ancho de banda es proporcional a la frecuencia central de operación. Todo esto llevó a que el desarrollo de la tecnología no tuviera avances significativos hasta la década del 2000.

En el año 2002, dado el avance de la tecnología asociada a componentes y técnicas para comunicaciones, la FCC autorizó el uso de señales UWB para aplicaciones civiles. Esto dio un segundo impulso a la investigación en el área. En 2003 el trabajo de Yang y Giannakis [4] resume las técnicas de procesamiento señales utilizadas en sistemas UWB y hace un recuento de ciertos desarrollos en el área, resultando uno de los trabajos más citados en la literatura. En los años subsiguientes se produjo el surgimiento de muchos trabajos de investigación explorando diferentes técnicas de diseño y arquitecturas de hardware para lograr desarrollar una plataforma UWB. Dado el objetivo de este trabajo, en la Sección 5.2 se exploran trabajos actuales de autores focalizados en el diseño de plataformas impulsivas. Se prestó especial atención a la disponibilidad de componentes, costo y capacidades de procesamiento.

Durante el desarrollo de la presente tesis surgieron unas pocas plataformas comerciales, de las cuales se enumeran sus características en la Sección 5.3. Todas ellas contienen la implementación del *front-end* analógico integrado en un chip personalizado, y se comercializan como soluciones de alto nivel para, por ejemplo, detección de personas en interiores y domótica. Esto lleva a la discusión encarada en la Sección 5.4 sobre la disponibilidad de datos crudos de señales UWB para su procesamiento y análisis.

5.2. Trabajos de investigación previos

Uno de los primeros trabajos de implementación de una plataforma actual *Ultra Wide-Band Impulse Radar* (UWB-IR) con aplicación a *wireless body area network* (WBAN) fue [76] y data del año 2005. En el mismo se presenta solamente la arquitectura del generador de pulsos y del transmisor, y se integran en silicio sobre tecnología CMOS. La arquitectura general propuesta por los autores se muestra en la Figura 5.1. La misma consta de un generador de pulso triangular, el cual es modulado mediante un oscilador en anillo a través del multiplicador. El oscilador es controlado por un circuito de disparo (*gating*) y fue implementado usando una topología de tipo anillo debido a sus rápidos tiempos de encendido. La elección de un pulso triangular por parte de los autores se basa en la dificultad que introduce la generación de pulso gaussiano, que típicamente requiere de componentes especiales y limita su configurabilidad. Adicionalmente, un pulso cuadrado generaría lóbulos laterales en el espectro cuyo filtrado reduce la eficiencia del sistema. Este sistema fue desarrollado para transferencia de datos mediante modulación por posición de pulso o *Pulse Position Modulation* (PPM) y no prevé su uso para sensado.

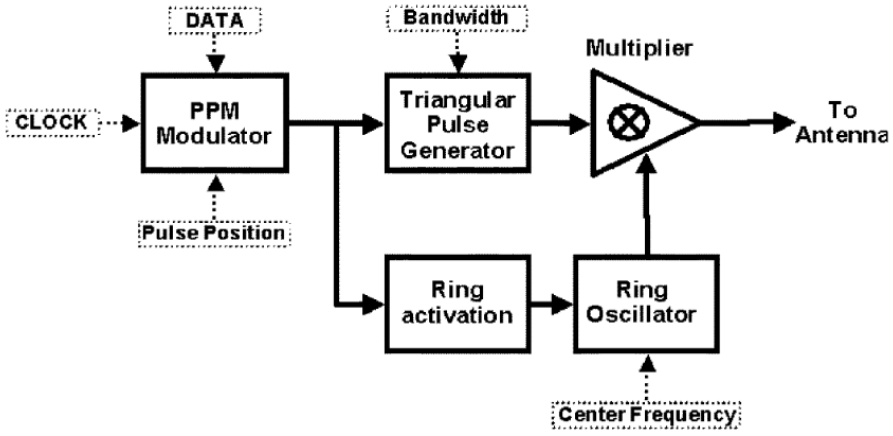


Figura 5.1: Arquitectura del transmisor propuesta por Ryckaert et al[1].

Las especificaciones obtenidas por éste y los demás trabajos analizados en esta sección se encuentran en la tabla comparativa 5.1.

En cuanto al desarrollo de plataformas en base a componentes disponibles en el mercado (*off-the-shelf*) De Angelis et al [77] propusieron una plataforma de hardware UWB para aplicaciones de localización. La misma transmite un pulsos monociclo con un tiempo de crecimiento de $0{,}8ns$ y una duración de $1{,}3ns$ a una frecuencia central de $1GHz$, logrando un ancho de banda proporcional de aproximadamente $70\,\%$. Su funcionamiento depende de dos dispositivos idénticos, con un retardo programable. Cada dispositivo

[1] Imagen extraída de [76].

retransmite un pulso luego de recibir un pulso por su antena receptora y aplicar el mencionado retardo. De esta forma, su estimación de distancia se basa en la medición de la frecuencia de repetición de los pulsos recibidos, que es proporcional al tiempo de vuelo de la señal entre prototipos. La arquitectura de la plataforma se muestra en la Figura 5.2.

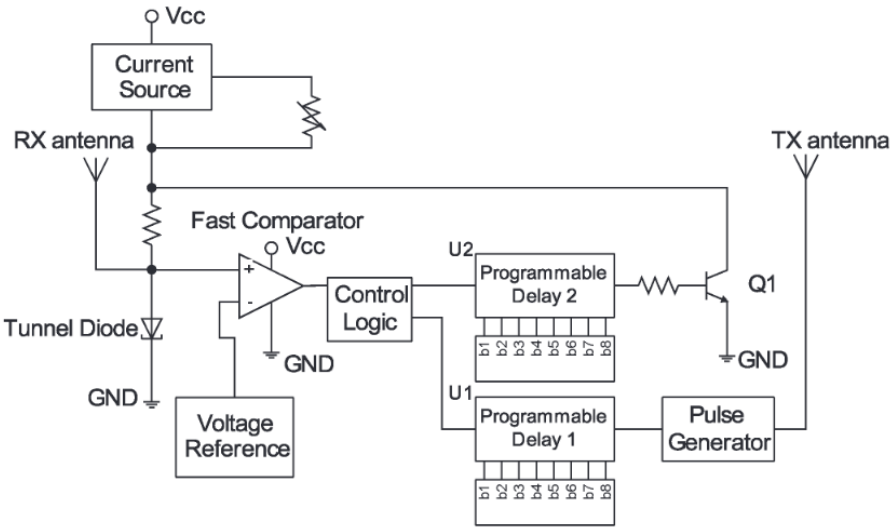


Figura 5.2: Arquitectura del transmisor propuesta por De Angelis et al[2].

Se puede observar que la plataforma propuesta no presenta facilidades de digitalización de las señales recibidas. La medición de la frecuencia de repetición de pulsos se realiza mediante instrumental de laboratorio. Más allá de estas limitaciones, resulta interesante observar el uso de un transistor convencional disparado en modo avalancha para la generación del pulso. También se destaca la propuesta de implementación enteramente con componentes *off-the-shelf*, logrando un ancho de banda proporcional elevado.

La primera implementación de una plataforma UWB-IR completa y detallada con fines científicos, al mejor saber del autor de esta tesis, es la desarrollada por Colli-Vignarelli [78] en el año 2012. En dicha tesis se presenta un desarrollo detallado de toda la cadena de transmisión-recepción orientado a la replicabilidad y configurabilidad. La misma incluye una FPGA tanto para el potencial procesamiento de los datos muestreados cómo para el control de la generación del pulso, y se basa en un transmisor/receptor coherente en cuadratura. La arquitectura general se muestra en la Figura 5.3.

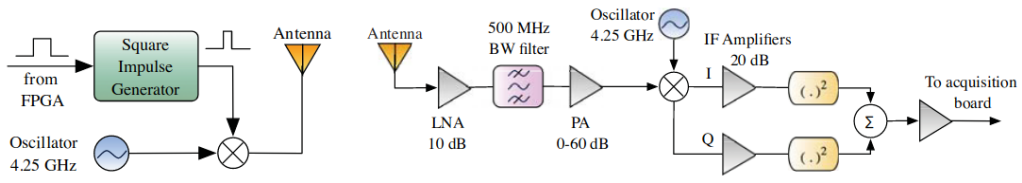


Figura 5.3: Arquitectura del transmisor propuesta por Colli-Vignarelli[3].

En el segundo prototipo presentado en dicho trabajo, la cadena de transmisión comienza con un pulso de 6*ns* generado por la FPGA que se utiliza como disparador de un

[2]Imagen extraída de [77].
[3]Imagen extraída de [78].

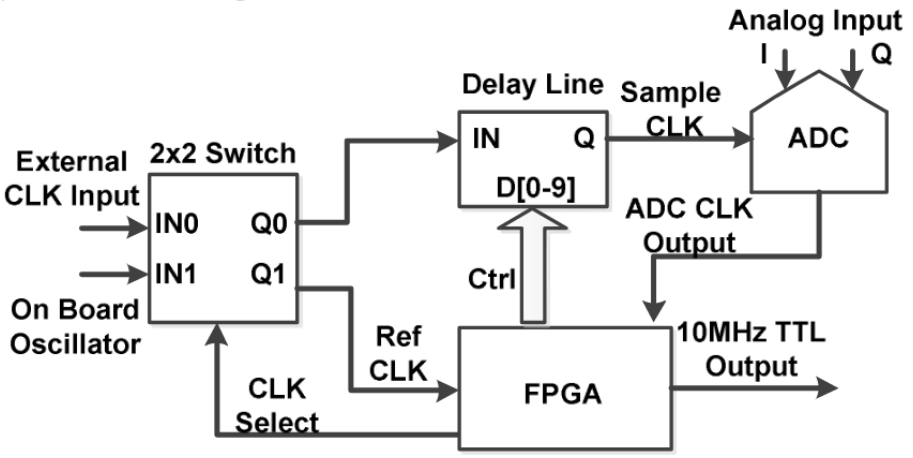


Figura 5.4: Esquema de muestreo en tiempo equivalente propuesto por Liu et al[4].

circuito que reacondiciona al mismo para reducir su ancho $3ns$. Dicho circuito conformador se basa en comparadores rápidos y *Schmitt triggers* disponibles comercialmente. El pulso es modulado en frecuencia a través de un mezclador por un oscilador a $4{,}25GHz$ para ser trasmitido.

La estructura del receptor comienza con un amplificador de bajo ruido o *low noise amplifier* (LNA) de $10dB$ de ganancia que amplifica la señal recibida desde la antena y la envía a un filtro pasa-banda de $500MHz$ de ancho de banda centrado en $4{,}25GHz$. Luego, una segunda etapa de ganancia ajustable de hasta $60dB$ es introducida para finalmente alimentar un mezclador que realiza la demodulación de la señal recibida en cuadratura que incluye la suma cuadrática de I y Q. Finalmente, es muestreada por una placa adquisidora *Acquiris AC-240* a una tasa de $2GS/s$ y $8bits$ de resolución. Dicha adquisidora posee la FPGA *Virtex-II Pro* de Xilinx que se utiliza para el procesamiento de la señal recibida y funciona a una frecuencia máxima de $166MHz$, lo que produce a su vez el pulso de $6ns$ mencionado anteriormente.

Es importante destacar que la conversión *A/D* se realiza con una placa adquisidora comercial, elevando el costo de la plataforma y limitando su versatilidad en cuanto a la experimentación con la tecnología.

En cuanto al muestreo, Liu et al [79] (2012) propusieron un módulo de muestreo de 100 GS/s orientado a la generación de imágenes de radar en tiempo real. El módulo de muestreo fue probado con una plataforma similar a la presentada en [78] operando con una frecuencia central de $3GHz$ y un ancho de pulso de $0{,}7ns$. Las principales mejoras en cuanto a las especificaciones de este trabajo se basan mayormente en nuevas tecnologías de componentes, por ejemplo, el uso de una FPGA *Virtex-5* en lugar de la *Virtex-II Pro* utilizada en [78]. El módulo de muestreo diseñado por Liu et al. utiliza la técnica de muestreo en tiempo equivalente implementada mediante una línea de retardo comercial como se muestra en la Figura 5.4.

Durante el desarrollo del presente trabajo, Van Herbruggen et al. [13] (2019) propusieron una plataforma totalmente *open source* que incluye un *backbone* de radio y un *stack* de red para la localización de objetos en interiores. La misma está basada en el chip *Decawave DW1000*[5] y es un diseño de integración. Esta pensada exclusivamente para

[4]Imagen extraída de [79].
[5]Ver Sección 5.3

localización activa, es decir, el sistema general está compuesto por nodos "ancla", cuya posición es conocida, y nodos "tag" que son los que se desea localizar. El usuario no tiene acceso a los datos crudos de las señales, por lo que no es posible realizar un procesamiento de las mismas para obtener información adicional.

Trabajo	**Frecuencia central**	**Ancho de pulso**	**Potencia**	**Muestreo**	**Disponibilidad de componentes**
[76]	3,432, 3,960 y 4,488GHz (medido)	1,1 a 4,5ns	2mW	-	No: silicio a medida
[77]	1GHz	2,1ns	N/D	No, sólo detección	Si, *off-the-shelf*
[78]	4,25GHz	3ns	10dBm	2GS/s placa adquisidora	Si, *off-the-shelf*
[79]	3 GHz	0,7ns	25dBm	100GS/s, T.E.	Si, *off-the-shelf*
[13][6]	3,5 a 6,5GHz	0,74ns TX - 2ns RX	-35 dBm/MHz	N/D	*Decawave DW1000*

Tabla 5.1: Tabla comparativa de trabajos analizados.

La Tabla 5.1 muestra un resumen de algunas especificaciones de los trabajos mencionados. Se puede observar que todos los trabajos presentan en cierta medida una vacancia en cuanto al desarrollo de una plataforma adecuada para la experimentación con la tecnología UWB. El trabajo de Ryckaert, siendo uno de los primeros, propone una solución de silicio a medida del transmisor solamente. De Angelis introduce una solución que incluye el transmisor y receptor, y adicionalmente es posible implementarla con componentes *off-the-shelf*. Sin embargo, está puramente orientada a localización por lo que no incluye una etapa de muestreo de la señal UWB. Por su parte, las propuestas [78] y [79] son las más completas y reproducibles aunque incluyen componentes cerrados de alto nivel de integración (por ejemplo, placa adquisidora) y alta complejidad, respectivamente. Por último, la propuesta más actual [13] está orientada a la implementación del *stack* completo de localización e integración con servicios de red.

5.3. Plataformas comerciales

Aproximadamente, en simultáneo con el comienzo del presente trabajo, han surgido las primeras soluciones comerciales para UWB. Todas ellas basadas en circuitos integrados personalizados, que incluyen el *front-end* analógico y algún tipo de muestreo y/o digitalización. Principalmente, los actores involucrados en esos desarrollos son *Decawave* (actualmente *Qorvo*), *Time-domain* (actualmente *Humatics*) y *Novelda*. Otras compañías como *Ubisense* y *Bespoon* también desarrollaron soluciones comerciales, pero orientadas a aplicaciones particulares y no ofrecen plataformas de desarrollo.

[6]Especificaciones determinadas por el módulo *Decawave DW1000*

Como se menciona en el párrafo anterior, dos de las compañías que desarrollaron transceptores UWB (*Decawave* y *Time-domain*) fueron absorbidas durante el desarrollo de este trabajo orientándose más a la comercialización de soluciones empaquetadas. Adicionalmente, *Novelda* ya no ofrece su kit de desarrollo de forma comercial, solamente está disponible para clientes que deseen adquirir gran cantidad de sus integrados para producción. Esta situación produce que al momento de escribir este trabajo, la accesibilidad a plataformas experimentales o de desarrollo sea más limitada que al momento de ejecución del presente desarrollo.

A continuación se mencionan las características más importantes de cada solución comercial.

Decawave DW1000

El integrado *DW1000* de *Decawave*[14] es un transceptor UWB, orientado a la implementación de sistemas de localización en interiores y comunicaciones de corto alcance. Implementa completamente el estándar *Institute of Electrical and Electronics Engineers* (IEEE) 802.15.4-2011 [80] para comunicaciones de baja potencia para redes de área personal. La arquitectura del transceptor se muestra en la Figura 5.5.

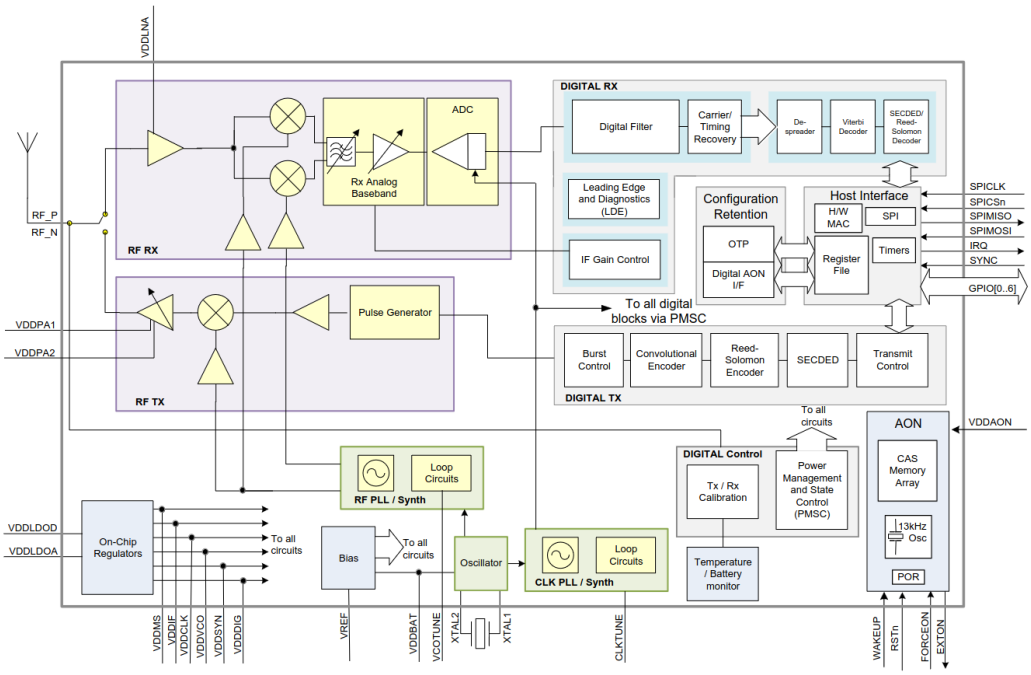


Figura 5.5: Arquitectura del transceptor DW1000[7].

La arquitectura del transceptor muestra que el mismo está diseñado para la implementación de aplicaciones de comunicaciones y localización activa en interiores. No hay disponibilidad de muestras crudas de la señal UWB recibida, sino que el usuario accede solamente a los datos transmitidos y adicionalmente a parámetros de transmisión/recepción. Estas limitaciones implican que no es posible explotar las características (o *features*) de las señales UWB ni la utilización del mismo como sensor monoestático.

[7]Imagen extraída de [14].

Humatics (ex *Time-domain*) P440

A diferencia del integrado de Decawave, la integración del *front-end* de Humatics, involucra solamente la etapa analógica y de digitalización. Esto implica que para el desarrollo de aplicaciones, Humatics proveyó en su momento un kit de desarrollo que incluye el *front-end* integrado *P400*, una FPGA y un microprocesador para el procesamiento de las señales, según se puede observar en la Figura 5.6.

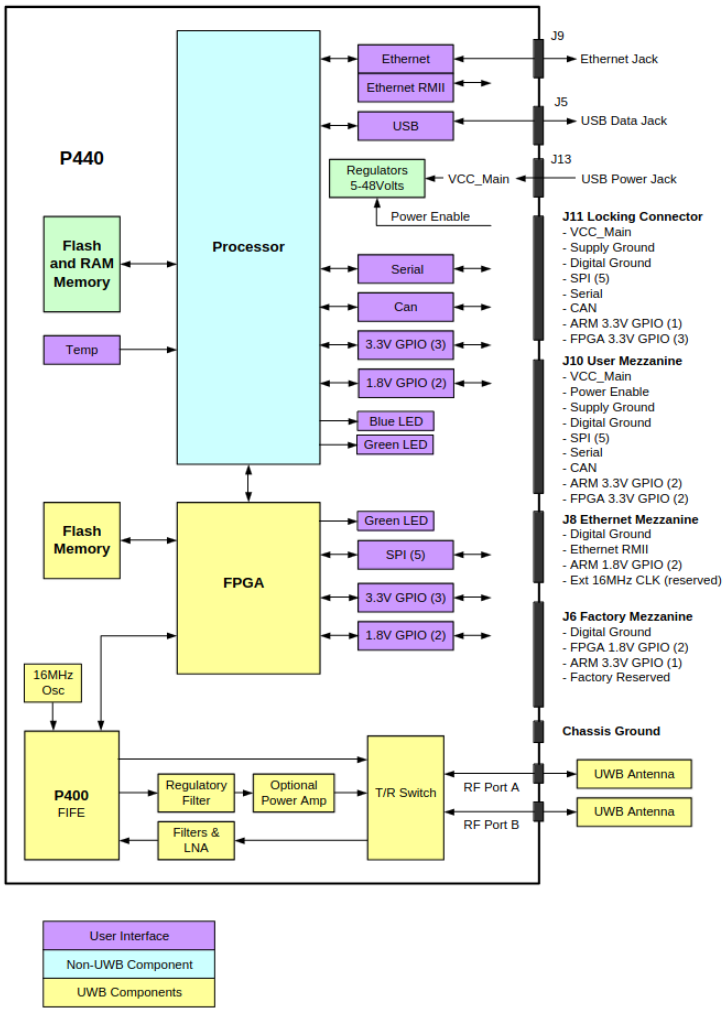


Figura 5.6: Arquitectura del transceptor P440[8].

El presente módulo ya no se encuentra disponible comercialmente ni se encuentran públicas las especificaciones del mismo. La información que se puede obtener se basa en el archivo de internet [82] del año 2022.

Xethru X4M06

El kit de desarrollo X4M06 de *Novelda* [83] fue puesto comercialmente a la venta en el año 2018. Es una implementación en base al chip *X4* desarrollado por la misma compañía. La plataforma esta compuesta por la integración del módulo *System On Package X4SIP02*

[8]Imagen extraída de [81].

[84], el controlador digital *XTMCU02* [85] y el módulo de antena *X4A04* [9]. La Figura 5.7 muestra el esquema de la plataforma.

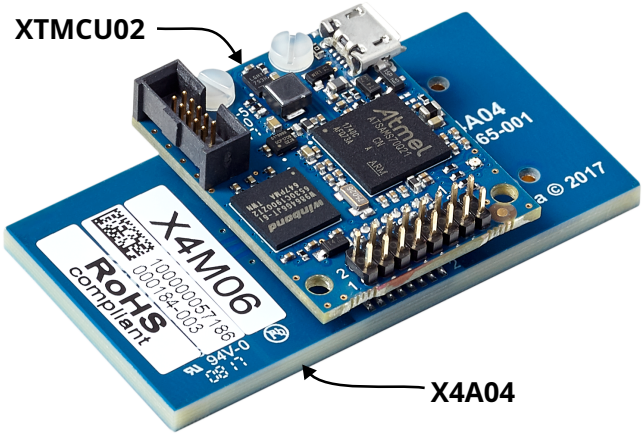


Figura 5.7: Esquema de la plataforma X4M06.

Dado el objetivo del presente trabajo (y dada la falta de documentación respecto del módulo de antenas *X4A04*), nos interesa analizar la arquitectura del front-end implementado en el chip X4, particularmente su esquema de muestreo y su interacción con la plataforma de procesamiento (típicamente una PC) a través del puerto *Universal Serial Bus* (USB) del módulo controlador *XTMCU02*. El transceptor integrado posee una arquitectura de conversión directa [12]. Utiliza un muestreo de un bit, en tiempo equivalente y acumulación a través de los sucesivos pulsos. La Figura 5.8 muestra un esquema de dicho muestreo.

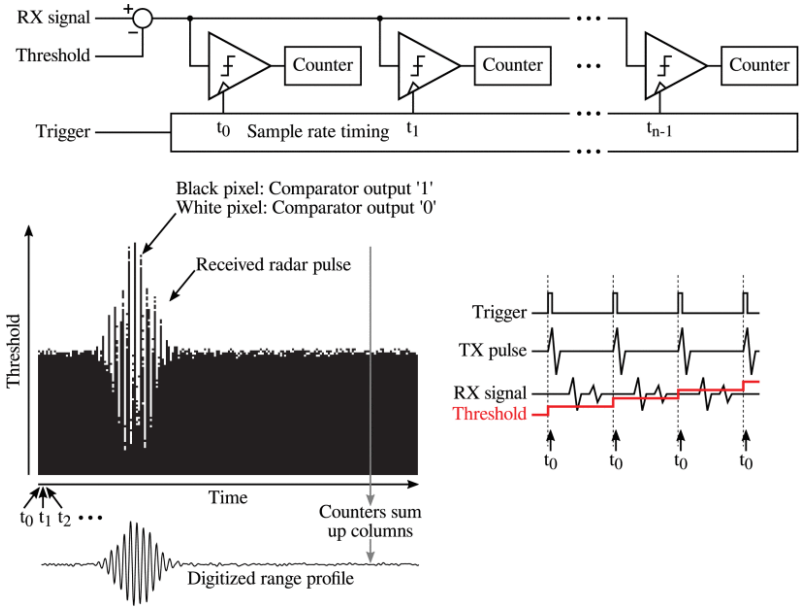


Figura 5.8: Esquema de muestreo del chip X4[10].

[9] Oficialmente Novelda no provee las hojas de datos para los módulos *X4M06* y *X4A04*. En su lugar el lector interesado podrá encontrar las de los módulos *X4M05* [86] y *X4A02* [87]

[10] Imagen extraída de [12]

	Por defecto	Máximo	Mínimo
Frecuencia central [GHz]	7,29	8,748	7,29
Ancho de pulso [ns]	2	-	-
PRF [MHz]	15,1875	40,5	Según configuración
Muestreo [GS/s]	23,328 @1 bit	-	-

Tabla 5.2: Especificaciones del kit de desarrollo X4M06.

El principio de funcionamiento del conversor se basa en un PLL con 12 salidas correspondientes a 12 fases de una señal de referencia de $1944MHz$. Cada una de estas salidas alimenta a la entrada de clock de un comparador diferente, produciendo de esta forma 12 muestras de 1 bit, separadas en un tiempo de $(12 \cdot 1944MHz)^{-1} = 42{,}86ps$, o equivalentemente una frecuencia de muestreo de $f_s = 23{,}328GSps$.

Cada comparador evalúa el nivel de la señal de entrada contra un umbral variable, el cuál se mueve dentro de un rango pre-establecido cada N pulsos transmitidos (configurable). El *frame* temporal de la señal de radar es dividido en 1356 *bins*, cada uno asociado a un acumulador, el cuál determina el valor de la muestra resultante. Con cada pulso transmitido cada acumulador es alimentado con un 1 o un 0 dependiendo de si la señal de entrada supera o no el umbral correspondiente. Los 12 comparadores son multiplexados y reutilizados 128 veces por frame, de forma de alimentar a los 1536 acumuladores. Al aumentar la cantidad de iteraciones, esto es, la cantidad de *frames* analizados antes de presentar las muestras, se logra un incremento de la resolución en bits de cada muestra, en detrimento de la tasa de *frames* por segundo (FPS). Del mismo modo se puede modificar la cantidad de pulsos transmitidos antes de modificar el nivel de comparación, lo que reduce el ruido o la dispersión de cada muestra, pero de la misma forma se produce una reducción de FPS. Cabe aclarar que se debe considerar al escenario como totalmente estacionario durante el tiempo inverso a FPS para que el principio de funcionamiento del conversor sea válido. El umbral de comparación es generado por un conversor digital analógico o *Digital-to-Analog Converter* (DAC) de 11 bits, del cual se pueden controlar su valor inicial y su valor final, limitando el rango de valores de un barrido, y por ende acelerando el tiempo de conversión. Cabe mencionar que si se estrecha demasiado ese rango, la señal de entrada puede caer por fuera, por lo que se produciría una saturación del valor de la muestra.

La Tabla 5.2 resume las prestaciones de esta plataforma.

Cabe aclarar que la comunicación con el módulo de procesamiento que utilice el usuario se realiza mediante una interfaz *UART* sobre USB. La velocidad de la misma es de 115200 baudios. Esta restricción limita la tasa de datos crudos que se pueden obtener. En el caso de requerir una lectura de la señal en banda base, el mismo módulo realiza una decimación de 8 sobre la señal, reduciendo la longitud del *frame* a 180 muestras. El módulo transmite lo datos en formato Q15 (es decir, 15 bits más un bit de signo), lo que implica una tasa máxima de transmisión continua de

$$FPS = \frac{115200}{2 \cdot 180 \cdot 16} = 20\ frames/s \tag{5.1}$$

siendo éste el límite máximo de datos para aplicaciones en tiempo real. En caso de superar este valor, el buffer del dispositivo se llenará produciendo una pérdida de datos.

5.4. El problema de adquisición de señales experimentales

Del estudio del estado del arte presentado en este capítulo se identificó una carencia en el acceso a señales UWB crudas para su procesamiento. El avance de la investigación de nuevas técnicas de procesamiento para extracción de características de dichas señales es fuertemente dependiente del acceso a los datos de forma de poder planificar y realizar experimentos adecuados en función del problema a resolver. Los trabajos analizados proponen soluciones para aplicaciones específicas, por ejemplo localización, o no están orientados a la reproducibilidad, ya sea por ser soluciones de silicio a medida o por tener componentes de muy alto costo. Finalmente, el trabajo de Colli-Vignarelli, si bien orientado a la reproducibilidad e implementación con componentes comerciales, no provee una infraestructura completa (incluyendo software) para la utilización de la plataforma como banco de pruebas para algoritmos de procesamiento. Durante el desarrollo de la presente tesis, el Centro de Simulación Computacional (CSC) tuvo la posibilidad de adquirir una plataforma Xethru de Novelda, discontinuada al momento de la escritura de este documento. Para la misma, el tesista desarrolló un software de procesamiento y visualización en tiempo real (ver Sección 7.3.2.1), pero dado que no es posible la actual adquisición de la misma, el aporte a la comunidad científica es limitado en este aspecto.

Por lo mencionado, en el capítulo siguiente se detalla el desarrollo realizado de una plataforma de adquisición de señales UWB con capacidades de procesamiento.

Capítulo 6

Plataforma experimental UWB-IR basada en *System-on-Chip*

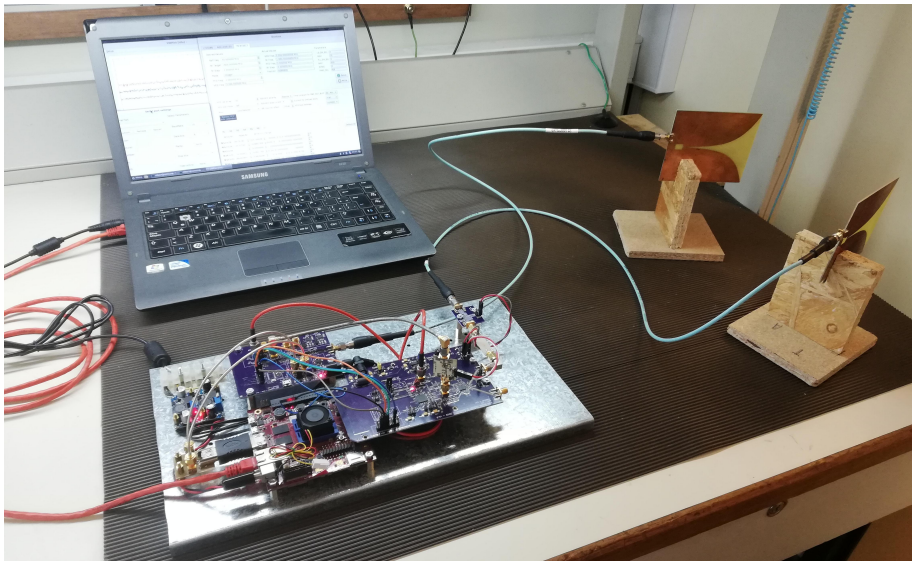


Figura 6.1: Plataforma experimental implementada.

RESUMEN: En el presente capítulo se presenta en profundidad el desarrollo completo de una plataforma digital de adquisición y procesamiento de señales UWB-IR. Se describe la arquitectura de hardware tanto circuital como de procesamiento digital, incluyendo el sistema de adquisición. Adicionalmente, se introduce el software de control, visualización y grabación de datos. Finalmente, se presentan los resultados de la implementación en cuanto a recursos de hardware y señales adquiridas.

6.1. Introducción

En el Capítulo 5 se presentó el ecosistema de implementaciones y estado del arte de las soluciones tecnológicas disponibles. Cómo fue explicitado, existe un déficit en cuanto a la disponibilidad y/o accesibilidad a plataformas experimentales de adquisición

y procesamiento de señales UWB. Con el objetivo de poder disponer de datos y señales UWB se realizó el desarrollo documentado en este capítulo.

Si bien se partió del conocimiento generado por los trabajos citados en el capítulo anterior, la disponibilidad local de componentes determinó en gran parte la arquitectura y las técnicas utilizadas para la generación y digitalización de datos. En particular, en lugar de utilizar circuitos analógicos específicos, se buscó integrar la mayor cantidad de funcionalidades en los circuitos digitales, especialmente para aprovechar las capacidades de las FPGA o SoC, y su versatilidad. Puntualmente, la generación del pulso se realizó haciendo uso de la configurabilidad y velocidad de los puertos de entrada/salida de estos dispositivos. Esto adicionalmente permite la configuración de parámetros como el ancho de pulso, la impedancia de salida, si se requiere un pulso diferencial o no, etc.

Por otro lado, la digitalización debería ser tal que permitiera diferentes tasas de muestreo, pero con componentes disponibles y de costo razonable. Para ello, se diseñó e implementó un sistema en tiempo equivalente que permite configurar la tasa de muestreo de forma flexible con un ADC de gama media. La única limitación tecnológica de este sistema es el ancho de banda analógico del conversor seleccionado. En la Sección 6.3 se profundizará sobre este tema.

Otro limitante identificado en las soluciones analizadas es la tasa de transferencia de datos al sistema de procesamiento (por ejemplo, computadora personal (PC)). En ese sentido, el diseño presentado introduce dos ventajas importantes:

- Acerca capacidad de procesamiento a la fuente de información. Esto se logra teniendo una FPGA y un procesador de prestaciones medias (pero dedicado) en el mismo hardware, interactuando directamente con el conversor analógico-digital.

- Implementa un enlace de comunicación de alta velocidad (GbE) capaz de transferir datos a una tasa de hasta 1 Gbps. Lo que permite explotar la alta velocidad de adquisición del dispositivo.

Se utilizó como base para el desarrollo, la Computadora Industrial Abierta Argentina para aplicaciones de Alto Costo Computacional (CIAA-ACC)[88], debido a su naturaleza open-source y su diseño local por parte del INTI. A la misma se le adosaron dos placas hijas: una para la modulación y ganancia del pulso transmitido, y otra para la demodulación y digitalización de las señales. El esquema general de la plataforma se muestra en la Figura 6.2. La CIAA-ACC dispone de un SoC *Zynq 7030* de *Xilinx*, el cual está integrado por un procesador de doble núcleo *ARM Cortex A9* y un área de FPGA del tipo *Kintex*, llamadas *Processing System* (PS) y *Programmable Logic* (PL) respectivamente. El PS se utiliza principalmente para la configuración de los periféricos a través de un bus *Serial Peripheral Interface* (SPI), el manejo de la memoria RAM principal, y la conexión *User Datagram Protocol* (UDP) con la PC, mientras que el PL se utiliza para pre-procesamiento de la señal digitalizada según se detalla en la Sección 6.3.

Teniendo en cuenta el estado del arte y la disponibilidad de componentes *off-the-shelf* se propusieron los requerimientos de diseño enumerados en la Tabla 6.1.

En el resto del capítulo se presentan los detalles etapa por etapa de la plataforma. Se describen las etapas analógicas por completitud y se detalla el diseño y desarrollo del hardware digital, desde el circuito impreso o *Printed Circuit Board* (PCB) hasta el software de control y visualización de datos.

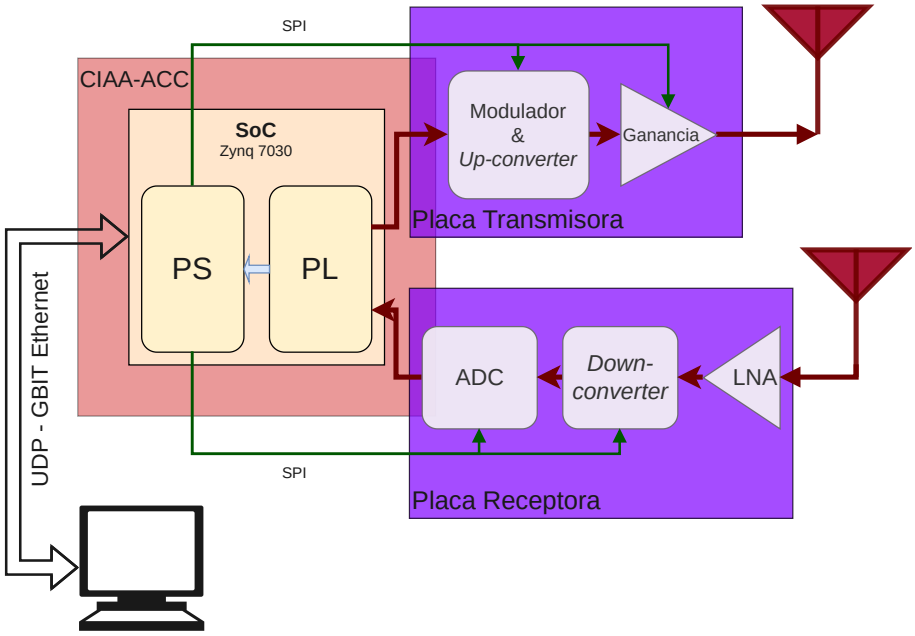


Figura 6.2: Esquema general de la plataforma.

Tabla 6.1: Requerimientos de diseño de la plataforma.

Requerimiento	Valor	Descripción
f_{RF}	Hasta 4,5GHz	Frecuencia de portadora
BW	Mayor a 500MHz	Ancho de banda de señal
$f_{muestreo}$	Mayor a 2$GSps$	Frecuencia de muestreo
N_{bits}	Mayor a 8 bits	Resolución de conversión
Modulación	en cuadratura	canales I + Q
PRF	Mayor a 30 Frames por segundo	Frecuencia de repetición de pulsos

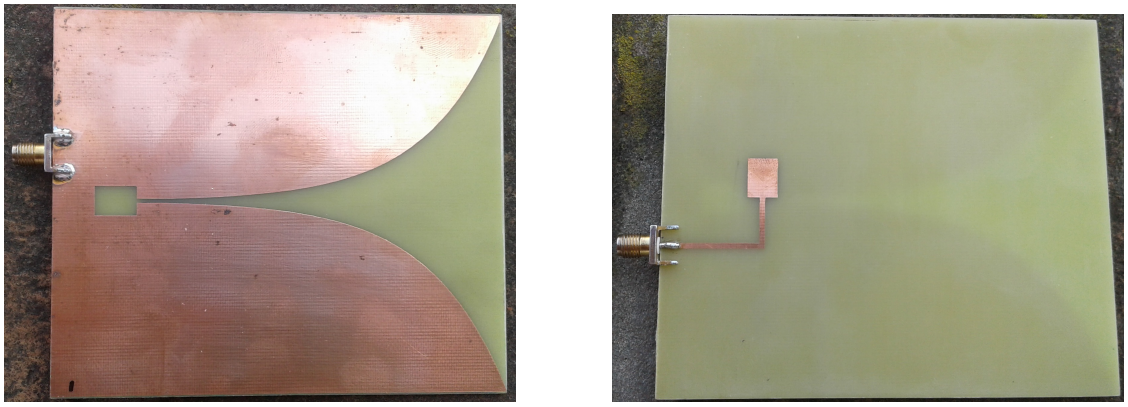
6.2. *Front-end* de RF

Si bien el desarrollo de la etapa de Radio Frecuencia (RF) no forma parte del presente trabajo (particularmente el *Analog Front-End* (AFE)), se considera importante describir el diseño de la misma para comprender el funcionamiento de la plataforma completa. En [89],[90] el lector puede ahondar en detalles del diseño de radiofrecuencias. En las próximas secciones se presentan las antenas utilizadas y las etapas de transmisión y recepción de RF.

6.2.1. Antenas

La antena utilizada en la plataforma presenta requerimientos exigentes respecto de la dispersión del pulso transmitido. Para ello se utilizó una antena Vivaldi según se muestra en la Figura 6.3, cuyo diagrama de radiación fue presentado en la Figura 3.2a del Capítulo 3.

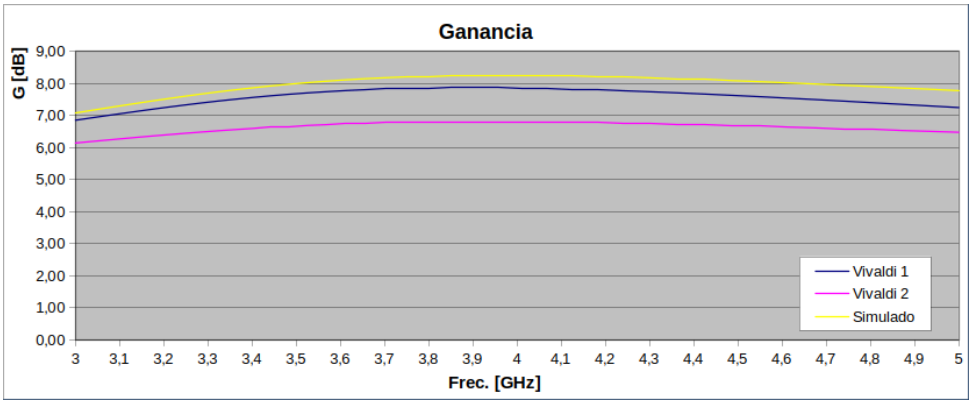
El diseño de la antena se verificó mediante simulaciones en *ADS* que luego fueron contrastadas con mediciones realizadas en la cámara semi-anecóica del INTI. Se midieron



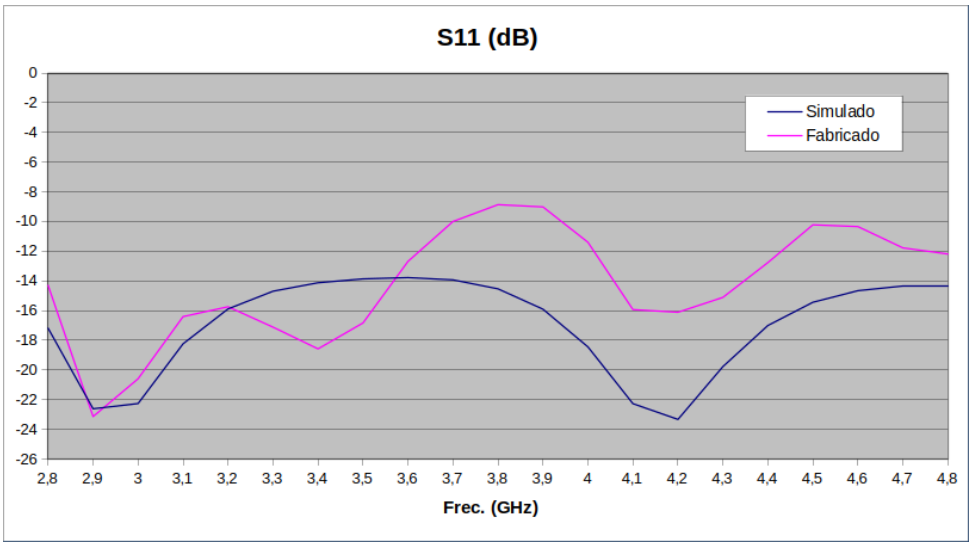
(a) Vista frontal. (b) Vista posterior.

Figura 6.3: Antena Vivaldi utilizada en la plataforma.

ambas antenas y el resultado de la ganancia y del parámetro S11 se puede observar en las Figuras 6.4a y 6.4b respectivamente.



(a) Ganancia simulada vs medida de dos prototipos (“Vivaldi 1” y “Vivaldi 2”).



(b) Parámetro S11 simulado vs medido.

Figura 6.4: Antena Vivaldi utilizada en la plataforma.

Se pudo verificar que las antenas fabricadas resultaron tener un comportamiento similar al esperado en base a las simulaciones. Particularmente, la respuesta en frecuencia de la ganancia mostró la misma distribución que la simulada teniendo una dispersión constante en toda la banda, más notoria para el prototipo 2. Por otro lado, el parámetro S_{11} presenta las dispersiones esperadas, atribuidas principalmente a la unión mecánica con el conector de alimentación SMA.

6.2.2. Transmisor RF

El transmisor de RF fue diseñado para modular directamente el pulso de banda base producido por la FPGA. Se utilizó el integrado TRF372017[91] como mezclador debido a su versatilidad y sus prestaciones. Las especificaciones del mismo son compatibles con un sistema UWB-IR:

- Oscilador local hasta 4,8 GHz.
- Ancho de banda analógico de 1 GHz.
- PLL integrado (elimina la necesidad de un sintetizador externo).
- Corrientes I y Q ajustables para eliminar el *feedthrough* de portadora.

Se le prestó especial atención al jitter del oscilador local de forma de obtener un ruido de fase bajo. Para ello se utilizó un oscilador de la serie TB de *Connor Winfield* de 80MHz con un jitter de 0,5ps RMS y una estabilidad en frecuencia de 1ppm. El ajuste final de potencia transmitida se realiza mediante una etapa de ganancia de 16dB implementada a través de un amplificador integrado ADL5545[92] y un atenuador PE43712[93] controlado digitalmente por SPI. El esquema completo del transmisor se muestra en la Figura 6.5.

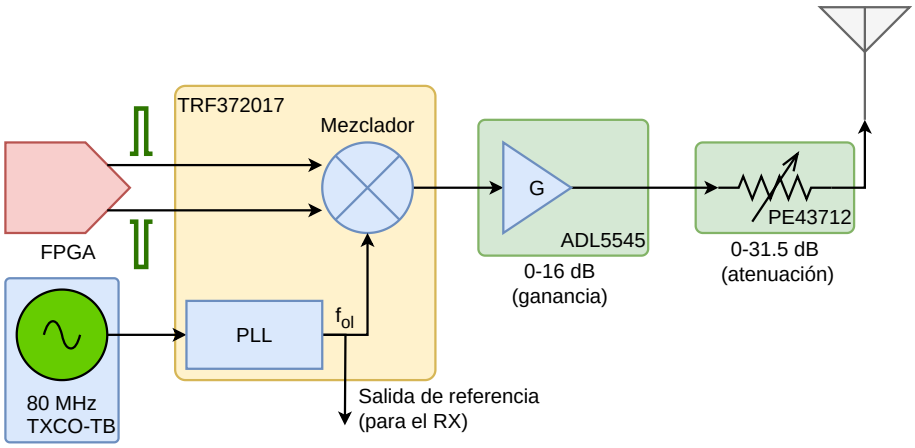


Figura 6.5: Esquema del transmisor.

El PCB del mismo fue diseñado y fabricado sobre un un sustrato FR408-4 debido a sus características dieléctricas a las frecuencias utilizadas. En la Sección 6.2.3.3 se detallan algunas consideraciones al momento de diseño. La Figura 6.6 muestra la implementación del transmisor.

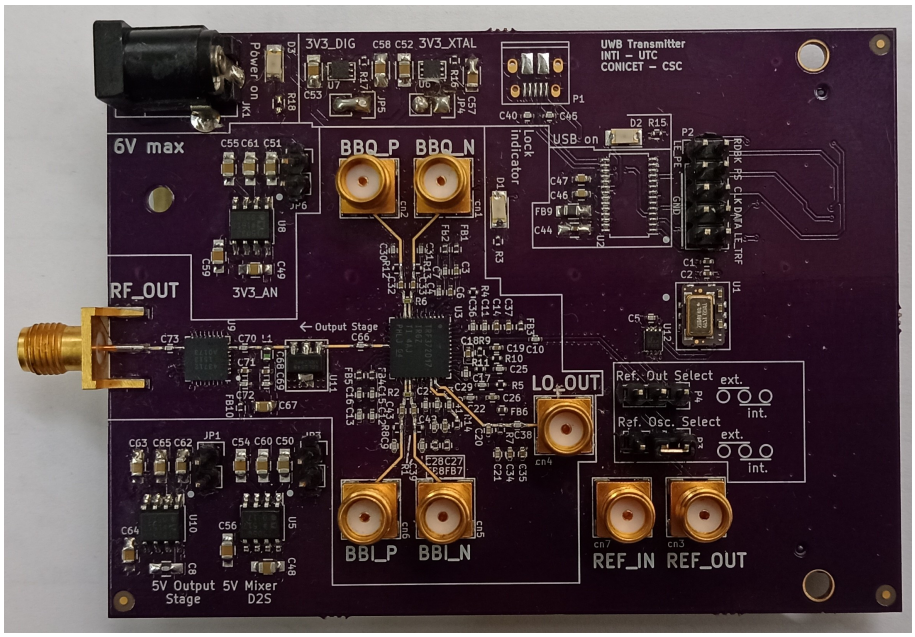


Figura 6.6: Transmisor completo implementado.

6.2.3. Receptor RF

El módulo receptor cuenta con dos etapas bien definidas, el AFE y el *Downconverter Sampler-Stage* (DSS). El mismo se integró en una única placa hija de forma de introducir la menor cantidad posible de desadaptaciones de impedancia, y por consiguiente pérdidas, en el camino de RF. La excepción a esto son los módulos de LNA y el filtro pasa-banda de entrada, debido a que los requerimientos eléctricos de este último imponen la implementación del filtro en un sustrato diferente (*RT/Duroid*). El esquema general del receptor se presenta en la Figura 6.7.

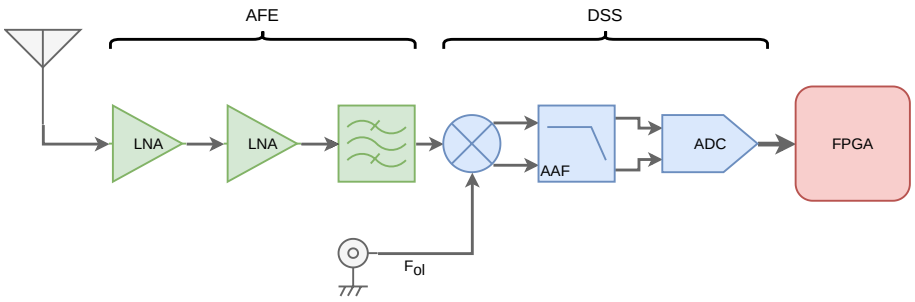


Figura 6.7: Esquema del receptor.

6.2.3.1. AFE

La primera etapa del AFE es un LNA de banda ultra ancha, de forma de aumentar la sensibilidad del receptor. Los requerimientos técnicos se pueden encontrar en [89]. El mismo se implementó con un transistor HBT BFP840FESD debido a su figura de ruido por debajo de $1dB$ en el rango de frecuencias de interés. Dada la limitada ganancia del LNA, fue necesario utilizar dos etapas en cascada. En condiciones normales de operación esta configuración no se ve impactada por el punto de compresión del mismo. El filtro

pasa-banda es del tipo Butterworth *hairpin* construido sobre sustrato *RT/Duroid 6006* de 1,9*mm* de espesor. La implementación de las dos primeras etapas del AFE se muestran en la Figura 6.8.

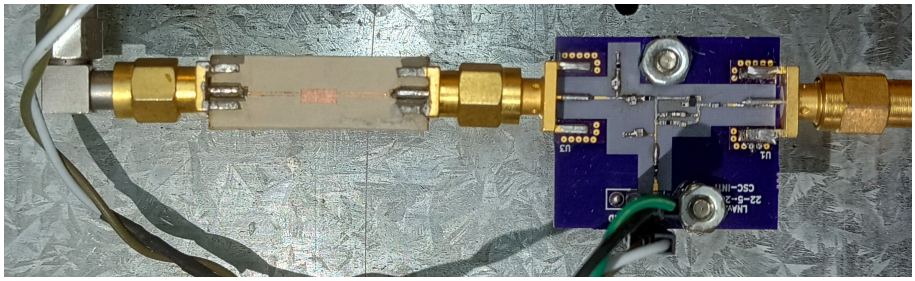


Figura 6.8: LNA y filtro pasa-banda del AFE.

6.2.3.2. DSS

Si bien las etapas de RF quedan por fuera del objetivo de este trabajo como fue mencionado, la etapa DSS por consistir mayormente en señales de banda base o analíticas, fue diseñada e implementada por el tesista. El hardware de esta etapa de conversión a banda base y muestreo está basado en dos componentes principales, el demodulador en cuadratura LTC5586[94] y el conversor analógico-digital ADC12DS105[95].

El esquema de demodulación mediante conversión directa se realiza mediante el integrado LTC5586 que posee un ancho de banda de $1GHz$ $(-1dB)$ en el rango de frecuencias de 300 a $6000MHz$. Dicho integrado posee varias etapas de ganancia internas, eliminando la necesidad de utilizar circuitos externos adicionales en el camino de frecuencia instantánea o *Instantaneous Frequency* (IF), y configuración vía SPI de parámetros que permiten maximizar el rechazo de frecuencias imagen y/o pérdida de portadora. La demodulación se realiza en forma coherente, alimentando al mezclador con la portadora generada en el transmisor. El filtro anti-alias (AAF) cumple el tradicional propósito de filtrar $f_{RF} + f_{LO}$ y otro productos del mezclado y su implementación se basó en el propuesto por el fabricante [94]. El voltaje de modo común del mezclador se alimenta directamente a la entrada de referencia del ADC, permitiendo un seguimiento directo de las variaciones de modo común por parte del ADC.

El ADC ADC12DS105 es un conversor de $12 + 12$ bits (I+Q), con una frecuencia máxima de muestreo máxima de $105MHz$. La virtud de este conversor, y que determinó su elección en el diseño, es que dispone de un ancho de banda analógico de $1GHz$, por lo que permite, mediante muestreo en tiempo equivalente (TE), muestrear señales UWB. La comunicación con la FPGA se realiza mediante señales serie LVDS.

La implementación completa de la placa hija del receptor se muestra en la Figura 6.9.

6.2.3.3. Consideraciones de diseño del PCB

El gran ancho de banda de las señales introduce complejidades en el diseño que no necesariamente aparecen en diseños de elevada frecuencia de portadora pero ancho de banda relativo angosto. Especialmente, se deben a la dispersión de las formas de onda, que en teoría abarcan desde 0 a $BW\,Hz$ siendo BW el ancho de banda configurado para UWB. Para minimizar dichos efectos, fue necesario prestar extrema atención al diseño electromagnético.

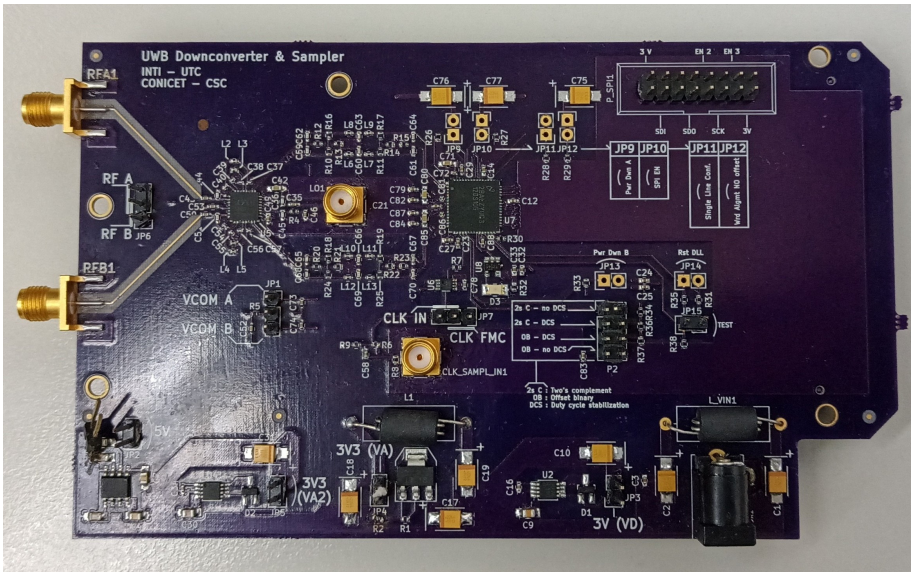


Figura 6.9: Implementación del receptor y digitalizador UWB.

Teniendo en cuenta lo mencionado[1], para la implementación del PCB se utilizaron 4 capas sobre un sustrato FR-408[96] con la siguiente distribución:

Capa 1 - Señales de RF.

Capa 2 - GND.

Capa 3 - Alimentaciones.

Capa 4 - Señales digitales de alta velocidad.

La misma permite:

- Poder determinar la impedancia característica de las pistas de RF con un modelo electromagnético adecuado. Ya que se puede asumir la pista conductora sobre un plano conductor infinito (Capa 1 - Capa 2).
- Generar una capacidad parásita de desacople entre GND y las diferentes alimentaciones de manera de agregar un filtrado adicional al ruido de alta frecuencia que pueda existir en las mismas (Capa 2 - Capa 3).
- Usar los planos de alimentación como referencias para las líneas de transmisión digital de alta velocidad ya que las mismas son líneas diferenciales y no requieren una referencia tan ideal como en el caso de la RF (Capa 3 - Capa 4).
- Aislar los acoplamientos mutuos entre RF y señales digitales (Capa 1 - Capa 4).

La primera etapa del receptor involucra señales de RF ya que aún no se realizó la demodulación. Para explotar al máximo las capacidades del mezclador seleccionado, se debe asegurar integridad de señal hasta los $6GHz$ con un ancho de banda de al menos $1GHz$. Para ello se utilizó el modelo de línea de transmisión microstrip, y se eliminó la máscara

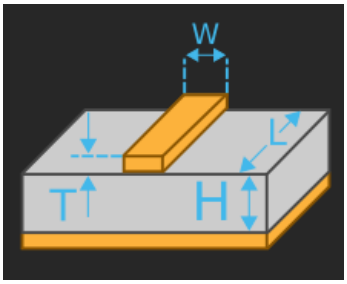
[1] y teniendo también siempre presente las limitaciones de costo y accesibilidad de los componentes.

anti-soldante del diseño, de forma de no introducir distorsiones debido a la dispersión de la impedancia característica. Las dimensiones de línea *microstrip* fueron calculados por la herramienta de diseño del PCB (*KiCad*) en base a los parámetros informados tanto por el fabricante de la placa (*OSHPark*)[97] cómo por el fabricante del sustrato (*Isola*)[96]. Los mismos se resumen en la Tabla 6.2.

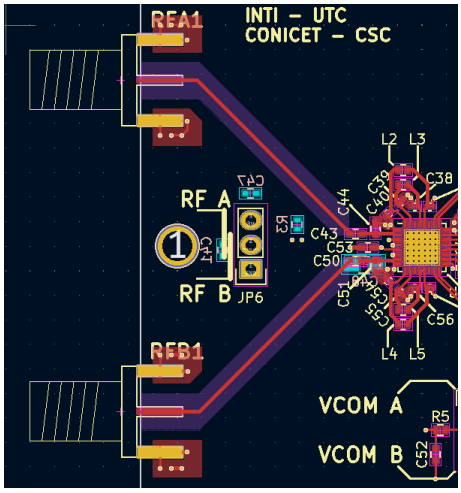
Tabla 6.2: Parámetros del proceso de fabricación del PCB.

Parámetro	Valor	Unidad	Descripción
ε_r	3.66	F/m	Constante dieléctrica FR408 @$5GHz$
$tan\delta$	0.002	-	Pérdidas dieléctricas FR408 @$5GHz$
H	0.127	mm	Altura de sustrato (entre capas 1 y 2)
T	0.017	mm	Espesor de cobre (capa 1)

El esquema del modelo y el diseño resultante de la etapa de entrada en RF se muestran en las figuras 6.10a y 6.10b respectivamente.



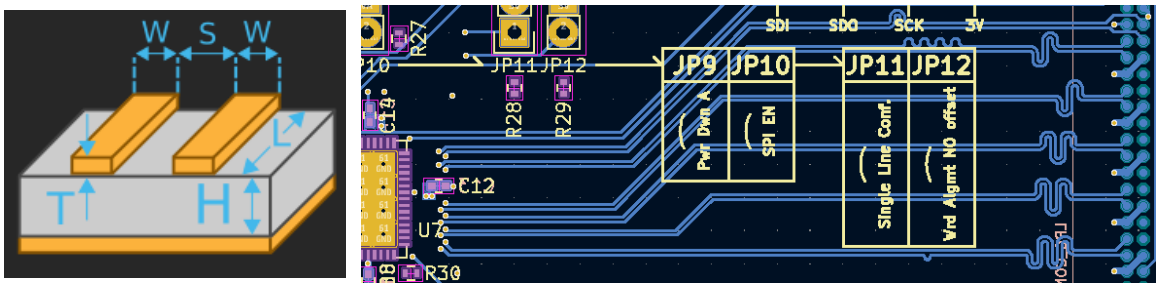
(a) Esquema del modelo de línea microstrip.



(b) Diseño del PCB resultante.

Figura 6.10: Diseño de la etapa de entrada en RF.

Para las pistas digitales de alta velocidad (capa 4) se modificó el modelo de línea de transmisión a *microstrip* acoplada, dado que se utiliza el estándar LVDS. Adicionalmente, se debió configurar la herramienta de diseño para ecualizar la longitud de los diferentes canales, de forma que los datos lleguen sincronizados a la FPGA ya que la tasa de transmisión para cada línea puede llegar a $625Mbps$. La conexión con la placa madre que contiene al SoC se realizó mediante un conector FMC-HPC, el cual también se encuentra ecualizado en dicha placa, por lo que el ajuste de longitud se debe considerar particularmente en el trayecto desde el ADC hasta el conector. El modelo modificado de línea y la implementación de la ecualización de pistas de alta velocidad se muestran en las Figuras 6.11a y 6.11b, respectivamente.



(a) Esquema del modelo de línea microstrip acoplada.

(b) Diseño del PCB resultante.

Figura 6.11: Diseño de la comunicación de alta velocidad con la FPGA.

6.3. Procesamiento digital en hardware

La versatilidad ofrecida por el SoC permite la integración de muchas etapas dentro de un sólo dispositivo. Esto se consideró como una premisa de diseño para la arquitectura implementada como ya fue mencionado. La Figura 6.12[2] describe en forma general la arquitectura global del sistema implementado dentro del SoC.

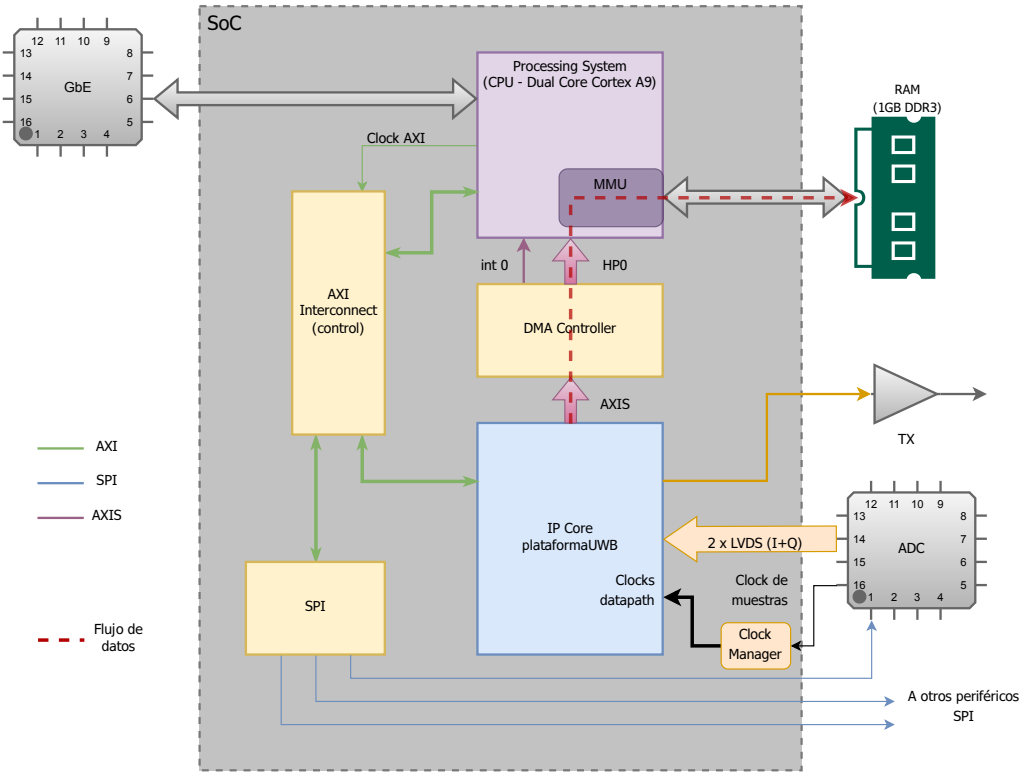


Figura 6.12: Arquitectura implementada en el SoC.

El PS interactúa con sus periféricos *on-chip* a través de un bus AXI. El mismo se utiliza para control y configuración de los mismos. El PS también genera el *clock* general para dicho bus.

[2]Para ver un mayor detalle de la implementación referirse al apéndice A.1

Todos los clocks para el *datapath* se generan a partir del *clock* de referencia provisto por el ADC.

La arquitectura fue desarrollada en forma de *IP Core* dentro de la PL que actúa como periférico del PS. La comunicación entre ambos módulos se realiza a través de dos canales. El primero es un bus AXI para configuración y control, dónde al igual que el resto de los periféricos conectados a este bus, los registros de configuración son expuestos al PS como dispositivos mapeados a memoria. El segundo es un bus del tipo *AXI Stream*, o AXIS por sus siglas, para transferir los datos de las muestras pre-procesadas. Éste último se diferencia respecto del AXI tradicional, debido a que está diseñado para alta tasa de datos. No utiliza cómo medio de comunicación un mapeo a memoria, sino que se conecta directamente a un dispositivo de acceso directo a memoria o *Direct Memory Access* (DMA) que intercambia información con el PS a través del puerto HP0 (*High Performance port 0*). Éste tiene acceso directo a la Unidad de gestión de memoria o *Memory Management Unit* (MMU) para poder escribir directamente en la RAM sin requerir recursos del procesador. Cuando una transferencia se completa, el controlador DMA emite una interrupción al procesador. La rutina de atención de interrupciones correspondiente fue programada para leer una cantidad de datos definida de la posición de memoria particular (configurada por software al momento de inicialización del dispositivo). Finalmente, el software corriendo en el PS puede hacer uso de los datos como si estos fueran un arreglo en memoria (por ejemplo, en `C` se manipulan como `arrays` o punteros). Este diseño de arquitectura permite que el procesador pueda hacer uso eficiente del tiempo de cómputo ya que no necesita ejecutar movimientos de memoria, y cada vez que hay datos disponibles se atienden vía interrupción.

Internamente, el *IP Core* "`plataformaUWB`" se diseñó con una topología tipo *pipeline* de forma de poder trabajar con alta frecuencia de reloj y alto *throughput*. El mismo fue programado en lenguaje descritor de hardware (HDL), particularmente en *VHSIC Hardware Description Language* (VHDL). La cadena de procesamiento se muestra en la Figura 6.13 y consta de las siguientes etapas que se detallarán en las sub-secciones a continuación:

- Generación de pulsos.
- Interfaz de datos con el ADC.
- Interleaving de muestras para el esquema de muestreo en tiempo equivalente o *Equivalent Time Sampling* (ETS).
- Control de *throughput*.
- Promediado de realizaciones o frames.
- Lógica de control.

6.3.1. Generación de pulsos

El pulso transmitido se genera directamente en banda base a través del área de PL del SoC. El método de generación se basa en un contador programable alimentado con una señal de reloj (o *clock*) configurada a una frecuencia acorde al ancho de banda deseado. Adicionalmente, el número de cuenta máximo se puede programar en tiempo de ejecución

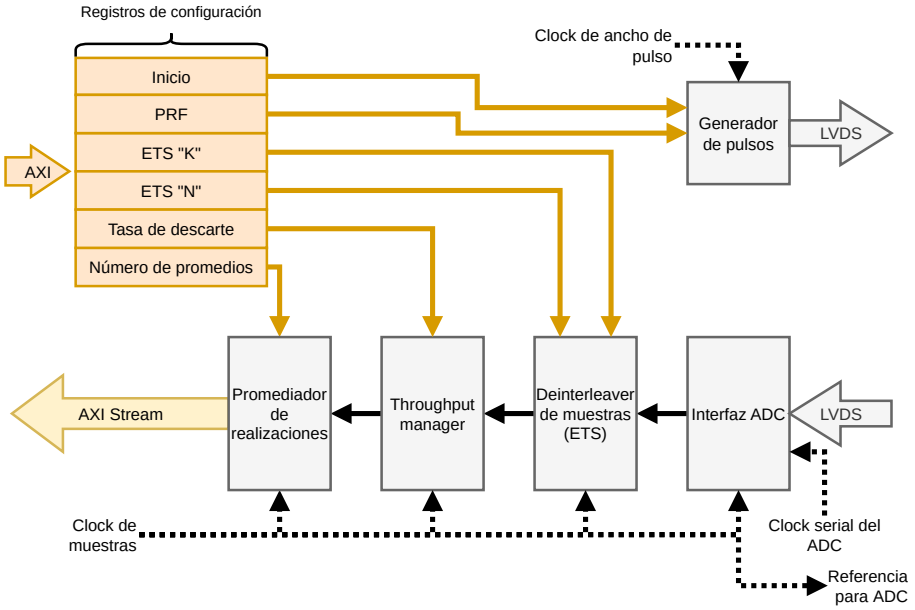


Figura 6.13: Esquema de la cadena de procesamiento del *IP Core* desarrollado en la PL.

para fijar la PRF. El *flag* de cuenta máxima se alimenta directamente al buffer de salida según se muestra en el esquema de la Figura 6.14.

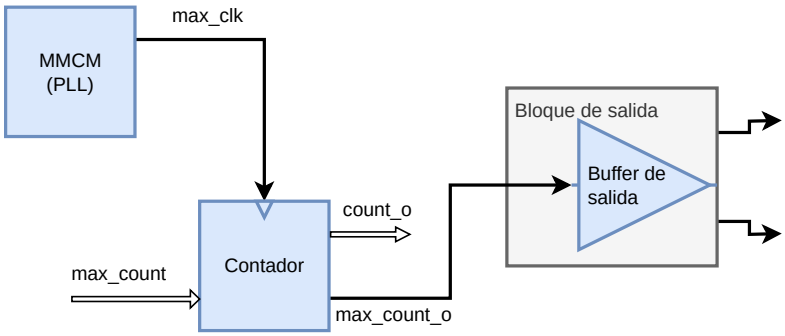


Figura 6.14: Esquema del generador de pulsos.

El bloque de salida del dispositivo permite controlar la terminación (diferencial o común) y la impedancia de la salida a través de un control llamado Impedancia Controlada Digitalmente o *Digitally Controlled Impedance* (DCI). Un esquema simplificado de este *buffer* se puede encontrar en la Figura 6.15[3]. Es posible también definir la tensión de salida por banco de pines, por lo que para esta aplicación en particular, se seleccionó una salida diferencial de $2{,}5V$ con una $Z_o = 100\Omega$, compatible con el estándar LVDS25.

6.3.2. Conversión analógico-digital

La primera etapa resuelta en hardware programable (FPGA) del *datapath* de recepción es la adaptación del formato de las muestras entregadas por el ADC en forma serial bit a bit en dos líneas por canal I-Q. Este bloque presenta dos canales de salida de 12 bits cada

[3]Imagen extraída de [98]

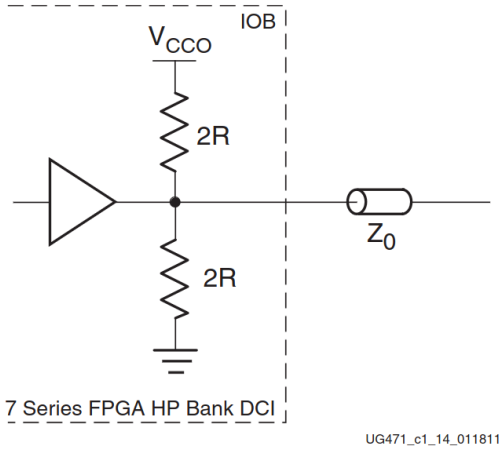


Figura 6.15: Esquema del buffer de salida del SoC con DCI.

uno, donde por cada pulso de clock de salida, se presenta una muestra por canal (I+Q). Adicionalmente, genera adaptación de dominios de clock a través de FIFOs. La interfaz de este módulo se presenta en la Figura 6.16.

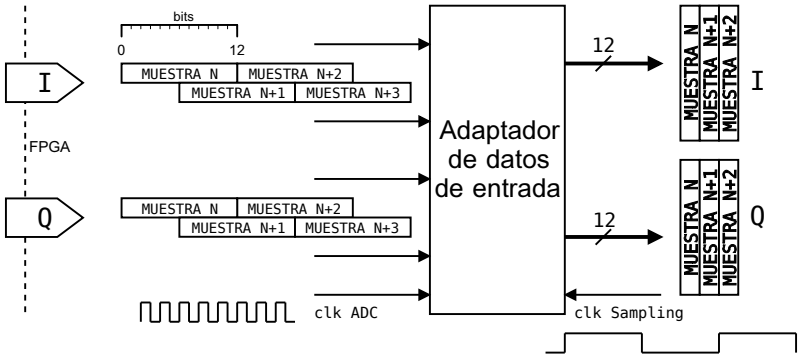


Figura 6.16: Esquema del bloque adaptador de formato de muestras.

Teniendo las muestras reconstruidas y sincronizadas entre ambos canales, el paso siguiente es el reordenamiento de las mismas para el armado del *frame*, según el método de muestreo presentado en la sección siguiente.

6.3.2.1. Esquema de muestreo

El esquema de ETS propuesto en el presente trabajo se basa en muestreo coherente. La tasa de muestreo obtenida es configurable mediante la selección de dos parámetros:

- K: Cantidad de muestras deseadas por *frame*.
- N: Cantidad de pulsos transmitidos por ventana de observación ó numéricamente coincidente con la relación entera entre la frecuencia de muestreo aparente y la real.

La idea detrás de la metodología propuesta se basa en que si K/N es una fracción irreducible, entonces luego de N pulsos transmitidos (o equivalentemente *frames*), se habrán muestreado K muestras (particularmente con una relación de K/N por *frame*). Bajo la premisa que el escenario permanece estático durante la ventana de observación de N

pulsos, se pueden reordenar las muestras de forma tal que se obtenga una secuencia de K muestras consecutivas de un *frame* equivalente.

6.3.3. *De-interleaving* de muestras

Esta estrategia de muestreo se implementó en hardware mediante un *ping-pong* buffer. Esto es, un buffer dual de acceso alternado según se muestra en la Figura 6.17. Este tipo de *buffer* está compuesto por dos memorias de doble puerto de acceso. La lectura/escritura a ambas memorias se da de forma excluyente, es decir, mientras se escribe en la memoria A, se lee de la memoria B y viceversa. Para la aplicación particular al esquema de ETS mencionado, el control de direcciones se da con dos lógicas distintas para la escritura (entrada) y para la lectura (salida).

Las direcciones de escritura se calculan según la siguiente ecuación:

$$Addr_w = (i.N) \text{ mód } K \qquad \text{con} \quad i = 0,1,2,...,K-1 \tag{6.1}$$

donde i se incrementa con cada ciclo de clock. Secuencialmente el avance de direcciones se da de a N posiciones, con una lógica modular a K. Dado que, como se mencionó anteriormente, K/N es una fracción irreducible cuándo $i = K-1$ se recorrieron todas las direcciones de una memoria. En el ciclo siguiente se comienza a escribir en la otra memoria, con la misma lógica. Paralelamente en la memoria que no está siendo escrita, se leen todas las direcciones, con una lógica de incremento de a una posición.

Este proceso permite que las muestras sean reordenadas produciendo un frame equivalente muestreado a $N.fs_R$ dónde fs_R es la frecuencia de muestreo real del ADC.

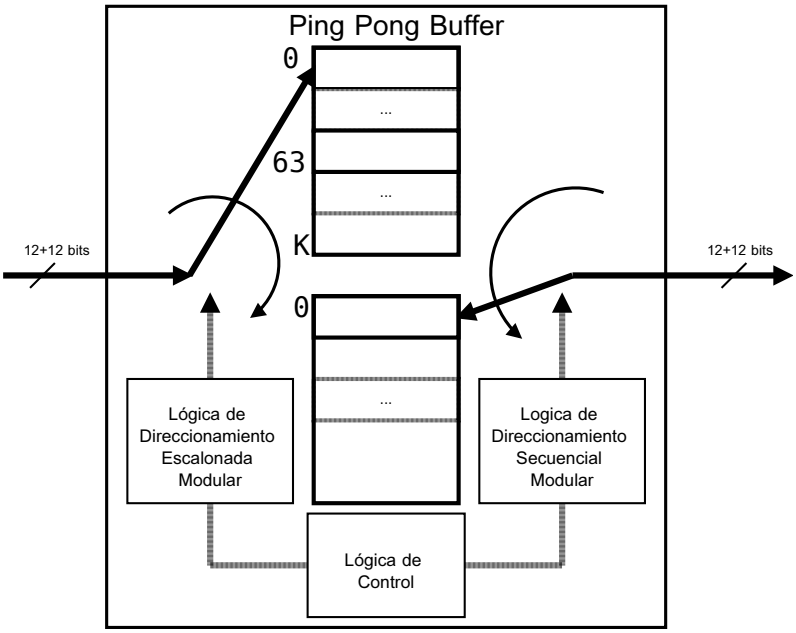


Figura 6.17: Esquema del bloque de reordenamiento o desentrelazado de muestras.

6.3.4. Promediador de realizaciones

En este punto de la cadena, la tasa de datos aún es muy elevada, particularmente de $1{,}92Gbps$, dado que tenemos $(12+12)\,\text{bits} \times 80MSps$. Por este motivo se implementó

un control de tasa mediante dos etapas: un promediador de realizaciones que permite una mejora en la SNR mediante la fusión de varios *frames* y un descartador de frames o paquetes (*packet dropper*).

Este último se encarga simplemente de descartar un paquete (en este contexto un paquete equivale a un frame) cada P paquetes enviados. Se implementó simplemente con un contador (cuenta frames o paquetes) y una compuerta `AND` con la señal de *enable* de la salida del promediador. Esto permite que cuándo se llega a la cuenta máxima configurada, un paquete sea descartado mediante la deshabilitación de la señal de salida válida (o *data valid*). El parámetro P se puede configurar en tiempo de ejecución y de esta forma, permite controlar la tasa de datos de salida del dispositivo.

Para la implementación del promediador de realizaciones se observó que si se ordenan las muestras en memoria según la Figura 6.18 es posible utilizar una nueva instancia del *ping-pong buffer* para realizar el promediado de forma eficiente.

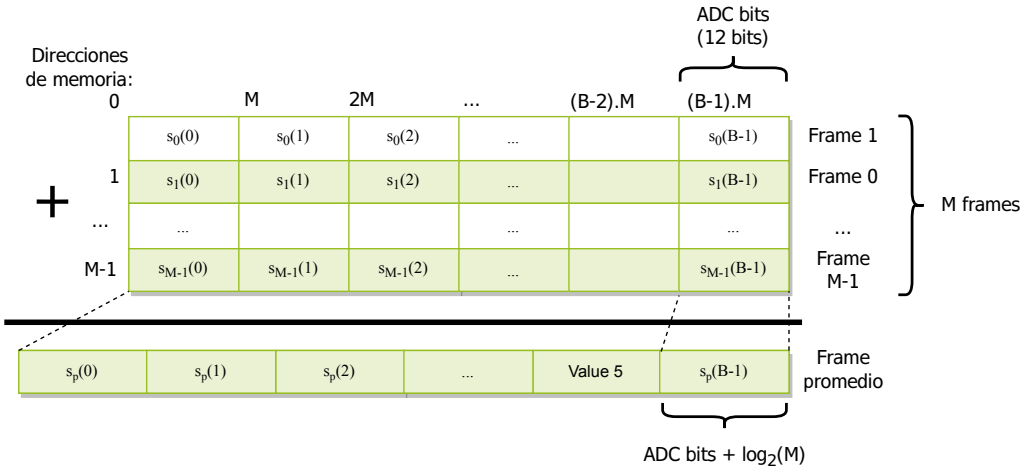


Figura 6.18: Esquema del bloque promediador de realizaciones.

La escritura de muestras en la memoria se realiza de forma escalonada de a M posiciones, siendo M la cantidad de *frames* a promediar. Luego, se lee en forma secuencial y contigua (por columnas), y esos datos se alimentan a un acumulador que cada M entradas produce una salida correspondiente a la suma de las muestras de entrada.

Eligiendo M de forma que sea potencia de 2, se puede implementar el promediador mediante un *shift* de bits, lo que reduce la complejidad del hardware necesario. Sin embargo, se debe considerar que en este punto de la cadena de procesamiento implementada, las muestras serán enviadas a un procesador de 32 bits (o 64 si se implementa sobre otro soporte). El procesamiento en software requiere de tipos de datos estandarizados, e.g. `int16_t`, y no se pueden trabajar con valores enteros de, por ejemplo, 12 bits. Teniendo en cuenta este motivo, se seleccionó $M = 16$ ya que permite aprovechar el crecimiento en la representación de las muestras dado por $log_2(M)$ bits, en la presente configuración $log_2(16) = 4$ bits, convirtiendo finalmente los frames en vectores de valores de $(16+16)$ bits.

Cabe aclarar que estos frames con representación extendida atraviesan el *packet dropper* y son transmitidos mediante el bus AXIS hacia la memoria DDR a través de la MMU del PS. Con el *pipline* propuesto, la tasa efectiva promedio de datos hacia el PS es de

$$\text{Tasa} = \frac{\text{ADC bits} \times f_s}{\text{M frames promediados}} \times \frac{\text{Representación final}}{\text{Representación inicial}}, \tag{6.2}$$

particularmente con la configuración por defecto resulta en

$$\text{Tasa} = \frac{(12+12)\text{bits} \times 80\text{Msps}}{16} \times \frac{(16+16)}{(12+12)} = 160\text{Mbps}^4.$$

6.3.5. Empaquetamiento de tramas y transmisión GbE

Los frames de datos son convertidos a paquetes UDP y transmitidos a través de Gigabit Ethernet (GbE) mediante software corriendo en el PS. Teniendo en cuenta el *Maximum Transmission Unit* (MTU) de 1500 bytes de GbE, el sistema se encarga también de fragmentar los frames en paquetes considerando que el comienzo de un frame coincida con el comienzo de un paquete[5]. Por ejemplo, para la configuración por defecto la estructura de datos queda según la Tabla 6.3.

Tamaño de frame [muestras]	640
Tamaño de muestra [bytes]	4 (2I + 2Q)
Tamaño de frame [bytes]	2560
Cantidad de paquetes UDP por frame	2
Tamaño de paquete UDP [bytes]	1280

Tabla 6.3: Estructura de datos para la configuración por defecto.

6.3.5.1. Arquitectura de Software del sistema embebido

Para la implementación de la transmisión de paquetes a través de UDP también se tuvo en cuenta el objetivo de ofrecer una arquitectura de software para prueba de algoritmos. En este punto particular se buscó desarrollar una infraestructura con una baja pisada o *footprint* de forma de dejar la mayor cantidad de recursos de procesamiento (CPU) disponibles, pero al mismo tiempo lo suficientemente flexible como para que el usuario final de la plataforma pueda implementar algoritmos (en software, para este punto en particular) sin tener que afrontar una complejidad excesiva.

El CPU disponible en la plataforma seleccionada posee dos núcleos ARM Cortex-A9 con coherencia de chaché que implementan la *Instruction Set Architecture* (ISA) ARMv7. Además, posee una MMU que permite la gestión de memoria virtual y paginada, interrupciones vectorizadas, y un conjunto de periféricos que incluyen UART, SPI, I2C, USB, entre otros. Dada la complejidad de la arquitectura, se decidió utilizar un Sistema Operativo de Tiempo Real o *Real-Time Operating System* (RTOS) para la gestión de los recursos del sistema. En este sentido, *FreeRTOS*[99] surgió como opción principal por su bajo *footprint* y su amplia documentación y soporte para la plataforma Zynq. El *First Stage Boot Loader* (FSBL) generado por las herramientas de Xilinx para la inicialización de la plataforma, inicializa el núcleo 0 de la CPU (CPU0) y desde allí se carga el sistema operativo.

[4]Notar que si la conexión con la PC o *host* es de 100 Mbps es necesario activar el *packet dropper* para no saturar el enlace.

[5]Notar que la recíproca no es válida

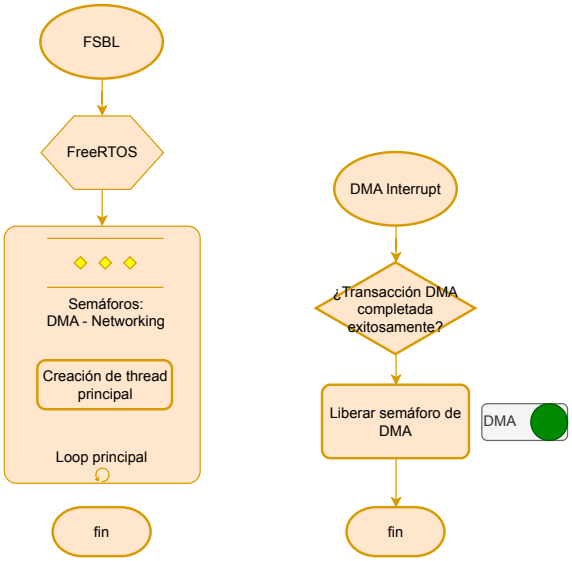
La Figura 6.19a muestra un esquema del flujo de booteo del sistema y de la rutina de interrupción asociada al DMA que se introdujo en la Sección 6.3. Las tareas básicas a atender por el PS para el funcionamiento del sistema son:

- UART: Recibir comandos de configuración y control desde el software de la PC y configurar los periféricos correspondientemente.

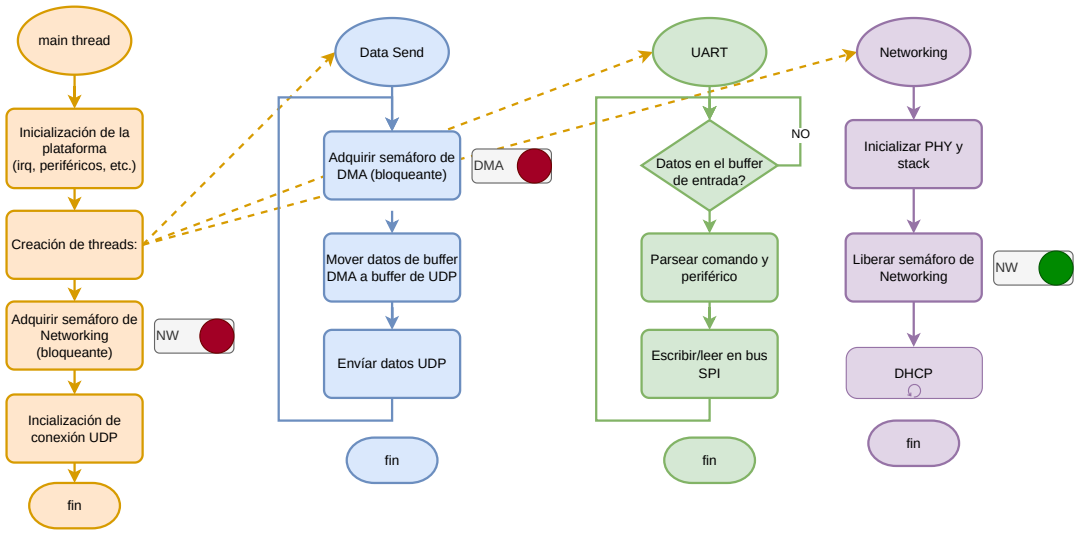
- DMA: Cada vez que existan datos disponibles en memoria provenientes del DMA, armar los paquetes UDP correspondientes y enviarlos por un puerto determinado.

- *Networking*: Proveer un cliente UDP que permita la transmisión de datos a través de la red, incluyendo las capas de IP y resolución de nombres (DHCP).

Para hacer un uso eficiente del tiempo de CPU, se definió un hilo de ejecución (*thread*) para cada tarea, de forma que el *scheduler* del sistema pueda administrar correctamente el tiempo de CPU asignado a cada una. Esto implicó prestar particular atención a la comunicación de datos entre *threads* para evitar condiciones de carrera y pérdida de datos por colisiones de escritura y/o lectura. Para ello se utilizó la infraestructura de semáforos provista por FreeRTOS. Particularmente, cabe destacar el uso de dos semáforos: DMA y Networking (NW). El primero se utiliza para preservar la integridad del buffer DMA, mientras que el segundo se utiliza para evitar que el *thread* principal intente inicializar un cliente UPD antes que el hardware físico esté listo. El detalle del flujo de estos hilos se encuentra en la Figura 6.19b.



(a) Arquitectura principal del sistema.



(b) Threads o hilos de ejecución

Figura 6.19: Esquema de la arquitectura de software del sistema embebido (CPU0).

Dentro del entorno de FreeRTOS, un *thread* se denomina *task*. En este trabajo se utiliza la denominación *thread* en concordancia con su significado general. A diferencia de un sistema operativo "completo" dónde el propio sistema crea procesos con su propio entorno de ejecución (espacio de memoria, estado del sistema, etc.) y los procesos solicitan la creación de threads "hijos", en FreeRTOS se puede pensar al propio sistema como un único proceso y cada "tarea" es un *thread* en el sentido mencionado anteriormente. Esto último implica que no existe una relación de "pertenencia" entre *threads* (o "tareas"), permitiendo que un *thread* se siga ejecutando aún cuando el *thread* que lo creó haya finalizado.

En la Figura 6.19a (izquierda) se muestra el proceso principal del *kernel* del sistema operativo. El mismo instancia los semáforos que controlan la sincronización entre *threads* y lanza el denominado `main thread` (Figura 6.19b izq.) Parte del sistema es la rutina de atención de interrupciones que se llama al haberse realizado una transacción desde el

DMA (Figura 6.19a derecha). La misma simplemente checkea que la transacción realizada por el DMA haya sido exitosa, y de ser así libera el semáforo para acceder a los datos.

El mencionado `main thread` es simplemente la "tarea" principal que crea al resto. Por todo lo mencionado, este *thread* finaliza una vez completada su función, sin que esto implique la finalización de los *threads* (o "tareas") que creó (los cuales nunca alcanzan el estado `fin` en condiciones normales). En la Figura 6.19b se puede observar la estructura de cada tarea o *thread*.

Finalmente, la utilización de los semáforos se puede analizar observando la Figura 6.19 en su totalidad. Los símbolos verdes indican la liberación de un semáforo, mientras que los rojos muestran dónde se lo quiere adquirir. Cabe destacar que la adquisición de semáforos es un proceso bloqueante que permite la gestión eficiente de tiempo de CPU por parte del *scheduler* del sistema. Es decir, mientras el proceso está bloqueado esperando por la liberación del recurso, el mismo no consume tiempo de CPU ya que el scheduler lo desaloja temporalmente.

A modo de ejemplo se menciona el funcionamiento del semáforo DMA en estado estacionario:

- La tarea de envío de tramas intenta enviar un buffer de datos, para ello solicita la adquisición sel semáforo DMA.
- Dado que dicho semáforo aún no fue liberado por la rutina de interrupción correspondiente, la tarea bloquea.
- El scheduler desaloja dicha tarea del CPU.
- En algún momento los datos provenientes del IP Core `plataformaUWB` generan una transacción DMA produciendo una interrupción.
- El CPU atiende la rutina de interrupción de DMA y la misma checkea si dicha transacción fue exitosa. De cumplirse esto último, se libera el semáforo DMA.
- Cuándo sea el turno de ejecución de la tarea DMA, esta ya no está bloqueada ya que puede adquirir el semáforo correspondiente por lo que el *scheduler* le dará tiempo de CPU a dicha tarea para completar sus actividades.
- La tarea luego de completar volverá a intentar adquirir el semáforo, pero como éste ya fue previamente adquirido (por ella misma al momento de enviar los datos) bloqueará nuevamente hasta repetirse el proceso.

6.4. Software visualizador y grabador de datos

El software visualizador y grabador de datos es el medio por el cuál el usuario final interactúa con la plataforma, ya sea para la configuración de los parámetros de la misma como para la adquisición de señales. Al mismo se lo denomina en este trabajo con el nombre de "UWPlot" y, del mismo modo que con el diseño de los circuitos de RF, se lo incluye por completitud aunque no formó parte del trabajo de investigación del tesista. La aplicación fue desarrollada en C++ debido a la búsqueda de rendimiento y en particular se utilizó el *framework* Qt, por lo que utiliza extensivamente el patrón de diseño

de *signal-slot* para el manejo de eventos asincrónicos y comunicación entre *threads*. Está compuesta por tres sub-módulos: comunicación, configuración y visualización (y grabación) corriendo en hilos de ejecución separados.

Comunicación

El módulo de comunicación se encarga de proveer un servidor UDP que permite la conexión de la plataforma como cliente a los puertos preestablecidos. La estructura de los paquetes que provee la plataforma (mencionada en la Sección 6.3.5) es conocida por el software, es decir, se definió un simple protocolo. Esto permite que el módulo ordene adecuadamente el contenido de los paquetes UDP y los almacene en una cola *thread-safe* para que luego el módulo visualizador y grabador los procese.

Visualización y grabación

Este sub-módulo se encarga de, por un lado, proveer una visualización en tiempo real de las señales UWB enviadas por la plataforma y recibidas y desempaquetadas por el sub-módulo de comunicación. Por otro lado, en otro hilo de ejecución registra a disco las señales en formato `numpy`. La visualización provee al usuario de cursores, zoom, etc. de forma de poder hacer un análisis completo de las señales recibidas, mientras que el proceso de grabación de señales, permite por ejemplo, controlar cada cuánto tiempo se quiere grabar un frame. Esto último es especialmente útil para mediciones de larga duración dónde no se requieren observar características que varíen muy rápidamente a lo largo del tiempo, permitiendo ahorrar espacio de almacenamiento. La Figura 6.20 muestra la interfaz de usuario para la visualización de señales.

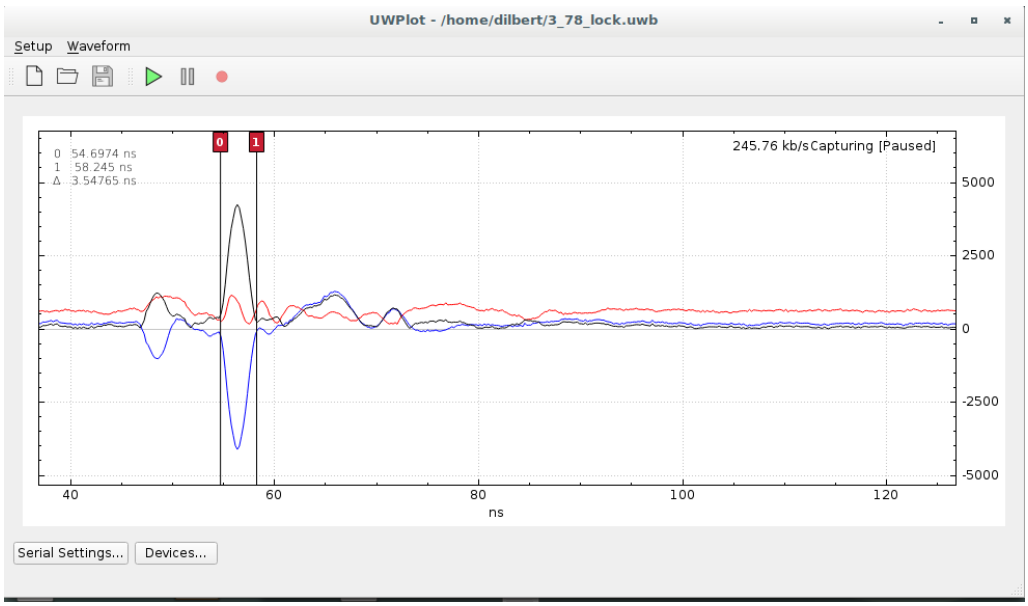


Figura 6.20: Interfaz gráfica de visualización de señales.

Configuración

Finalmente, el software facilita al usuario la configuración de los periféricos de la plataforma (ADC12DS105, LTC5586, TRF372017, PE43712). Para ello, este sub-módulo provee una interfaz gráfica de configuración según se muestra en la Figura 6.21. El mismo pre-calcula los valores necesarios en los registros de configuración de los dispositivos para obtener los parámetros deseados y envía la información al PS de la plataforma a través de una USB-UART. Para dicha comunicación también se definió un sencillo protocolo que enumera los dispositivos de forma que el PS pueda interpretar los mensajes y escribir los valores correspondientes al periférico SPI correcto.

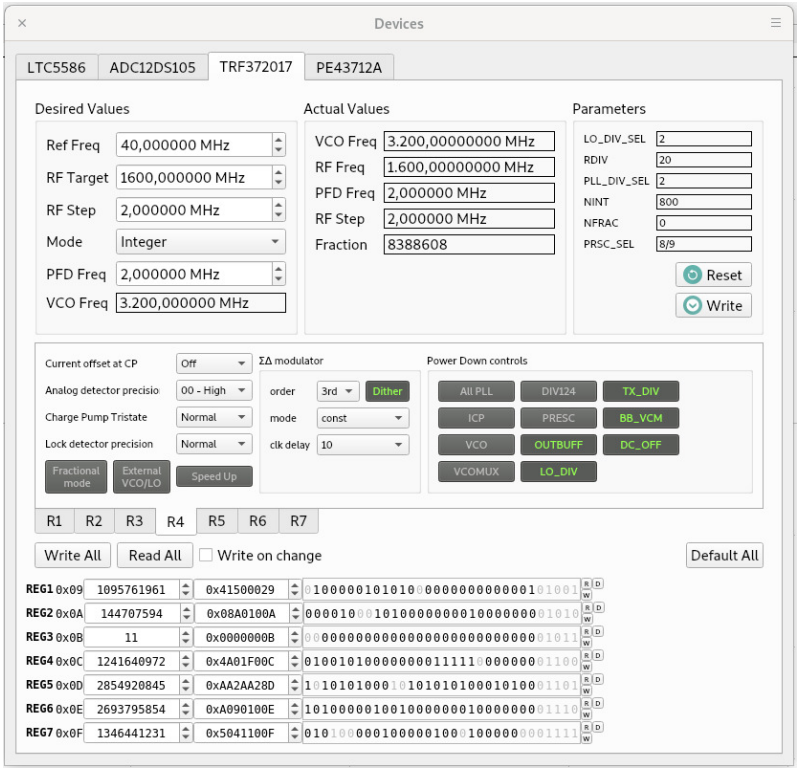


Figura 6.21: Interfaz gráfica de configuración de periféricos.

6.5. Resultados de implementación

La integración de todas las etapas (hardware analógico, hardware digital, hardware programable, sistema embebido y software de alto nivel) resultó en una plataforma experimental altamente configurable. La Tabla 6.4 muestra la cantidad de recursos utilizados dentro del SoC. Es importante destacar que la infraestructura desarrollada ocupa aproximadamente el 5 % de los recursos lógicos programables del área de PL. Esta economía de recursos permite utilizar la plataforma para desarrollar, implementar y evaluar distintos tipos de algoritmos de procesamiento, en concordancia con los objetivos propuestos para el desarrollo.

En cuanto al software embebido, la arquitectura propuesta deja 100 % disponible el CPU1 del PS y una baja carga sobre el CPU0 (dependiente de los parámetros de configuración).

Recurso	Utilizado	Disponible	Utilizado %
LUT	4540	78600	5.77
LUTRAM	268	26600	1.00
FF	4759	157200	3.02
BRAM	31	265	11.69
DSP	2	400	0.5
IO	23	250	9.2
MMCM	1	5	20.0
PLL	1	5	20.0

Tabla 6.4: Utilización de recursos del diseño completo (XC7Z030)

Finalmente, en la Figura 6.22 se muestra un análisis de la pisada (o *footprint*) de memoria del software de alto nivel. Del mismo surge que es necesario un bajo requerimiento de recursos, permitiendo que la arquitectura completa tenga potencial aplicación en procesamiento en el "borde" (*edge computing*)

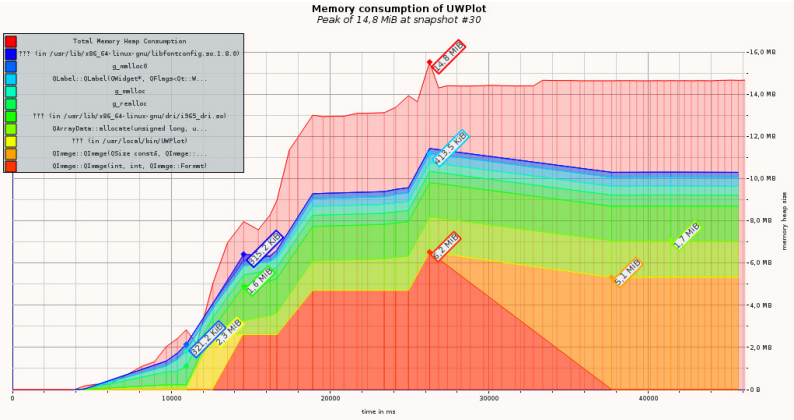


Figura 6.22: Uso de memoria del software en PC.

6.5.1. Mediciones de laboratorio

El correcto funcionamiento de la plataforma fue validado en el laboratorio. Para mediciones en el dominio del tiempo se utilizo un osciloscopio digital analizador de señales marca Tektronix, modelo MSO70404. El mismo posee una tasa de muestreo en tiempo real de 25 GS/s (con 5,5 bits efectivos por muestra) y un ancho de banda de $4GHz$ (@ $-3dB$)[6]. Para evaluar la cadena de transmisión-recepción-adquisición se utilizó el banco que se esquematiza en la Figura 6.23.

El objetivo de la medición fue:

- Validar la correcta generación de pulsos UWB por parte de la FPGA.

[6]La configuración de la plataforma por defecto en banda base cae dentro del rango de trabajo, sin embargo, cuando se analiza la señal modulada, esta posee algunas componentes por fuera del mismo. Se espera que esto genere cierta distorsión a los pulsos elevados en frecuencia. De todas formas, las mediciones permiten verificar cualitativamente el diseño en el dominio del tiempo.

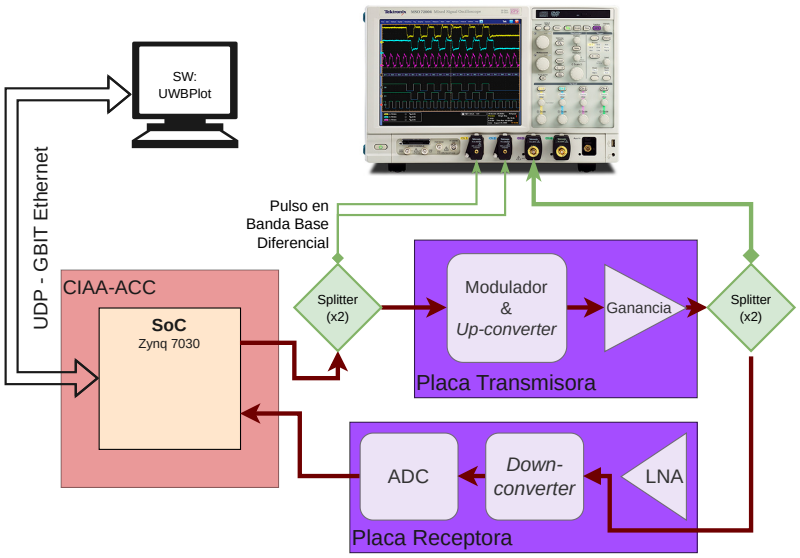


Figura 6.23: Configuración de la medición en laboratorio.

- Validar el esquema de muestreo propuesto en la Sección 6.3.2.1, incluyendo la correcta digitalización de los pulsos UWB por parte del ADC.

- Verificar la correcta modulación de los pulsos UWB por parte del transmisor.

- Obtener una medición cualitativa de las distorsiones producidas al pulso por las dispersiones en los componentes, desadaptaciones, limitaciones físicas, etc.

Con el osciloscopio se capturó por un lado el pulso en banda base (diferencial) generado por la FPGA, y adicionalmente la salida modulada del transmisor. En la Figura 6.24 además se puede observar el pulso en modo común calculado por el instrumento como la diferencia entre las señales de los canales correspondientes. En la misma se puede notar también el retardo introducido en la señal modulada por la propagación del pulso a través del prototipo y el transmisor.

El ancho de pulso ideal a partir de una frecuencia de $700\,Hz$ sería de $\sim 1{,}43\,ns$, mientras que el ancho medido fue de $\sim 1{,}8\,ns$. Se puede observar también la fuga de portadora a través del transmisor, que se encuentra en el orden de los $-20\,dB$. Ambas magnitudes se encuentran dentro de las dispersiones esperadas.

En paralelo, con el software ”UWPlot” se capturó la señal recibida por la propia plataforma para evaluar la correcta digitalización. La captura se puede visualizar en la Figura 6.25.

Si bien ambas señales se obtienen en puntos distintos de la cadena (no se disponen de puntos de medición accesibles en la entrada del ADC), se puede observar que la señal digitalizada por la plataforma es consistente con la señal transmitida y si bien la forma sufre distorsiones al atravesar toda la cadena de transmisión/recepción, el ancho de pulso se mantiene dentro de los requerimientos.

Para evaluar más en detalle la respuesta, se almacenó la captura del software ”UWPlot” en formato `numpy` y los datos del osciloscopio en formato `csv`. Dado que ambas señales fueron digitalizadas con frecuencias de muestreo distintas, se realizó un preprocesamiento para poder realizar una comparación representativa. En particular la señal del

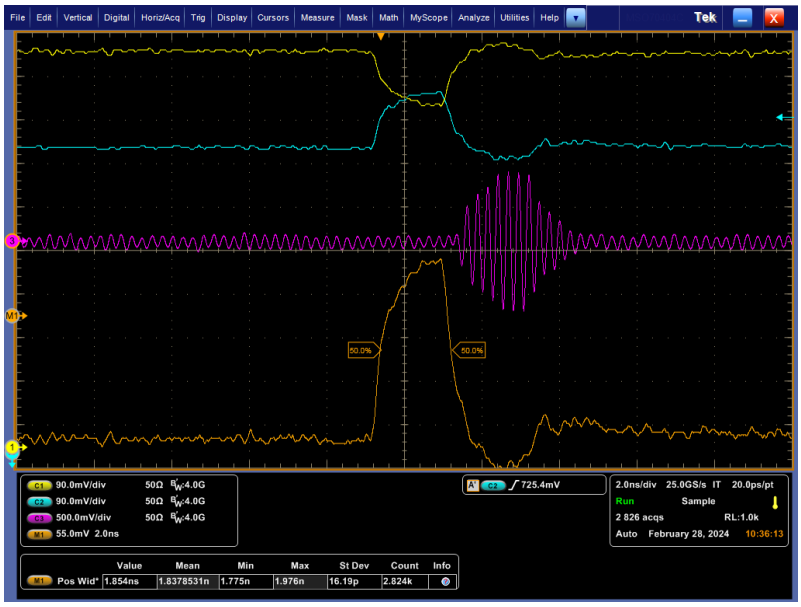


Figura 6.24: Pulso UWB capturado con el osciloscopio. Los canales 1 y 2 corresponden al pulso diferencial generado por la FPGA y el canal 3 a la salida del transmisor.

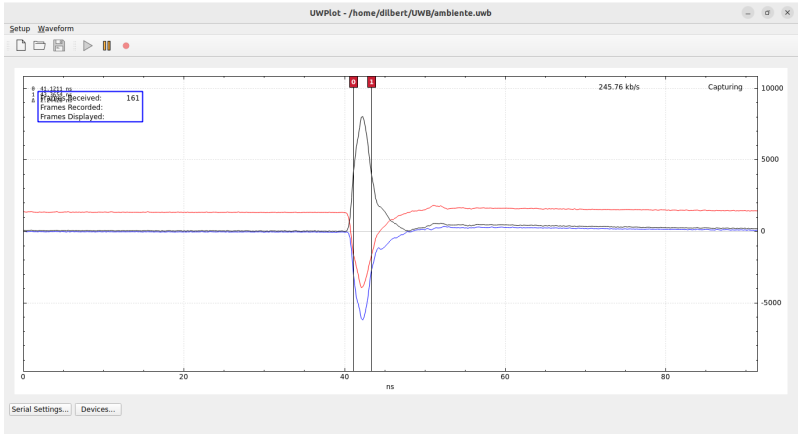


Figura 6.25: Pulso UWB capturado por la plataforma.

osciloscopio abarca un tiempo de $20ns$ y consta de 1000 puntos[7]. La señal capturada por el software desarrollado consta de $2 \cdot 640$ muestras (I+Q) a $f_s = 5{,}04GHz$ durante un período de $T = 640/f_s \simeq 126{,}9ns$. Para realizar la comparación entonces se pre-procesó según la siguiente metodología:

- Se obtuvo el valor absoluto punto a punto de la señal de "UWPlot".
- Se eliminó el valor medio de ambas señales.
- Se normalizó la amplitud de las mismas.
- Se recortó la señal del software a $20ns$ en torno al pulso en cuestión.
- Se remuestreó dicho fragmento de la señal a $50GHz$ (1000 puntos) mediante el método de Fourier.

El resultado de la comparación se muestra en la Figura 6.26.

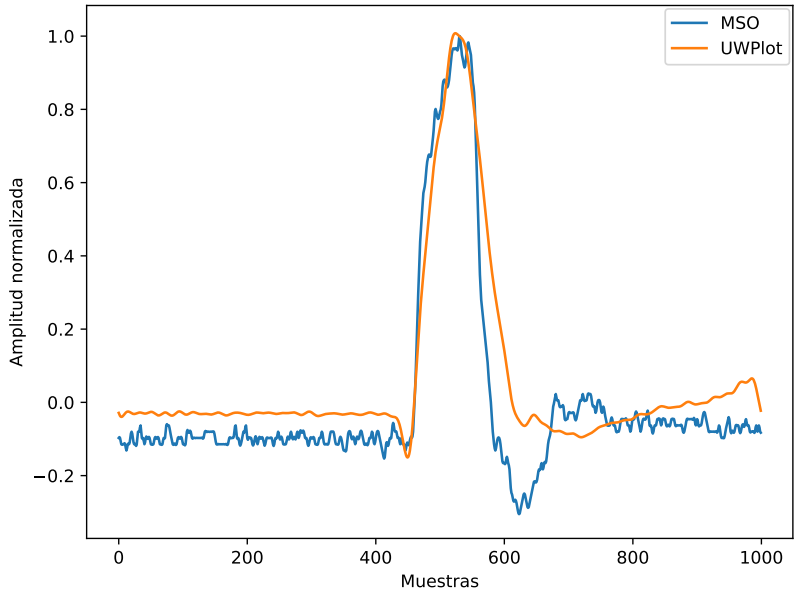


Figura 6.26: Comparación de la señal capturada por el software y el osciloscopio.

La comparación confirma que la señal digitalizada por la plataforma es consistente con la señal transmitida y que el ancho de pulso se mantiene dentro de los requerimientos. Se puede observar también que la señal digitalizada por la plataforma presenta un menor ruido debido al promediado de *frames* implementado en la FPGA. De esta forma se pudo validar el método de muestreo propuesto y la cadena completa, exceptuando la transferencia de las antenas transmisora y receptora.

Finalmente, se realizó la medición de contenido espectral del pulso transmitido mediante un analizador de espectro marca Agilent, modelo N9030A. Debido a que el pulso en banda base se genera en forma diferencial y que además posee un contenido de continua elevado, se utilizó la salida modulada para realizar la medición. La Figura 6.27 muestra el espectro obtenido.

[7]Cabe aclarar que el osciloscopio realiza un procesamiento digital de la señal para obtener la señal representada en pantalla y los puntos no necesariamente se condicen con las muestras adquiridas. De hecho, el cálculo en base a los valores presentados arroja una frecuencia de muestreo de $50GHz$, que es el doble de la frecuencia máxima del osciloscopio.

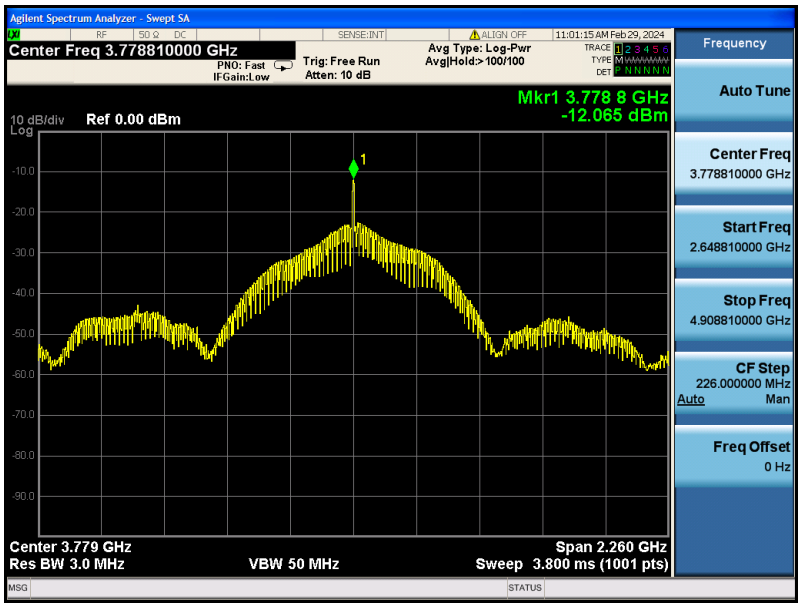


Figura 6.27: Espectro del pulso UWB transmitido (modulado).

La medición arrojó un ancho de banda de $565MHz$ a $-10dB$ y de $847{,}5MHz$ a $-20dB$ y una separación entre mínimos de $1{,}017GHz$ coherente con el medido en el dominio del tiempo de $2/1{,}85ns$, e inferior que el teórico estimado de $1{,}4GHz = 2 \cdot 700MHz$, tal como fue notado en las mediciones en el dominio del tiempo con el osciloscopio.

6.6. Resumen

En el presente capítulo se presentó el diseño e implementación de una plataforma configurable para la generación y adquisición de pulsos UWB y el procesamiento de los mismos. La Tabla 6.5 resume los requerimientos planteados y los resultados de la implementación.

Tabla 6.5: Resultados de la implementación vs. requerimientos.

Parámetro	Requerimiento	Valor obtenido
f_{RF}	Hasta $4{,}5GHz$	Hasta $4{,}8GHz$ (Configurable)
BW	Mayor a $500MHz$	$565MHz$ @$-10dB$; $847{,}5MHz$ @$-20dB$
$f_{muestreo}$	Mayor a $2GSps$	Totalmente configurable a través de K y N: $5{,}04GS/s$ para la configuración por defecto.
N_{bits}	Mayor a 8 bits	$12+12$ bits (I+Q)
Modulación	en cuadratura (I+Q)	en cuadratura (I+Q)
PRF	Mayor a $30Fps$	Configurable, por defecto: $7{,}85MHz$ sin promediado $7{,}85MHz/16 = 490{,}625kHz$ con promediado

El esquema de muestreo propuesto resultó adecuado para cumplir con los requerimientos para UWB. Los recursos de procesamiento disponibles y la configurabilidad de la plataforma la posicionan como una solución interesante para la investigación y desarro-

llo de aplicaciones UWB, desde la realización de experimentos con las señales generadas, hasta la implementación de algoritmos en hardware. Finalmente, la construcción del prototipo diseñado con componentes *off-the-shelf* demostró cumplir con los requerimientos planteados.

Capítulo 7

Resultados experimentales

RESUMEN: En este capítulo se presentan los resultados experimentales de pruebas realizadas para aplicaciones particulares de la tecnología UWB. Para las mismas se utilizó la plataforma desarrollada en este trabajo, cómo así también el algoritmo de ASST propuesto. La Sección 7.2 presenta los resultados obtenidos con blancos estáticos en aplicaciones de sensado de humedad en materiales. Luego, la Sección 7.3 avanza en los resultados obtenidos con blancos dinámicos aplicados a detección de signos vitales, ya sea mediante el procesado de un *dataset* con señales de referencia, como también a través de procesamiento en tiempo real.

7.1. Introducción

Como fue mencionado, la tecnología UWB posee el potencial de ser utilizada en una amplia variedad de aplicaciones tecnológicas. Disponiendo de una plataforma de pruebas propia y de una plataforma comercial Xethru (ver Sección 5.3), el presente capítulo presenta los resultados de los experimentos realizados con el objetivo de estudiar la aplicabilidad de la tecnología en varios contextos. La primera etapa se focalizó en la aplicación de las señales UWB en la extracción de características de blancos estáticos debido al mayor control disponible de las condiciones experimentales. Particularmente, la detección de contenido de humedad es una aplicación de interés en la industria de la construcción y la agricultura.

La segunda etapa avanzó hacia la detección de características dinámicas. La aplicación que se estudió es la detección de signos vitales sin contacto y es del mayor interés en el área de la salud. Específicamente, se platea extraer la frecuencia respiratoria y cardíaca a partir de la iluminación del torso humano con señales UWB. En este caso, las condiciones experimentales son más desafiantes y la validación de los resultados más compleja. Esto se debe a la dificultad práctica de obtener señales de referencia para comparar los resultados obtenidos con la tecnología UWB. Para encarar esta etapa se plantearon dos estrategias. Por un lado, se utilizó un *dataset* público[100] que contiene señales de radar UWB y señales de referencia sincronizadas de signos vitales, publicado durante el período de desarrollo del presente trabajo. Estas pruebas permitieron validar el rendimiento del algoritmo de ASST propuesto en la Sección 4.3 en cuanto a la extracción de componentes o modos de la señal. Por otro lado, se desarrolló un software en tiempo real que procesa con el algoritmo ASST las señales UWB provenientes de la plataforma Xethru. Esta

implementación muestra las capacidades en tiempo real del algoritmo en cuestión, y su economía de recursos computacionales.

7.2. Resultados con blancos estáticos

Cómo fue mencionado, la aplicación de estudio fue la detección y clasificación de humedad en materiales. Particularmente, se estudió la detección de humedad en arena, siendo esta un material central tanto en la industria de la construcción como en la agricultura. Se realizaron una cantidad considerable de mediciones con el objetivo de disponer de un *dataset* que permita entrenar y validar algoritmos de clasificación. Finalmente, se procesaron los datos obtenidos para evaluar la aplicabilidad de la tecnología.

7.2.1. Configuración del experimento

La plataforma se configuró con los parámetros por defecto:

Parámetro	Valor
Frecuencia de portadora	3,785 GHz
Ancho de pulso	1,8 *ns* (medido)
PRF	7,875 MHz
Promedidado de *frames*	16
Polarización de la antena	Vertical

Tabla 7.1: Parámetros de configuración de la plataforma de radar UWB.

Elementos bajo prueba

Se utilizaron 6 recipientes de plástico herméticos, de 10 *cm* de ancho por 24 *cm* de largo y 20 *cm* de altura. Cada recipiente se llenó con 6 *kg* de arena de construcción seca. Se numeraron los recipientes del 1 al 6 y el recipiente 5 se dejó como recipiente testigo.

Configuración del banco de ensayos

Se utilizó una mesa de madera (debido a su baja reflectividad electromagnética), dónde se fijó tanto la plataforma UWB como las antenas. A una distancia de 80 *cm* de la plataforma se fijó una grilla con diferentes ángulos y distancias para una fácil ubicación de los recipientes. La Figura 7.1 muestra un esquema de la vista superior del banco de ensayos.

Las mediciones fueron tomadas en condiciones controladas en la cámara semi-anecoica (CSA) de INTI para el recipiente 6 en todas las humedades, y para todos los recipientes secos. La configuración de la medición se muestra en la Figura 7.2.

Para el resto de los recipientes, las mediciones se realizaron en el laboratorio de Comunicaciones con condiciones de *clutter* reales para todas las humedades según la disposición de la Figura 7.3.

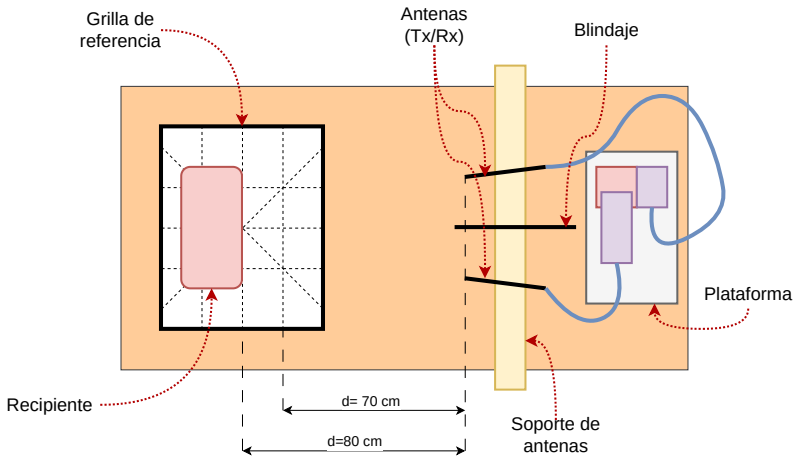
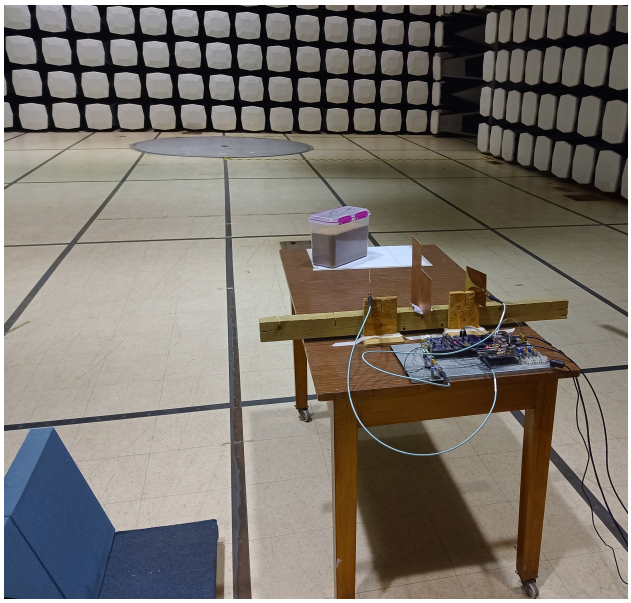
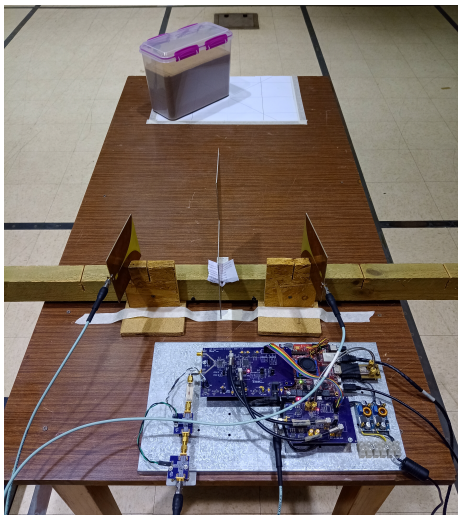


Figura 7.1: Esquema de la vista superior del banco de ensayos (no a escala).



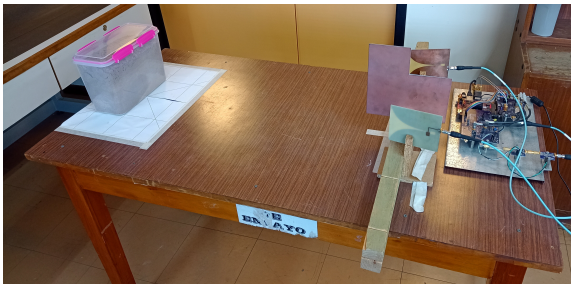
(a) Disposición general del ensayo.



(b) Detalle de la mesa de medición.

Figura 7.2: Configuración del banco de mediciones en la CSA de INTI.

(a) Disposición general del ensayo.



(b) Detalle de la mesa de medición.

Figura 7.3: Configuración del banco de mediciones en el laboratorio.

Metodología

El procedimiento de medición se resume en el siguiente flujo de trabajo:

Algoritmo 6 Procedimiento de medición

1: Para cada recipiente en $\{1,2,3,4,6\}$
2: Se coloca el recipiente con arena seca en la grilla y en la posición de $0°$ y $d = 80\ cm$.
3: Se graba aproximadamente $1 min$ de señal.
4: Se rota el recipiente $45°$.
5: **if** La posición del recipiente es $225°$ y $d = 80\ cm$ **then**
6: Acercar el recipiente a $d = 70\ cm$ y ubicar a $0°$ e ir a 3.
7: **else if** La posición del recipiente es $225°$ y $d = 70\ cm$ **then**
8: Se agregan $200 ml$ de agua y se homogeneiza la mezcla.
9: **if** La mezcla satura **then**
10: Finalizar medición del recipiente e ir a 17.
11: **else**
12: Ubicar a $d = 80\ cm$ y $0°$ e ir a 3
13: **end if**
14: **else**
15: Ir a 3
16: **end if**
17: **if** Quedan recipientes por medir **then**
18: Ir a 2 para el siguiente recipiente.
19: **else**
20: Finalizar medición.
21: **end if**

Finalmente, se obtuvieron 763 mediciones, en archivos `.npy` conteniendo cada uno información de aproximadamente 1 *min* de captura de señal. Cada archivo está guardado con formato `complex64` y contiene la información de los pulsos recibidos en banda base. Adicionalmente, se etiquetó cada captura con el objetivo de poder usar los datos para

entrenamiento y validación de algoritmos de clasificación.

7.2.2. Cadena de procesamiento de datos o *pipeline*

De una observación de los datos crudos se puede concluir que la información de humedad está presente junto con otras características más dominantes de la señal cómo la configuración geométrica de la medición. La Figura 7.4 muestra algunas señales correspondientes a diferentes recipientes con el mismo nivel de humedad, cómo así también del mismo recipiente con diferentes niveles de humedad para una misma posición/distancia.

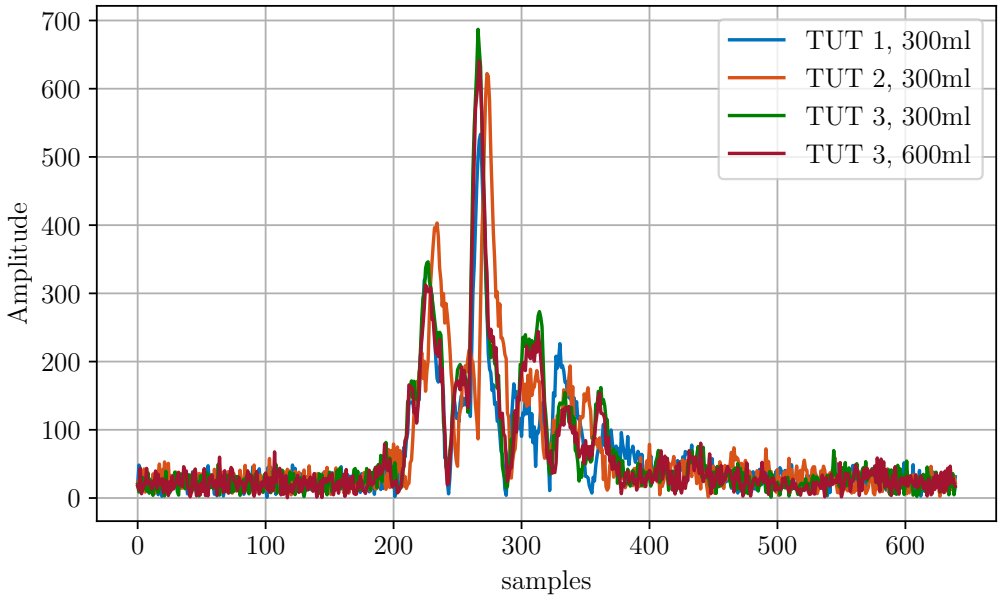


Figura 7.4: Señales para diferentes recipientes y niveles de humedad.

Para extraer la información de humedad se propuso entonces un *pipeline* de procesamiento que incluya la extracción de modos naturales de la señales, según [28]. La Figura 7.5 muestra el diagrama de flujo de la cadena de procesamiento compuesta por un estimador de cuadrádos mínimos (LSE) para la eliminación del clutter, la extracción de modos naturales, una *Radon - Cumulative Distribution Transform* para transformar el dominio a un espacio más separable y un análisis por discriminador lineal (LDA) para separar las clases.

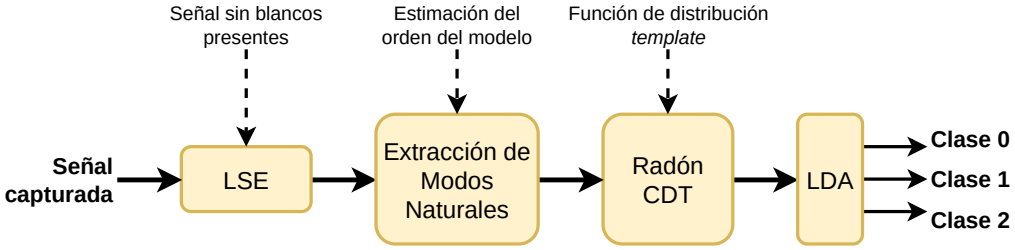


Figura 7.5: Diagrama de flujo de la cadena de procesamiento para la extracción de humedad.

Como primer aproximación, y con el objetivo de oficiar como prueba de concepto,

se definieron tres clases correspondientes a tres rangos de humedad añadida (Δ) según la Tabla 7.2.

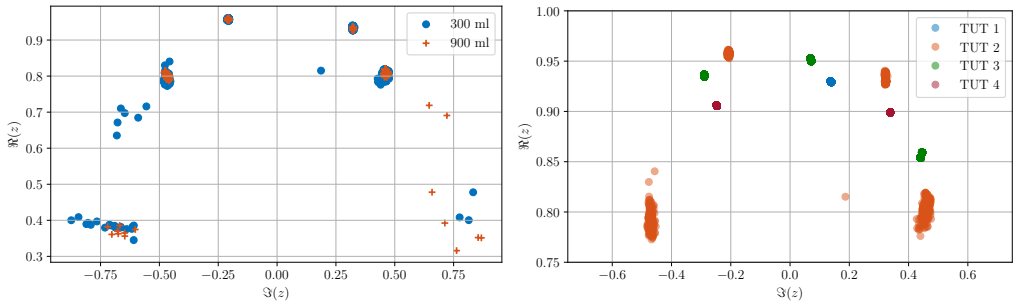
Clase	Clase 0	Clase 1	Clase 2
Humedad (Δ $[ml]$)	$\Delta \leqslant 400$	$400 < \Delta \leqslant 800$	$800 < \Delta \leqslant 1300$

Tabla 7.2: Clases de humedad definidas para la clasificación.

La cadena de procesamiento comienza con la eliminación del clutter mediante un estimador de cuadrados mínimos (LSE) en base a mediciones ambientales tomadas en el laboratorio.

La extracción de modos naturales se realiza mediante la construcción de un tensor en base a matrices de Hankel contenidas a partir de desplazamientos de cada *frame* de la señal de entrada. Con el tensor mencionado, se aplica primero una descomposición por rango del tensor [101] dónde el mismo se expresa como una suma de tensores de rango 1, y luego se aplica una estimación espectral [102]. Dicha estimación espectral realiza un proceso de eliminación de ruido que mantiene la estructura del problema para luego aplicar ESPRIT. En esta etapa se utiliza como parámetro de entrada el orden del modelo. El mismo es estimado utilizando la técnica descripta en [103].

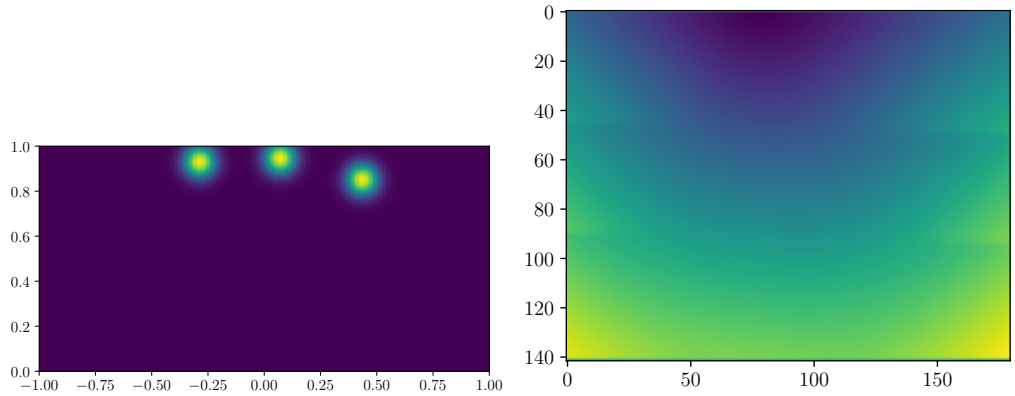
Los modos obtenidos se encuentran aglutinados en ciertas regiones del plano z, lo que dificulta la separación de las clases. Este hecho motiva la utilización de una transformación de dominio para lograr una separación de clases mayor. Adicionalmente, los modos tienden a ubicarse en regiones cercanas a la circunferencia unitaria. Esto se puede observar en la Figura 7.6 muestra la distribución de los modos naturales obtenidos para dos señales correspondientes al mismo recipiente con dos contenidos de humedad diferentes.



(a) Dos humedades para el mismo recipiente (b) Misma humedad para diferentes recipientes

Figura 7.6: Distribución de modos naturales.

Dada esta configuración de singularidades en el plano z, se propuso una transformación de Radon-Cumulative Distribution (RCDT) para llevar los modos a un espacio más separable. La misma es una transformación que convierte un espacio dónde las clases son no linealmente separables en uno dónde lo son [104]. Para poder aplicar la transformación se debe convertir el mapa de modos o singularidades del plano z en una función similar a una distribución de probabilidad. Para ello se convoluciona el mapa de modos con un kernel Gaussiano y se normaliza el resultado. Un ejemplo del resultado de esta última operación se muestra en la Figura 7.7a y su transformación con la R-CDT en la Figura 7.7b.



(a) Mapa de singularidades transformado a PDF (b) Radon Cumulative Distribution Transform

Figura 7.7: Transformación de modos naturales con R-CDT.

Habiendo transformado las señales al espacio linealmente separable, se procedió a aplicar un análisis discriminante lineal (LDA) para separar las clases. Los resultados de entrenar dicho discriminante con las señales provenientes de los recipientes $1, 2$ y 3 ubicados a 70 y 80 *cm*, con ángulos de $0°$ y $180°$, y testear con las señales del recipiente 4 se muestran en la Figura 7.8.

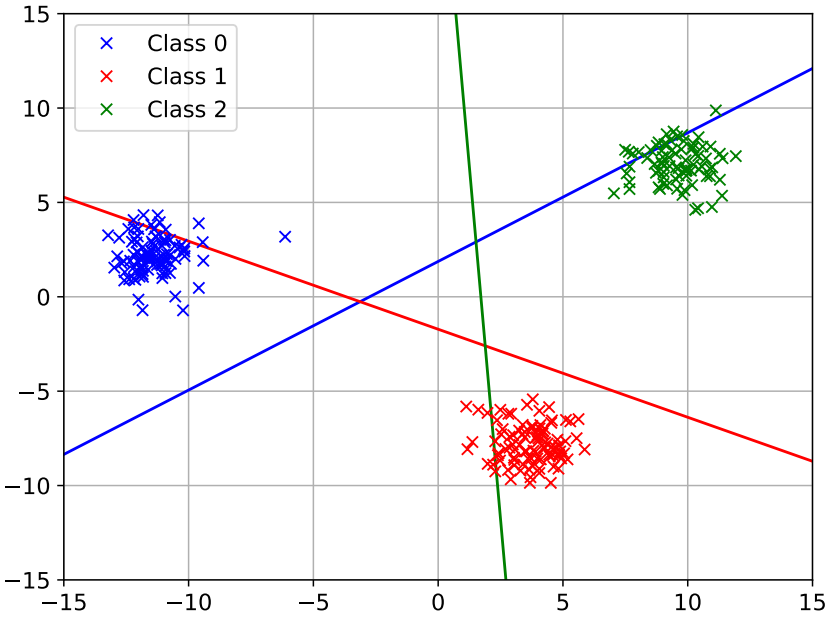


Figura 7.8: Resultados de la clasificación con LDA.

Para evaluar la presición del clasificador se utilizó un esquema de validación cruzada de 10 iteraciones. Los resultados de la validación se muestran en la Tabla 7.3, dónde se comparan con resultados obtenidos con un clasificador SVM lineal, sin aplicar la transformación R-CDT.

Se puede observar que la inclusión de la transformación R-CDT en el pipeline de procesamiento resulta indispensable para obtener resultados favorables respecto de la precisión del clasificador.

Tabla 7.3: Precisión de la estimación usando SVM lineal y la Radon-CDT.

Método	Precisión
Linear SVM	0.36
Radon-CDT	0.937

Asimismo, con estos resultados obtenidos se puede concluir que la tecnología UWB presenta gran potencial para la detección de humedad en materiales, y que tanto la plataforma desarrollada como la cadena de procesamiento propuesta arrojan resultados satisfactorios en la aplicación particular.

7.3. Resultados con blancos dinámicos

7.3.1. Extracción de signos vitales a partir de un conjunto de datos público

En esta sección se utilizó el *dataset* publicado por Shi et al [100], para validar la aplicación del algoritmo ASST en la separación de señales fisiológicas a partir de una señal de radar. El mismo contiene señales de radar provenientes de iluminar a diferentes personas en variadas condiciones con un radar UWB de onda contínua junto con las señales fisiológicas correspondientes obtenidas por métodos tradicionales (ECG, termistor, etc.) sincronizadas. Esto permite verificar los resultados obtenidos del procesamiento de las señales UWB contrastándolos con las señales de referencia. Un script de prueba (`process_data_example.py`) se proporciona en el repositorio del *toolbox* desarrollado en este trabajo (`adaptivesswt`)[59] para descargar y procesar automáticamente los datos a modo de demostración.

La señal de radar UWB se utilizó para estimar la señal respiratoria, la frecuencia cardíaca y la señal cardíaca en el rango de audio (fonocardiograma (PCG), o audio cardíaco). Cabe aclarar que los efectos físicos por los que se generan y obtienen las señales son diferentes a las señales de referencia. Esto es, la señal de radar se obtiene a partir de la reflexión y *scattering* de ondas electromagnéticas en el cuerpo humano que contienen información de movimientos mecánicos en el mismo, mientras que las señales de referencia se obtienen a partir de la actividad eléctrica del corazón asociadas a los impulsos generados internamente. Por otro lado, la señal respiratoria de referencia fué obtenida mediante un método térmico, sensando la temperatura del aire entrante/saliente. Es por esto que, si bien las características principales de las señales son comunes (por ejemplo, frecuencia instantánea, ubicación de los pulsos, etc.), las formas de onda correspondientes son inherentemente diferentes.

Para la extracción de las señales fisiológicas correspondientes a partir de la señal de radar, se usó el algoritmo de ASST propuesto en la Sección 4.3 como método de análisis TF en las bandas de interés de cada señal. Las estimaciones obtenidas a través del análisis TF se compararon con las obtenidas utilizando las técnicas tradicionales[1].

En los siguientes ejemplos ilustrativos se muestra el análisis de la señal contenida en el archivo `DATASET_2017-02-16_11-05-06_Person 11.mat` que contiene un evento de

[1] teniendo en cuenta lo mencionado respecto de la diferente naturaleza del origen de las mismas.

apnea embebido.

7.3.1.1. Cadena de procesamiento

El *pipeline* de procesamiento utilizado para esta aplicación particular se muestra en la Figura 7.9.

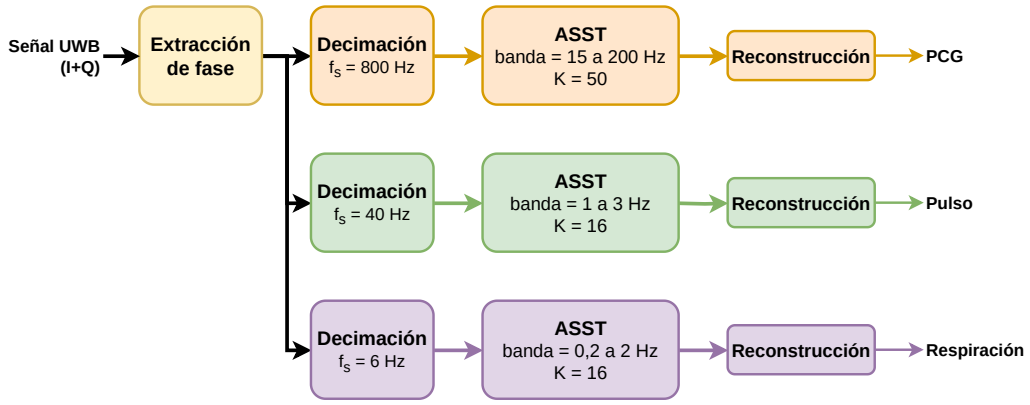


Figura 7.9: Diagrama de flujo de la cadena de procesamiento para la extracción de signos vitales.

Dado que la señal del *dataset* ha sido preprocesada[100] y se encuentra en banda base, la primera etapa en la cadena de procesamiento es el cálculo de la fase de la señal de radar. Las variaciones de fase de la señal se corresponden con variaciones de la distancia al blanco, pudiendo así capturar los movimientos toráxicos correspondientes a la respiración y los movimientos cardíacos. El análisis se realiza en tres bandas diferentes, una para cada señal a obtener. Para cada banda, el primer bloque de la cadena de procesamiento es un decimador entero, que reduce la frecuencia de muestreo de la señal a un valor adecuado. Cabe destacar que las señales del *dataset* no presentan todas la misma frecuencia de muestreo, por lo que este bloque ayuda también a homogeneizar los datos. Las bándas para el análisis TF son las siguientes:

- PCG: $15 - 200\ Hz$.
- Frecuencia cardíaca: $1 - 3\ Hz$.
- Respiración: $0{,}2 - 2\ Hz$.

Cada una de dichas bandas se analiza con el algoritmo de ASST para hacer el seguimiento instantáneo de las frecuencias de interés. Finalmente, se realiza una reconstrucción de la señal para obtener su representación en el dominio del tiempo.

7.3.1.2. Audio cardíaco

El audio cardíaco se obtiene analizando los datos de radar en el rango de frecuencias de 15 a $200Hz$ como fue mencionado. La información en la banda de PCG de la señal de radar está distribuida de manera uniforme dado su carácter impulsivo. Debido a esto, una distribución uniforme de frecuencias es la mejor opción. En este caso, el algoritmo propuesto no concentra las frecuencias en ninguna banda en particular, lo que lleva a que

la reconstrucción de las señales obtenidas con diferentes variaciones del método propuesto sea casi idéntica y no se logre ninguna ganancia de rendimiento con respecto a la SST tradicional. Cuándo se analiza la banda en cuestión con el método TSST, se observa una localización más precisa de los pulsos cardíacos, tal como se espera del método. Estos efectos se pueden ver en la Figura 7.10 donde, por un lado todas las trazas de la señal reconstruida en base a las diferentes variaciones de SST son similares, mientras que la señal obtenida a través de la TSST muestra una característica más impulsiva. Cuando las señales reconstruidas se comparan con las señales de electrocardiograma (ECG) y PCG sincronizadas, se puede observar que los pulsos correspondientes a los sonidos S1, S2 y ocasionalmente S3 se extraen correctamente y se sincronizan completamente con las mismas. Estos datos se pueden utilizar para mejorar la detección de la frecuencia instantánea (pulso), o para extraer características de la dinámica cardíaca.

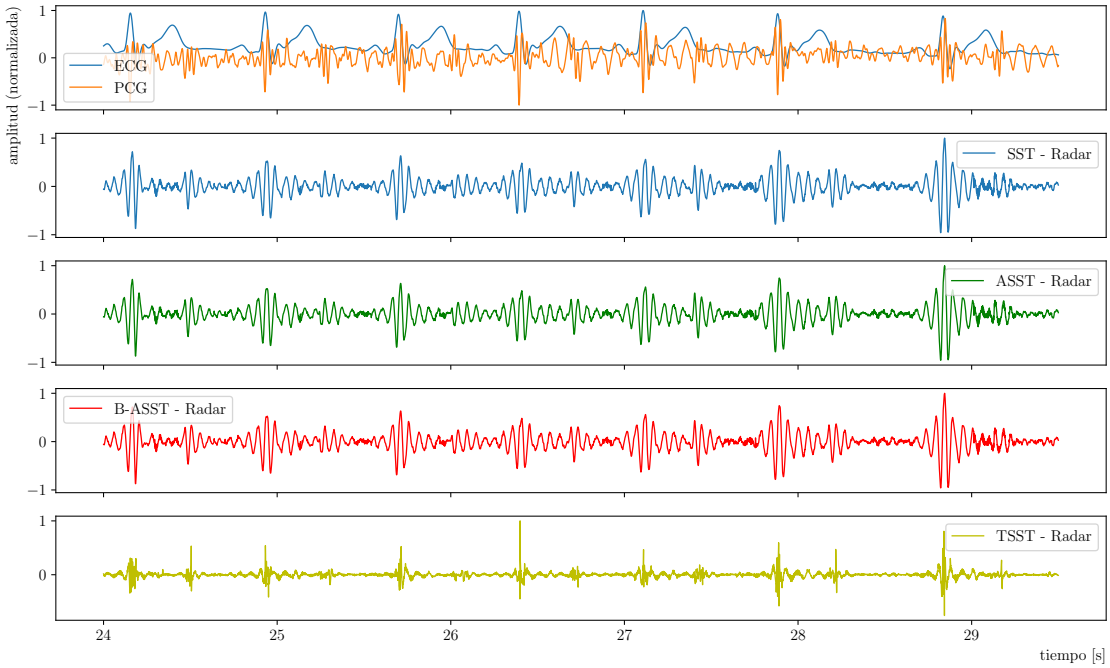


Figura 7.10: Señal de audio obtenida a partir de datos de radar vs ECG.

Si bien en esta aplicación particular de extracción de ”audio cardíaco” el algoritmo adaptativo no presenta ninguna ventaja respecto del método de *Synchrosqueezing* tradicional, el método de TSST muestra una resolución temporal mucho más alta.

7.3.1.3. Frecuencia cardíaca

La extracción de la frecuencia cardíaca o “pulso” es una tarea compleja, ya que las armónicas de la respiración (con mucha más energía) pueden estar presentes en las mismas bandas (1 a 3 Hz) [105]. Dicha señal se puede asociar a una única componente o modo con una frecuencia instantánea determinada, por lo que es de esperar que el algoritmo de ASST ofrezca una mejora respecto de la SST tradicional. Esto se puede observar en las representaciones TF de la Figura 7.11, dónde se observa una concentración de la localización de las frecuencias de análisis en torno a las bandas dónde se encuentra la energía de la señal. Cuando se analiza la misma señal por lotes (B-ASST), se observa una distribución

más dinámica de las frecuencias de análisis variando para cada lote en particular. Los resultados en el dominio del tiempo se muestran en la Figura 7.12 comparados con el ECG correspondiente. Si bien las señales reconstruídas por los diferentes métodos, proveen un seguimiento adecuado de la frecuencia cardíaca, se puede observar que en situaciones de variaciones altas de la frecuencia cardíaca las dos versiones adaptativas del algoritmo son capaces de seguir más precisamente los pulsos (por ejemplo, en los pulsos para $t \simeq 29$ y 30 s).

Para este caso particular, solo se han utilizado $K = 12$ frecuencias de análisis para probar el rendimiento del algoritmo utilizando recursos computacionales bajos.

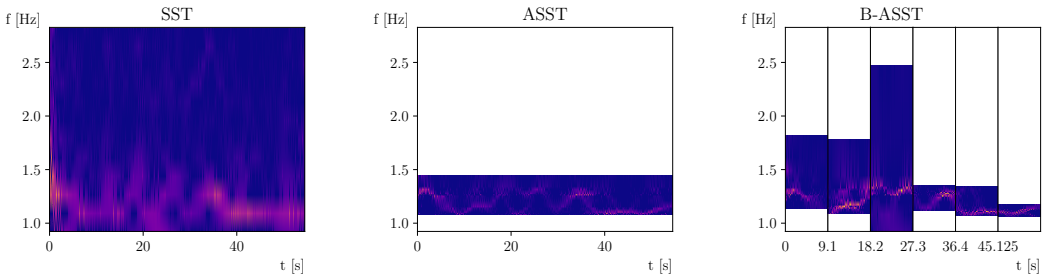


Figura 7.11: Comparación de métodos para extraer la frecuencia cardíaca instantánea de la señal de radar.

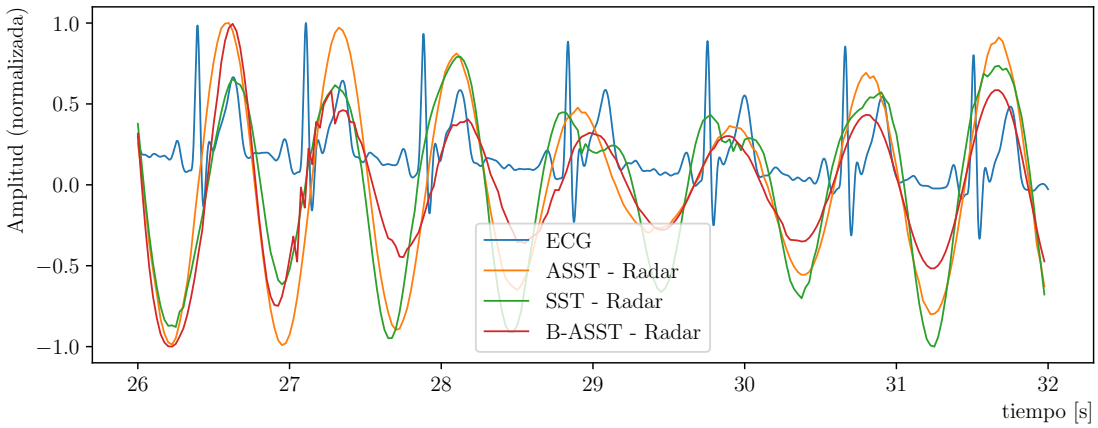


Figura 7.12: Señales de frecuencia cardíaca reconstruídas del radar UWB usando ASST y B-ASST comparadas con un ECG tradicional.

Para este caso el algoritmo mostró un rendimiento muy bueno en términos de seguimiento de la frecuencia cardíaca instantánea a partir de la señal de radar UWB (sin contacto) cuándo se compara con el ECG tradicional. Esto se pudo lograr incluso en condiciones adversas con muy pocas frecuencias de análisis y, por lo tanto, un bajo costo computacional.

7.3.1.4. Señal respiratoria

Para la señal respiratoria el análisis se ha realizado sobre el rango de 0,05 a $1\,Hz$ utilizando solo $K = 16$ frecuencias de análisis. La Fig. 7.13 muestra la señal reconstruida para el SST estándar, el ASST y su versión *batched* (B-ASST), en comparación con la señal de respiración recogida de un sensor térmico, incluida en el *dataset*. Es evidente que el algoritmo de *synchrosqueezing* proporciona una precisión notable en el seguimiento de las frecuencias instantáneas, sin embargo, se presenta un límite máximo en la resolución debido al número de frecuencias seleccionadas para el análisis.

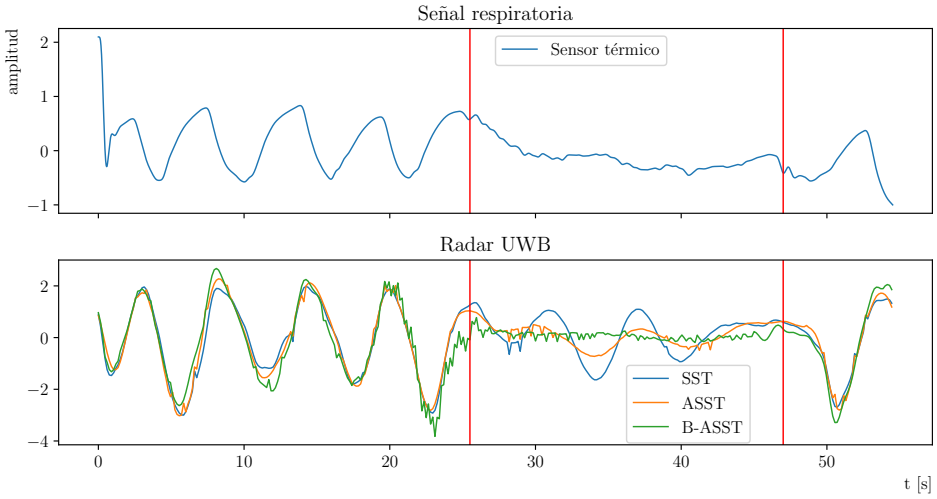


Figura 7.13: Comparación de la señal respiratoria obtenida a partir de un sensor térmico vs. radar UWB utilizando SST, ASST y B-ASST. Entre las bandas rojas se puede observar un evento de apnea.

Durante el evento de apnea, la versión adaptativa proporciona una mejor estimación de la señal, aunque la variante por lotes puede detectar con mayor precisión la interrupción del movimiento respiratorio. El algoritmo SST tradicional, por otro lado, ante el evento de apnea no realiza un seguimiento adecuado, manteniendo la frecuencia anterior a la interrupción.

7.3.2. Monitorización de signos vitales en tiempo real

Utilizando los resultados de la sección anterior, y en base a la capacidad del algoritmo desarrollado, se implementó un sistema (aplicación) de monitoreo de frecuencia cardíaca en tiempo real basado en la plataforma Xethru. En este caso se adaptó la cadena de procesamiento para eliminar el *clutter* de la señal antes de ser analizada con el algoritmo ASST. La misma se muestra en la Figura 7.14.

El primer bloque de la cadena realiza un PCA de un sólo término, ya que el *clutter* típicamente está asociado a la primer componente. El proceso de eliminación de *clutter* comienza con una etapa de entrenamiento, dónde durante un período de tiempo configurable se capturan frames de radar y se las organiza cómo una matriz. Sobre dicha matriz, se realiza el PCA de una única componente que se almacena para su uso en tiempo de ejecución. Para sustraer el *clutter* de la señal en tiempo real, se calcula la proyección de

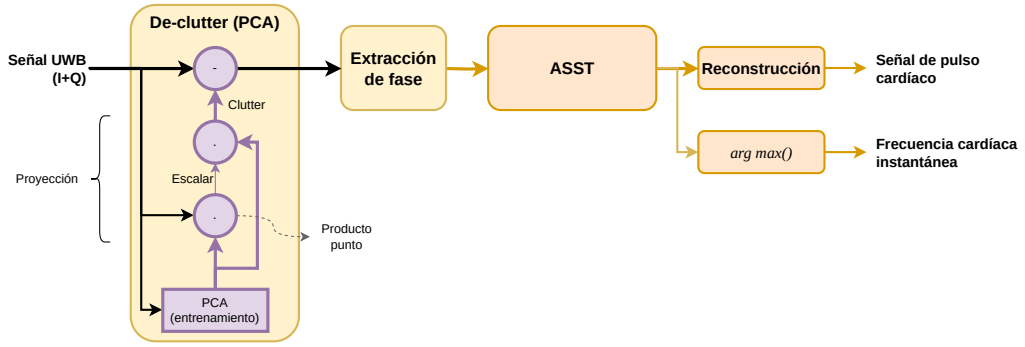


Figura 7.14: Diagrama de flujo de la cadena de procesamiento para la extracción de signos vitales en tiempo real.

la señal actual sobre la componente obtenida en el entrenamiento y se la resta de la señal original.

Luego, al igual que en el caso *off-line*, se realiza la extracción de la fase de la señal compleja, asociada a la posición instantánea del blanco, en este caso el tórax. Si bien se desarrolló la cadena de procesamiento solamente para la extracción de la frecuencia cardíaca o pulso, resulta sencillo observar la respiración a partir de la señal de fase. El software desarrollado también permite configurar en tiempo real todos los parámetros de análisis, de forma de poder extraer otras características de la señal.

7.3.2.1. Implementación

Uno de los objetivos del desarrollo de este sistema es evaluar la aplicabilidad del algoritmo desarrollado a problemas de resolución en tiempo real y adicionalmente evaluar la integración con otros paquetes de procesamiento de señales. Para ello se desarrolló el software `uwbmonitor` en Python, en base al *framework* Qt. La implementación está basada en un modelo de múltiples hilos de ejecución, lo que evalúa también la capacidad de interacción del paquete `adaptivesswt` en un ecosistema complejo. La arquitecutra de hilos del sistema es la siguiente:

- **Hilo de adquisición:** Se encarga de la lectura de cada frame de la plataforma Xethru, a través de un timer configurado para realizar una interrupción a la misma frecuencia que la cantidad de frames por segundo configurada en la plataforma. Con cada interrupción se emite la señal `dataAvailable`.
- **Hilo de procesamiento:** Se encarga de procesar la señal de radar. Recibe la señal `dataAvailable` y dispara la cadena de procesamiento. Por cada etapa del *pipeline* se emite una señal con los datos de salida de dicha etapa.
- **Hilo de grabación:** Se encarga de grabar en un archivo `.npy` la señal recibida. Los datos a grabar se reciben a través de la señal emitida por la etapa correspondiente del hilo de procesamiento.
- **Hilo de interfaz:** Contiene el loop de eventos de Qt. Se encarga de la interacción con el usuario y la visualización de los resultados.

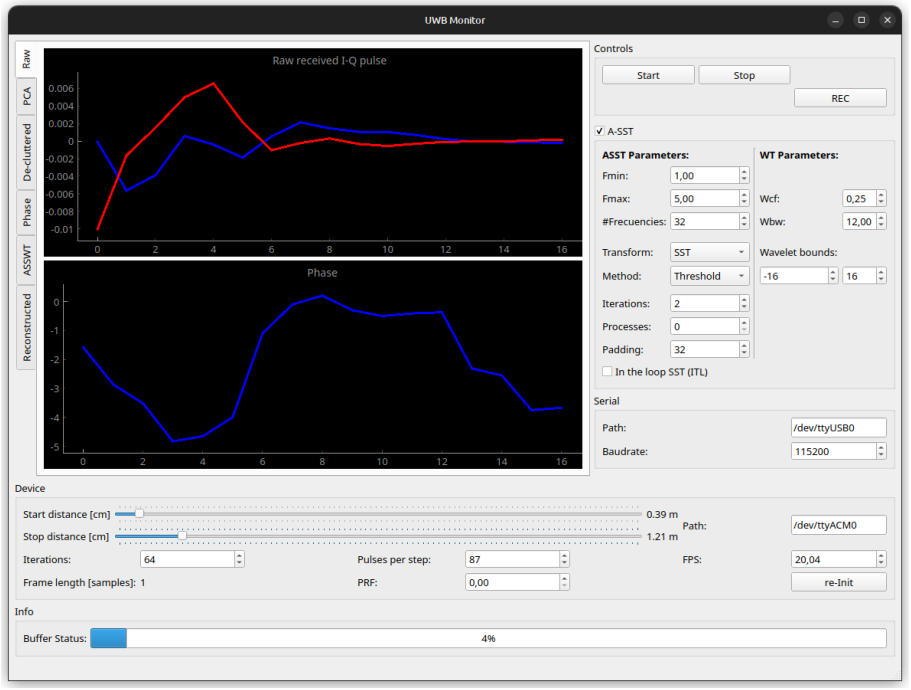
La interfaz del software permite monitorear las señales en cada etapa de procesamiento y ajustar los parámetros de análisis en tiempo real. La Figura 7.15 muestra la interfaz

gráfica para cada etapa del *pipeline*. La señal cruda se muestra en la pestaña *Raw* (Figura 7.16a). La pestaña *PCA* permite realizar el cálculo de la componente principal y visualizarla, mientras que la señal sin clutter puede ser observada en la pestaña *De-cluttered*. En dicha pestaña se puede configurar la distancia dónde se muestrea el pulso en el caso que esté configurado como manual este parámetro. Si está configurado como automático la distancia al torso se lee de un sensor ToF adosado a la plataforma. En dicha posición se evalúa la fase de la señal, que se puede monitorear en la pestaña *Phase*. A esa señal de fase, al igual que con el *dataset* de la Sección 7.3.1, se le aplica el algoritmo ASST y su representación TF se puede observar en tiempo real en la pestaña *ASST*. Finalmente, la señal reconstruída se puede observar en la pestaña *Reconstructed*. Globalmente se puede configurar las frecuencias mínima y máximas de análisis, la cantidad de frecuencias a utilizar, el método de remapeo (SST, TSST, SET, RM), la cantidad máxima de iteraciones, el padding y los parámetros de la Wavelet a utilizar. Adicionalmente se pueden configurar los parámetros de adquisición de la plataforma Xethru.

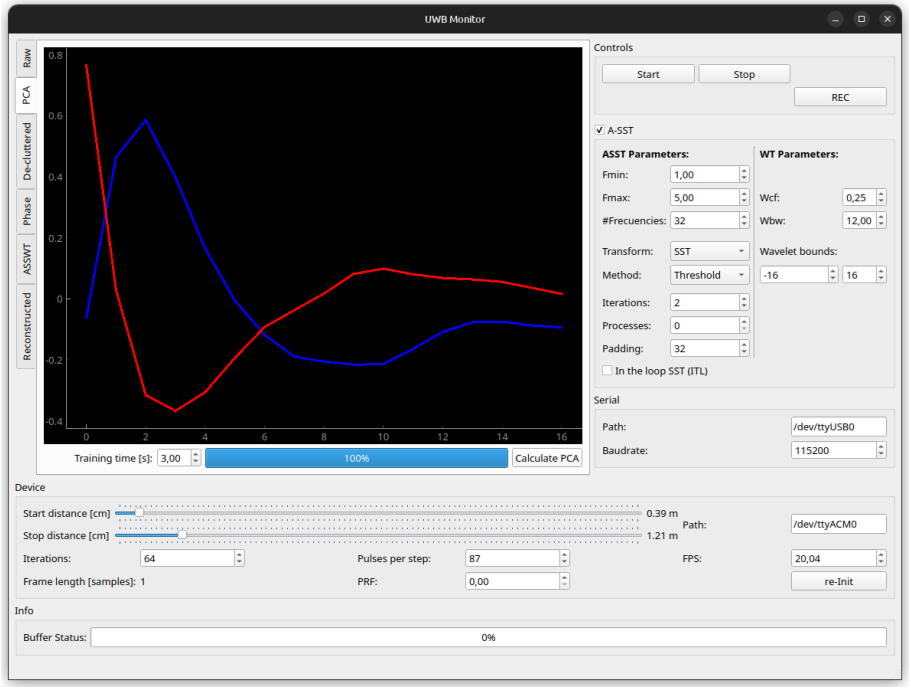
Toda la aplicación mostró un funcionamiento satisfactorio en tiempo real obteniendo la frecuencia cardíaca en situaciones controladas. Si bien el problema de detección de signos vitales sin contacto es complejo debido a la variabilidad de las señales por movimientos involuntarios o del entorno, el objetivo de lograr una aplicación en tiempo real se cumplió satisfactoriamente y nuevamente se pudo demostrar la aplicabilidad de la tecnología y el algoritmo desarrollado en aplicaciones concretas.

7.4. Resumen

En este capítulo se implementaron aplicaciones particulares de la tecnología UWB. Particularmente, se utilizó para la clasificación de diferentes humedades en arena y para la extracción de signos vitales. A partir de los resultados obtenidos se pudo observar un gran potencial de aplicabilidad de la tecnología. La plataforma desarrollada resultó ser una herramienta versátil y adecuada para la evaluación e implementación de algoritmos y aplicaciones. Adicionalmente, el algoritmo ASST propuesto obtuvo resultados satisfactorios en su aplicación a la extracción de signos vitales, tanto en su capacidad de extracción y seguimiento de componentes particulares, como en su despliegue en aplicaciones de tiempo real.

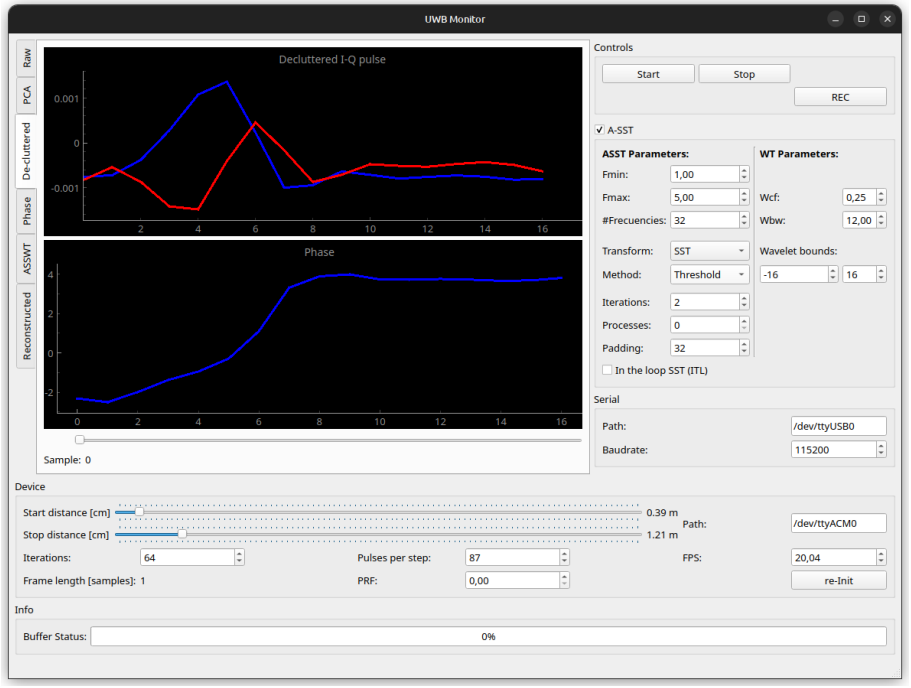


(a) Datos crudos.

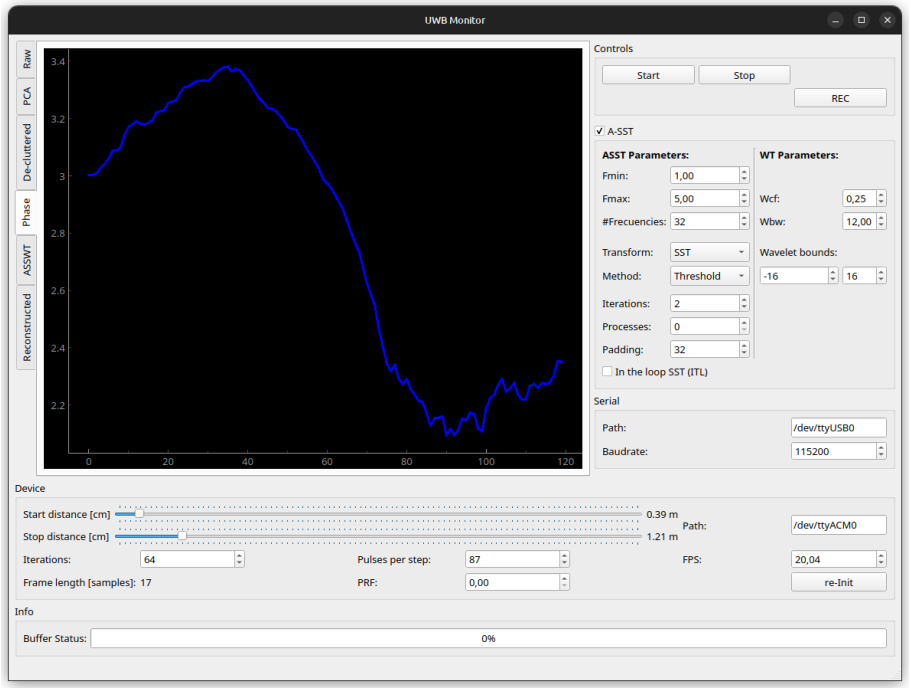


(b) PCA.

Figura 7.15: Interfaz gráfica del software `uwbmonitor`.

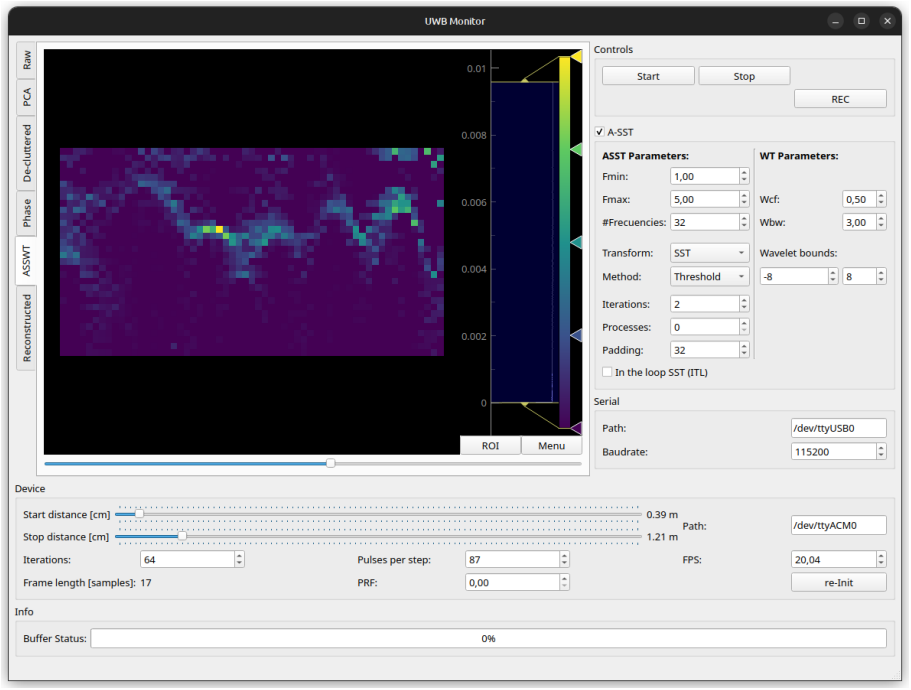


(c) Señal *decluttered*.

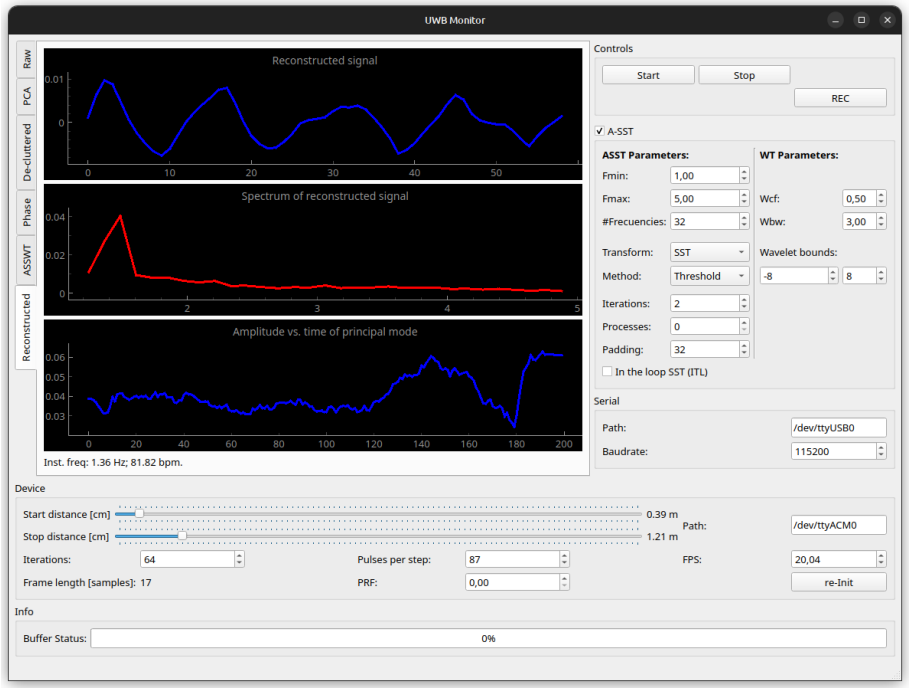


(d) Fase.

Figura 7.15: Interfaz gráfica del software `uwbmonitor` (cont.).



(e) ASST.



(f) Análisis de la señal reconstruida.

Figura 7.15: Interfaz gráfica del software `uwbmonitor` (cont.).

Conclusiones

En esta tesis se ha abordado el estudio del problema de clasificación y/o monitoreo de blancos en escenarios de cercanía a través de señales RF; motivado por el avance de las tecnologías de la información y la comunicación (TIC) hacia la integración del sensado dentro de los sistemas de transmisión de datos. La tecnología UWB, como una evolución del Radar tradicional, fue propuesta para la investigación, aprovechando sus características de alta resolución espacial, alta inmunidad a las interferencias por caminos múltiples (fundamental para ambientes interiores de *clutter* intenso) y alto contenido de información de las señales recibidas. El estudio inicial del estado del arte permitió identificar dos aspectos centrales que la literatura no aborda de manera conjunta: la generación/adquisición de señales UWB y la disponibilidad de algoritmos de análisis con capacidad de ser aplicados a aplicaciones de tiempo-real o requerimientos de baja latencia.

Este trabajo introdujo un nuevo algoritmo de análisis TF con el objetivo de extraer la evolución temporal de componentes de frecuencia embebidos en señales UWB. El algoritmo propuesto, denominado ASST, se basa en la transformada *Synchrosqueezing* y fue ideado desde su concepción para ser aplicado en tiempo-real. Como se menciona en la Sección 4.3.2, el mismo se basa en la reorganización de las frecuencias de análisis en función de la distribución espectral de energía de la señal, permitiendo una mejor resolución sin incrementar los recursos computacionales considerablemente. Esta redistribución junto con el método de reasignación seleccionado produjo una reducción en el error de estimación instantánea de frecuencia como se muestra en la Sección 4.5. Las técnicas utilizadas para la implementación permitieron, por un lado, parametrizar el método de reasignación de frecuencia, pudiéndose elegir entre SST, TSST, SET y RM. Por otro lado, se explotó el paralelismo del algoritmo y se logró una implementación eficiente en cuanto a recursos computacionales. La Sección 4.5.3 muestra un análisis en dicho sentido sobre diferentes plataformas.

En cuanto a la generación y adquisición de señales UWB, se presentó el desarrollo completo de una plataforma experimental. La misma fue desarrollada utilizando componentes de disponibilidad comercial (*off-the-shelf*) considerando la configurabilidad, replicabilidad, accesibilidad y costo. Asimismo, fue diseñada con objetivos de investigación y experimentales, de forma que permita su utilización en diferentes aplicaciones y escenarios. Como diferencial, la misma cuenta con una considerable capacidad de procesamiento interna, ofreciendo la posibilidad de descargar computo sobre el propio hardware. La Sección 6.5 muestra que la infraestructura desarrollada para la generación y muestreo tiene una huella muy pequeña en cuanto a utilización de recursos de la FPGA y el procesador embebido, dejando una considerable capacidad de cómputo disponible para experimentación. La medición de las señales generadas y capturadas con equipamiento de laboratorio mostró un buen desempeño de la plataforma, especialmente considerando su fabricación con componentes de bajo costo.

Finalmente, se evaluó la aplicabilidad tanto de la plataforma como de los algoritmos desarrollados por el tesista en escenarios reales. Particularmente se analizaron los problemas de detección de humedad en materiales y la detección de signos vitales sin contacto. En ambos casos se lograron resultados satisfactorios, mostrando la potencialidad de la plataforma experimental y el rendimiento del algoritmo desarrollado.

Contribuciones

La presente tesis aborda desde una perspectiva integral y conjunta los problemas de generación y adquisición de señales UWB, y de análisis no estacionario de las mismas. Con un enfoque en la aplicación de la tecnología a problemas del mundo real, se hizo foco en implementaciones orientadas a tiempo-real.

Particularmente:

- El tesista desarrolló un algoritmo adaptativo de análisis TF publicado en [22] y disponible públicamente y empaquetado como un paquete o *toolbox* de Python. Este algoritmo aprovecha el paralelismo de las diferentes plataformas de cómputo a través de *multiprocessing* y *multithreading* para sistemas con múltiples núcleos e hilos de procesamiento respectivamente. Adicionalmente, se incorporó el uso de *OpenCL* permitiendo además explotar el *ultra-paralelismo* de GPUs y FPGAs. Los resultados numéricos mostraron una mejora en cuanto al error de estimación de frecuencia instantánea en comparación con otros métodos de la literatura y un desempeño computacional competitivo.

- Como segunda contribución el tesista publicó en [106] (y complementariamente en [89],[90]) el desarrollo e implementación de una plataforma experimental de hardware para generación y adquisición de señales UWB. Con un método de muestreo en tiempo equivalente diseñado para no requerir ADC de costo elevado y ser altamente configurable, la plataforma permite obtener señales UWB de alta calidad. La plataforma también provee al usuario recursos computacionales para descargar procesamiento en el hardware. Para replicar la misma sólo se necesitan componentes *off-the-shelf* resultando en una mayor accesibilidad y un aporte para la comunidad científica/académica que necesite experimentar con señales UWB.

- Finalmente, se presentan aplicaciones del desarrollo encarado en este trabajo a problemas reales. Se muestra que la tecnología ofrece una solución viable para problemas de clasificación de materiales y monitoreo en escenarios dinámicos en tiempo real. En particular, se obtuvieron resultados satisfactorios en la detección de humedad en materiales utilizando la plataforma y un *pipeline* de procesamiento adecuado. Además, se logró la extracción de signos vitales sin contacto, mediante la aplicación del algoritmo ASST tanto en un *dataset*, cómo en datos adquiridos en tiempo real.

Trabajos futuros

A partir de los experimentos llevados a cabo en esta tesis, surgen varios aspectos para ser abordados en futuras investigaciones.

En términos de aplicación de la tecnología UWB a sensado de humedad, se propone utilizar redes Bayesianas para la clasificación. Durante la experimentación con redes neuronales tradicionales para la detección de diferentes humedades (Sección 7.2), se observó una clara tendencia al *overfitting* o "memorización" de casos cuando se usan las trazas temporales o representaciones TF en lugar de los modos naturales. Esto se atribuye a que las características (*features*) de las señales asociadas al contenido de humedad son débiles respecto de otras (como ser la configuración geométrica del ensayo/blanco) en dichas representaciones. El uso de redes Bayesianas permitía mitgar el *overfitting* y obtener una estimación de la incertidumbre de clasificación más adecuada.

Por otro lado, al momento de la escritura de esta tesis se está trabajando junto con empresas del sector agropecuario en el estudio del potencial despliegue de esta tecnología para detección de humedad en suelos en tiempo real a bordo de máquinas cosechadoras. En ese caso se propone el estudio de un problema geométrico distinto al de clasificación con objetos, ya que se puede modelar el escenario como un plano con dimensiones tendiendo a infinito, a una distancia relativamente cercana a las antenas. Adicionalmente, se plantea como objetivo poder caracterizar el perfil del suelo en términos de humedad y densidad, por lo que se abordará el estudio de detección de interfaces entre materiales con distinto ε_r y aplicar las técnicas probadas en este trabajo a porciones de la señal asociadas con una capa o estrato particular.

En el caso de detección de signos vitales, se propone el estudio de la utilización de *redes adversarias generativas* para transformar la señal obtenida a partir de UWB en una señal similar a un ECG. El problema a resolver en este sentido es la trasformación de dominio de señales obtenidas a partir de señales electromagnéticas perturbadas por el movimiento cardíaco (UWB) al dominio de señales eléctricas conducidas generadas a partir de la contracción del músculo (ECG). Este problema es de particular importancia para que los médicos, entrenados en interpretar señales de ésta última naturaleza, puedan reconocer y detectar patologías en señales cuyo origen físico es distinto. Lograr este objetivo permitiría avanzar en la utilización de sensores sin contacto para el monitoreo de signos vitales.

En cuanto a cuestiones de implementación, la investigación y desarrollo de una infraestrucura totalmente integrada entre PC y plataforma permitiría a investigadores y desarrolladores de aplicaciones, acelerar enormemente los tiempos para realizar experimentos y/o implementaciones. El uso de Linux en reemplazo de FreeRTOS en la plataforma permitiría una mayor flexibilidad tanto para el desarrollo de algoritmos de software embebido en el PS sino también en la capacidad de reconfiguración en tiempo de ejecución del hardware en la PL. Por otro lado, el desarrollo e implementación de una API en lugar de una aplicación permitiría una enorme versatilidad en cuanto a implementación en PC.

La tecnología UWB ha demostrado en base a lo explorado en este trabajo, un enorme potencial para el despliegue de aplicaciones de sensado, abriendo un abanico de tópicos a investigar y desarrollar a partir de los resultados expuestos en la presente tesis.

Apéndice A

Esquemáticos y Arquitecturas de Hardware

A.1. Arquitectura de hardware programable

En la Figura A.1 se muestra el esquemático de la implementación del *top-level* de la arquitectura de procesamiento. Todos los puertos de entrada/salida de este esquema se corresponden con pines físicos del circuito integrado (XC7Z030-2BFG676).

Nota: *Las figuras en esta sección son imágenes vectoriales generadas por las herramientas de diseño. En la versión electrónica de este documento es posible incrementar el zoom para analizar el esquemático/PCB sin pérdida de detalles.*

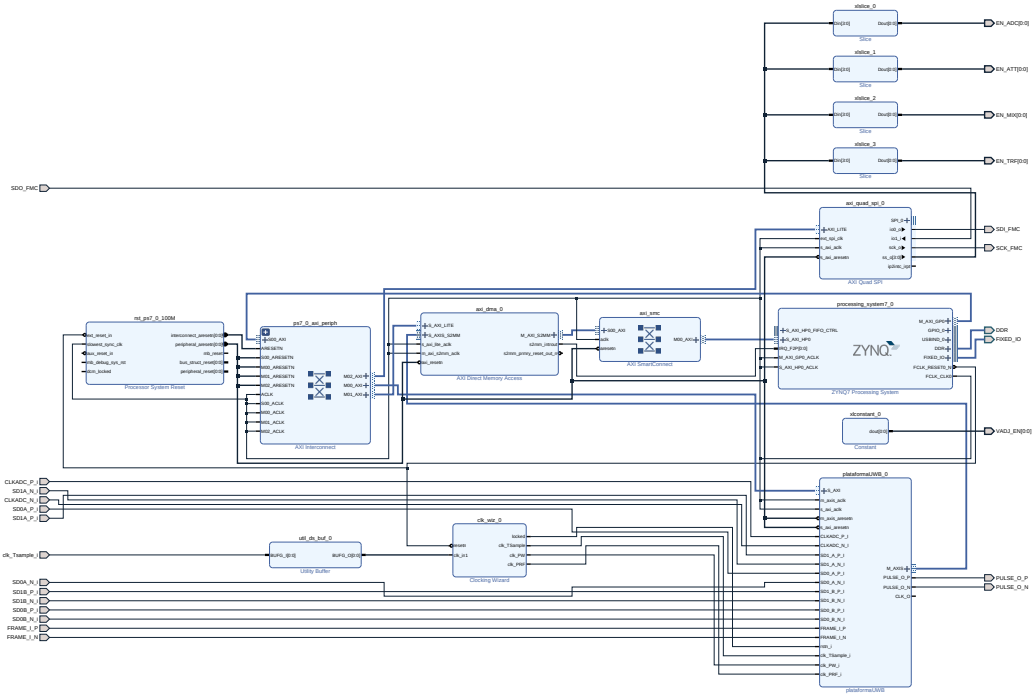


Figura A.1: Arquitectura implementada en el SoC

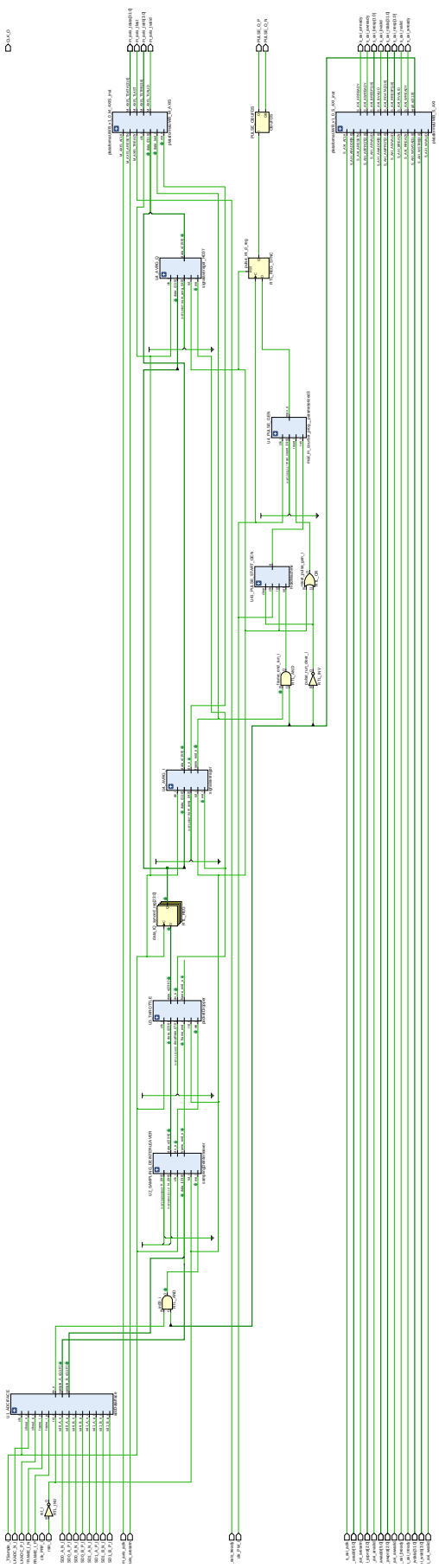


Figura A.2: Arquitectura del IP core `plataformaUWB`

A.2. PCB y Esquemático de la placa receptora

A.2.1. PCB

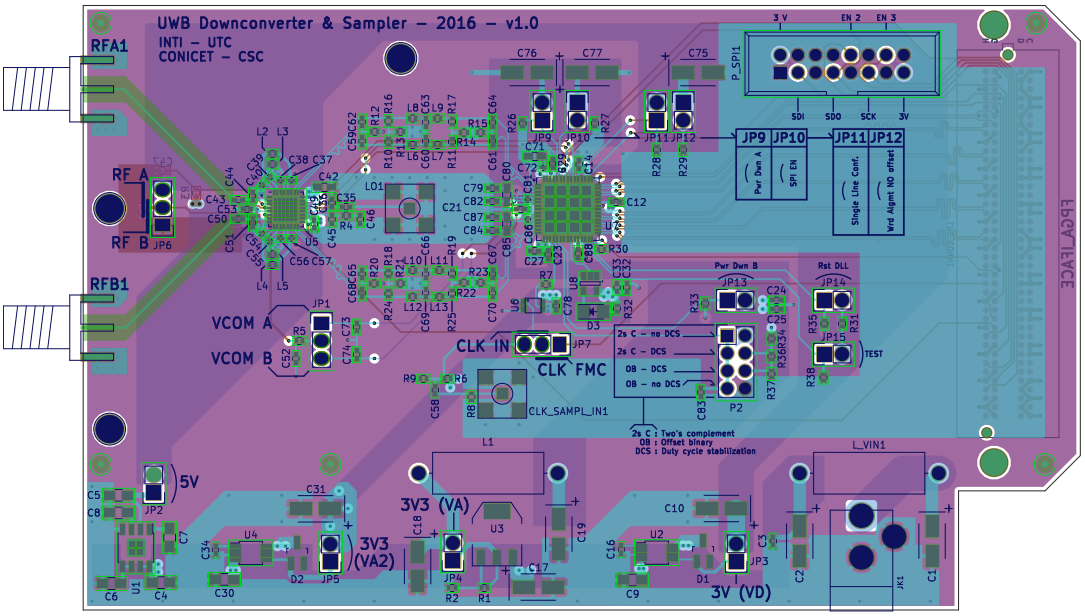
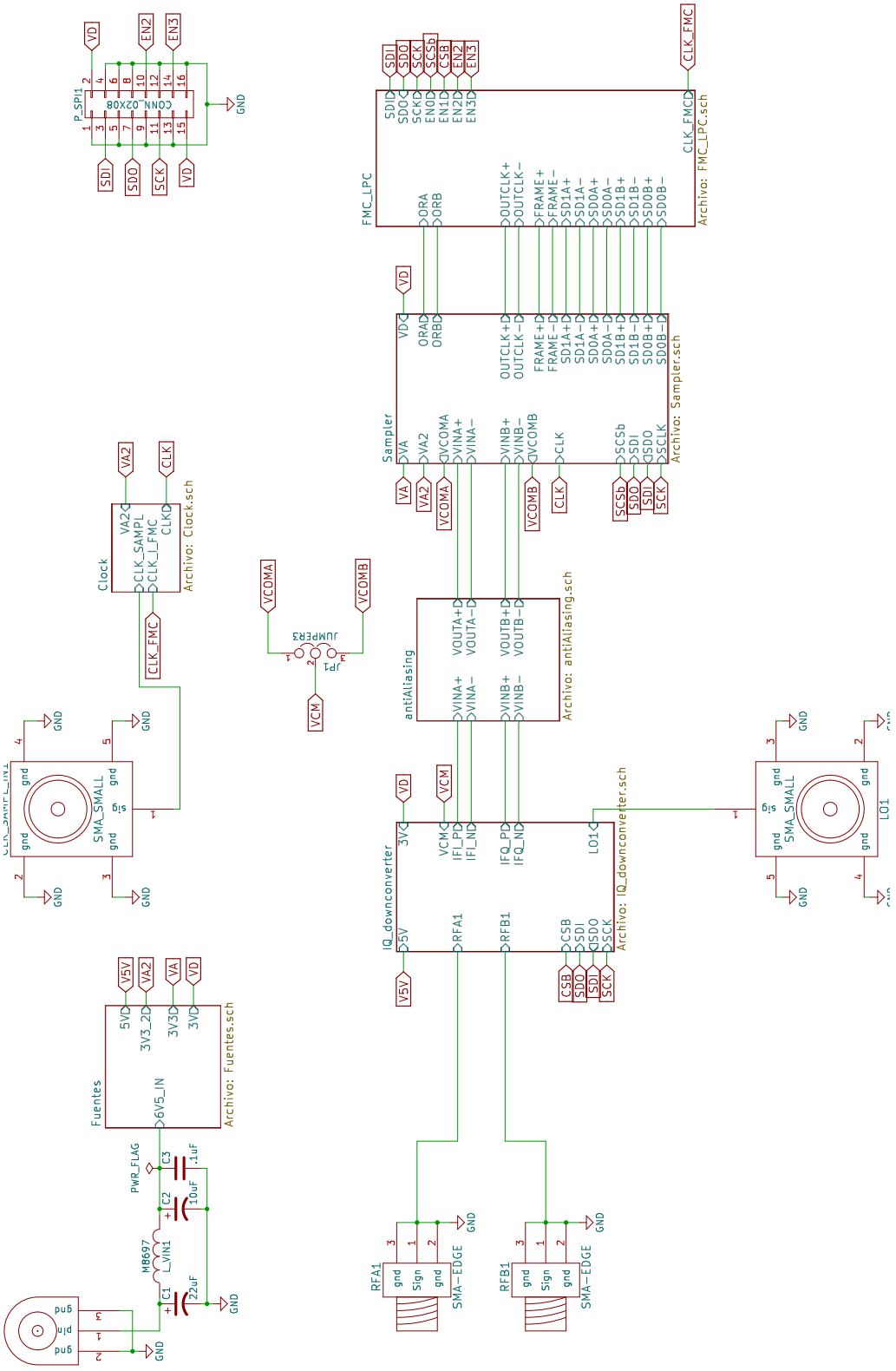


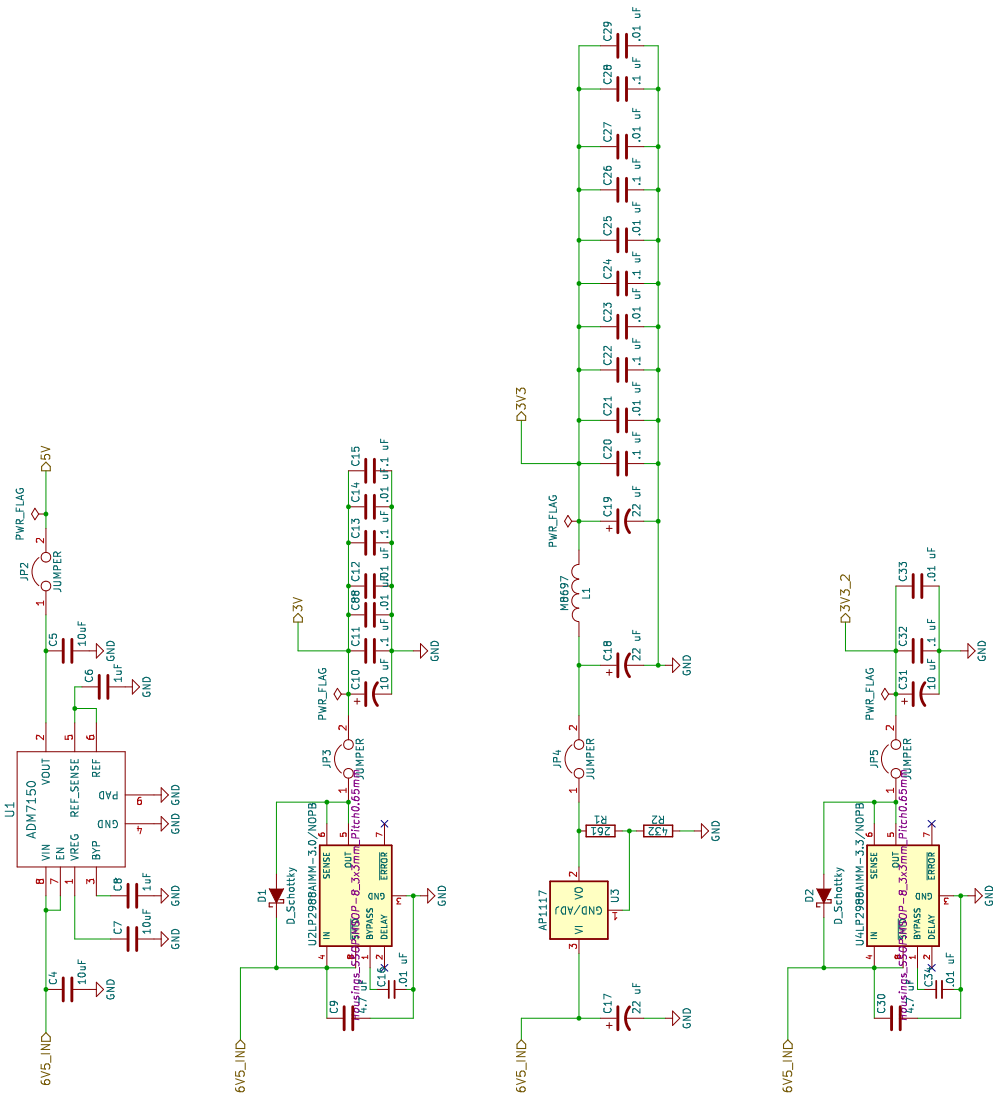
Figura A.3: PCB de la placa receptora

A.2.2. Esquemático

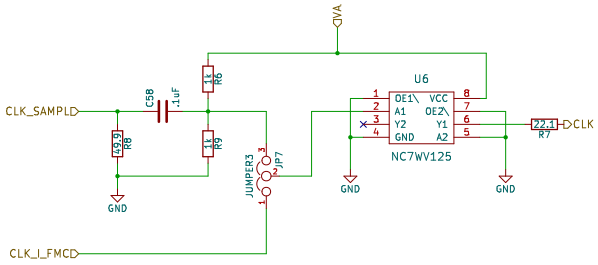
A.2.2.1. Esquema general



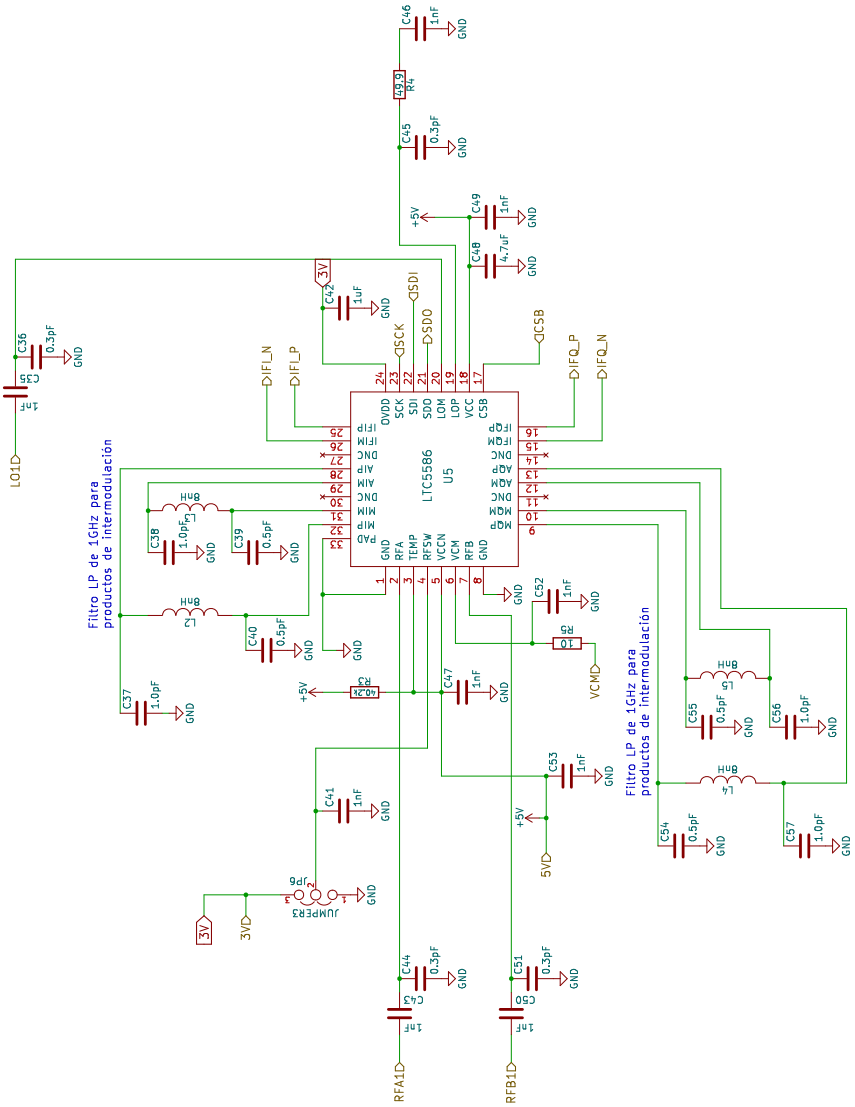
A.2.2.2. Fuentes



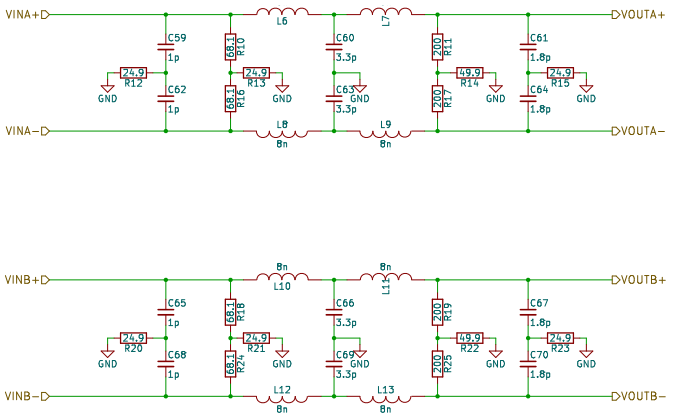
A.2.2.3. Clock



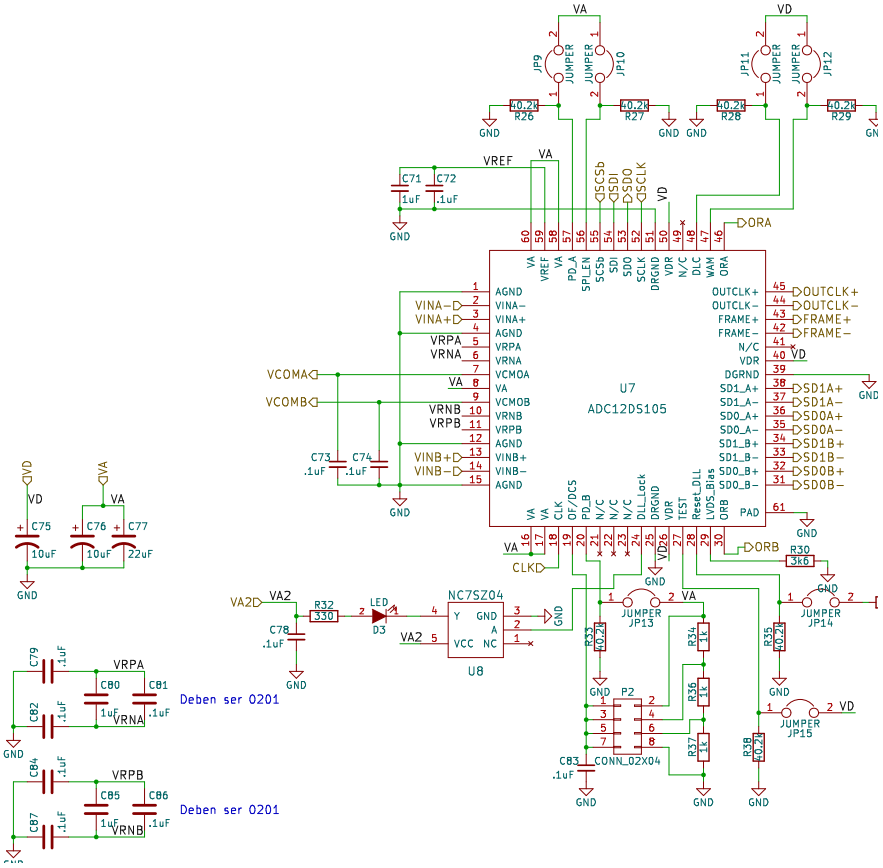
A.2.2.4. IQ Downconverter



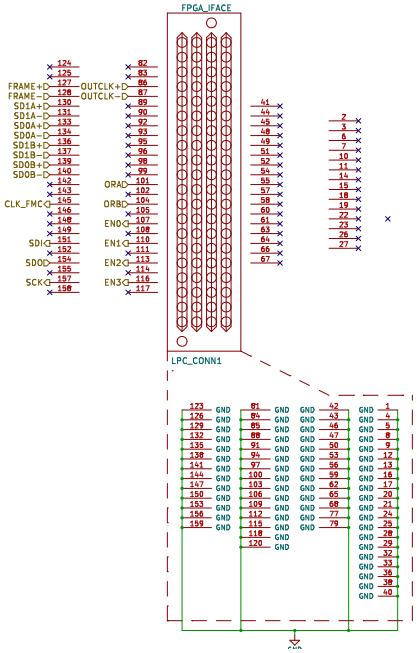
A.2.2.5. Anti-aliasing



A.2.2.6. Sampler



A.2.2.7. FMC LPC



Bibliografía

[1] A. Bourdoux, A. N. Barreto, B. van Liempd, C. de Lima, D. Dardari, D. Belot, E.-S. Lohan, G. Seco-Granados, H. Sarieddeen, H. Wymeersch, J. Suutala, J. Saloranta, M. Guillaud, M. Isomursu, M. Valkama, M. R. K. Aziz, R. Berkvens, T. Sanguanpuak, T. Svensson, and Y. Miao, "6G White Paper on Localization and Sensing," Jun. 2020, arXiv:2006.01779 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2006.01779

[2] ITU, "Characteristics of ultra-wideband technology," ITU-R, Tech. Rep. SM.1755-0, 2006. [Online]. Available: https://www.itu.int/rec/R-REC-SM.1755-0-200605-I

[3] J. D. Taylor, *Ultra-Wideband Radar Technology*. CRC Press, 2000.

[4] G. Giannakis, "Ultra-wideband communications: an idea whose time has come," in *2003 4th IEEE Workshop on Signal Processing Advances in Wireless Communications - SPAWC 2003 (IEEE Cat. No.03EX689)*. Rome, Italy: IEEE, 2003, p. 3. [Online]. Available: http://ieeexplore.ieee.org/document/1318908/

[5] Y. Lee, J.-Y. Park, Y.-W. Choi, H.-K. Park, S.-H. Cho, S. H. Cho, and Y.-H. Lim, "A Novel Non-contact Heart Rate Monitor Using Impulse-Radio Ultra-Wideband (IR-UWB) Radar Technology," *Scientific Reports*, vol. 8, no. 1, p. 13053, Dec. 2018. [Online]. Available: http://www.nature.com/articles/s41598-018-31411-8

[6] J.-Y. Park, Y. Lee, Y.-W. Choi, R. Heo, H.-K. Park, S.-H. Cho, S. H. Cho, and Y.-H. Lim, "Preclinical Evaluation of a Noncontact Simultaneous Monitoring Method for Respiration and Carotid Pulsation Using Impulse-Radio Ultra-Wideband Radar," *Scientific Reports*, vol. 9, no. 1, p. 11892, Dec. 2019. [Online]. Available: http://www.nature.com/articles/s41598-019-48386-9

[7] S. Sesha Vidhya, S. Rukmani Devi, and K. Shanthi, "Human Muscle Mass Measurement through passive Flexible UWB-Myogram Antenna sensor to diagnose Sarcopenia," *Microprocessors and Microsystems*, vol. 79, p. 103284, Nov. 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0141933120304439

[8] E. v. V. Jasper Massey, "Radar Based Human Vital Signs Detection in Cars - State of the Art Analysis," Jul. 2020.

[9] J.-E. Kim, J.-H. Choi, and K.-T. Kim, "Robust Detection of Presence of Individuals in an Indoor Environment Using IR-UWB Radar," *IEEE Access*, vol. 8, pp. 108 133–108 147, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9110845/

[10] M. Anabuki, S. Okumura, T. Sakamoto, K. Saho, T. Sato, M. Yoshioka, K. Inoue, T. Fukuda, and H. Sakai, "High-resolution imaging and separation of multiple pedestrians using UWB Doppler radar interferometry with adaptive beamforming technique," in *2017 11th European Conference on Antennas and Propagation (EUCAP)*. Paris, France: IEEE, Mar. 2017, pp. 469–473. [Online]. Available: http://ieeexplore.ieee.org/document/7928137/

[11] F. M. Sabzevari, R. S. C. Winter, D. Oloumi, and K. Rambabu, "A Microwave Sensing and Imaging Method for Multiphase Flow Metering of Crude Oil Pipes," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 1286–1297, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9022887/

[12] N. Andersen, K. Granhaug, J. A. Michaelsen, S. Bagga, H. A. Hjortland, M. R. Knutsen, T. S. Lande, and D. T. Wisland, "A 118-mw pulse-based radar soc in 55-nm cmos for non-contact human vital signs detection," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 12, pp. 3421–3433, 2017.

[13] B. Van Herbruggen, B. Jooris, J. Rossey, M. Ridolfi, N. Macoir, Q. Van Den Brande, S. Lemey, and E. De Poorter, "Wi-PoS: A Low-Cost, Open Source Ultra-Wideband (UWB) Hardware Platform with Long Range Sub-GHz Backbone," *Sensors*, vol. 19, no. 7, p. 1548, Mar. 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/7/1548

[14] Qorvo, "DW1000 Product Datasheet," accessed: 2023-08-24. [Online]. Available: https://www.qorvo.com/products/d/da007946

[15] Y. Kilic, H. Wymeersch, A. Meijerink, M. J. Bentum, and W. G. Scanlon, "Device-Free Person Detection and Ranging in UWB Networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 8, no. 1, pp. 43–54, Feb. 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6600737/

[16] K.-K. Shyu, L.-J. Chiu, P.-L. Lee, T.-H. Tung, and S.-H. Yang, "Detection of Breathing and Heart Rates in UWB Radar Sensor Data Using FVPIEF-Based Two-Layer EEMD," *IEEE Sensors Journal*, vol. 19, no. 2, pp. 774–784, Jan. 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8515230/

[17] Y. Y. et al., "Micro-vibration distinguishment between humans and animals based on ensemble empirical mode decomposition using ultra-wide band radar.pdf," Sep. 2019. [Online]. Available: doi:10.1049/joe.2019.0619

[18] Novelda, "Novelda uwb x7 dev kit - novelda," 2023, accedido el 20 de septiembre de 2023. [Online]. Available: https://novelda.com/x7-dev-kit

[19] J. L. Hennessy and D. A. Patterson, "A new golden age for computer architecture," *Communications of the ACM*, vol. 62, no. 2, pp. 48–60, Jan. 2019. [Online]. Available: https://dl.acm.org/doi/10.1145/3282307

[20] K. Swaminathan and A. Vega, "Hardware Specialization: From Cell to Heterogeneous Microprocessors *Everywhere*," *IEEE Micro*, vol. 41, no. 6, pp. 112–120, Nov. 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9623446/

[21] E. Marchi, "Adaptivesswt," GitHub repository, 2023. [Online]. Available: https://github.com/edgardomarchi/adaptivesswt

[22] E. Marchi, M. Cervetto, and C. Galarza, "Adaptive synchrosqueezing wavelet transform for real-time applications," *Digital Signal Processing*, vol. 140, p. 104133, Aug. 2023. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1051200423002282

[23] R. K. M. Ghavami, L. B. Michael, *Introduction*. John Wiley & Sons, Ltd, 2007, pp. 1–8. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470060490.ch

[24] J. D. Taylor, *Introduction to Ultra-Wideband Radar Systems*. CRC Press, 1995.

[25] L. Felsen, R. Mittra, C. Baum, D. Sengupta, C. Tai, J. Fuller, and J. Wait, *Transient Electromagnetic Fields*, ser. Topics in Applied Physics. Springer Berlin Heidelberg, 2006. [Online]. Available: https://books.google.com.ar/books?id=RYmQDAAAQBAJ

[26] J. R. Auton and M. L. V. Blaricum, "Investigation of procedures for automatic resonance extraction from noisy transient electromagnetics data. volume ii. appendices," in *Investigation of Resonance Extraction Procedures*, 1981. [Online]. Available: https://api.semanticscholar.org/CorpusID:58832222

[27] Y. Hua and T. Sarkar, "Matrix pencil method for estimating parameters of exponentially damped/undamped sinusoids in noise," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, no. 5, pp. 814–824, 1990.

[28] R. Roy and T. Kailath, "Esprit-estimation of signal parameters via rotational invariance techniques," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 7, pp. 984–995, July 1989.

[29] V. Kabourek, P. Černý, and M. Mazanek, "Clutter reduction based on principal component analysis technique for hidden objects detection," *Radioengineering*, vol. 21, 04 2012.

[30] S. Madhavan, R. K. Tripathy, and R. B. Pachori, "Time-frequency domain deep convolutional neural network for the classification of focal and non-focal eeg signals," *IEEE Sensors Journal*, vol. 20, no. 6, pp. 3078–3086, 2020.

[31] G. Thakur, E. Brevdo, N. S. Fučkar, and H.-T. Wu, "The Synchrosqueezing algorithm for time-varying spectral analysis: Robustness properties and new paleoclimate applications," *Signal Processing*, vol. 93, no. 5, pp. 1079–1094, May 2013. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0165168412004240

[32] A. Bianchetti, F. E. Veiras, P. Etchepareborda, A. L. Vadnjal, A. Federico, and G. H. Kaufmann, "Amplitude and phase retrieval in simultaneous $\pi/2$ phase-shifting heterodyne interferometry using the synchrosqueezing transform," *Appl. Opt.*, vol. 54, no. 8, p. 2132–2140, Mar 2015. [Online]. Available: http://opg.optica.org/ao/abstract.cfm?URI=ao-54-8-2132

[33] X. Tu, W. Bao, Y. Hu, S. Abbas, and F. Li, "Parameterized synchrosqueezing transform with application to machine fault diagnosis," *IEEE Sensors Journal*, vol. 19, no. 18, pp. 8107–8115, 2019.

[34] I. Daubechies, J. Lu, and H.-T. Wu, "Synchrosqueezed wavelet transforms: An empirical mode decomposition-like tool," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 243–261, Mar. 2011. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1063520310001016

[35] Ping Wang, Jinghuai Gao, and Zhiguo Wang, "Time-Frequency Analysis of Seismic Data Using Synchrosqueezing Transform," *IEEE Geoscience and Remote Sensing Letters*, vol. 11, no. 12, pp. 2042–2044, Dec. 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6809966/

[36] H.-T. Wu, Y.-H. Chan, Y.-T. Lin, and Y.-H. Yeh, "Using synchrosqueezing transform to discover breathing dynamics from ECG signals," *Applied and Computational Harmonic Analysis*, vol. 36, no. 2, pp. 354–359, Mar. 2014. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1063520313000663

[37] H. Cao, Y. Yue, X. Chen, and X. Zhang, "Chatter detection based on synchrosqueezing transform and statistical indicators in milling process," *The International Journal of Advanced Manufacturing Technology*, vol. 95, no. 1-4, pp. 961–972, Mar. 2018. [Online]. Available: http://link.springer.com/10.1007/s00170-017-1283-0

[38] Y. Imaouchen, R. Alkama, and M. Thomas, "Complexity based on synchrosqueezing analysis in gear diagnosis," *Mechanics & Industry*, vol. 16, no. 5, p. 508, 2015. [Online]. Available: http://www.mechanics-industry.org/10.1051/meca/2015026

[39] I. Daubechies, *Ten lectures on wavelets*, ser. CBMS-NSF regional conference series in applied mathematics. Philadelphia, Pa: Society for Industrial and Applied Mathematics, 1992, no. 61.

[40] A. Berrian and N. Saito, "Adaptive synchrosqueezing based on a quilted short-time Fourier transform," *arXiv:1707.03138 [cs, math]*, Sep. 2017, arXiv: 1707.03138. [Online]. Available: http://arxiv.org/abs/1707.03138

[41] T. Oberlin, S. Meignen, and V. Perrier, "The fourier-based synchrosqueezing transform," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Florence, Italy: IEEE, May 2014, pp. 315–319. [Online]. Available: http://ieeexplore.ieee.org/document/6853609/

[42] Z. Wu, T. Lan, C. Yang, and Z. Nie, "A Novel Method to Detect Multiple Arrhythmias Based on Time-Frequency Analysis and Convolutional Neural Networks," *IEEE Access*, vol. 7, pp. 170 820–170 830, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8913462/

[43] V. Sucic, N. Saulig, and B. Boashash, "Estimating the number of components of a multicomponent nonstationary signal using the short-term time-frequency Rényi entropy," *EURASIP Journal on Advances in Signal Processing*, vol. 2011, no. 1, p. 125, Dec. 2011. [Online]. Available: https://asp-eurasipjournals.springeropen.com/articles/10.1186/1687-6180-2011-125

[44] R. Baraniuk, P. Flandrin, A. Janssen, and O. Michel, "Measuring time-frequency information content using the Renyi entropies," *IEEE Transactions on Information Theory*, vol. 47, no. 4, pp. 1391–1409, May 2001. [Online]. Available: http://ieeexplore.ieee.org/document/923723/

[45] S. Meignen, D.-H. Pham, and S. McLaughlin, "On Demodulation, Ridge Detection, and Synchrosqueezing for Multicomponent Signals," *IEEE Transactions on Signal Processing*, vol. 65, no. 8, pp. 2093–2103, Apr. 2017. [Online]. Available: https://ieeexplore.ieee.org/document/7829418/

[46] S. Meignen, D.-H. Pham, and M. A. Colominas, "On the use of short-time fourier transform and synchrosqueezing-based demodulation for the retrieval of the modes of multicomponent signals," *Signal Processing*, vol. 178, p. 107760, Jan. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0165168420303030

[47] T. Oberlin, S. Meignen, and V. Perrier, "Second-Order Synchrosqueezing Transform or Invertible Reassignment? Towards Ideal Time-Frequency Representations," *IEEE Transactions on Signal Processing*, vol. 63, no. 5, pp. 1335–1344, Mar. 2015. [Online]. Available: http://ieeexplore.ieee.org/document/7006749/

[48] S. Meignen, T. Oberlin, and D.-H. Pham, "Synchrosqueezing transforms: From low- to high-frequency modulations and perspectives," *Comptes Rendus Physique*, vol. 20, no. 5,

pp. 449–460, Jul. 2019. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S163107051930101X

[49] R. Behera, S. Meignen, and T. Oberlin, "Theoretical analysis of the second-order synchrosqueezing transform," *Applied and Computational Harmonic Analysis*, vol. 45, no. 2, pp. 379–404, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1063520316300720

[50] H. Dong, G. Yu, and Y. Li, "Theoretical analysis and comparison of transient-extracting transform and time-reassigned synchrosqueezing transform," *Mechanical Systems and Signal Processing*, vol. 178, p. 109190, Oct. 2022. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0888327022003478

[51] G. Yu, M. Yu, and C. Xu, "Synchroextracting Transform," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 10, pp. 8042–8054, Oct. 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7906573/

[52] S. Lv, Y. Lv, R. Yuan, and H. Li, "High-order synchroextracting transform for characterizing signals with strong AM-FM features and its application in mechanical fault diagnosis," *Mechanical Systems and Signal Processing*, vol. 172, p. 108959, Jun. 2022. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0888327022001406

[53] D. Iatsenko, P. V. McClintock, and A. Stefanovska, "Linear and synchrosqueezed time–frequency representations revisited: Overview, standards of use, resolution, reconstruction, concentration, and algorithms," *Digital Signal Processing*, vol. 42, pp. 1–26, Jul. 2015. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1051200415000792

[54] L. Li, H. Cai, and Q. Jiang, "Adaptive synchrosqueezing transform with a time-varying parameter for non-stationary signal separation," *Applied and Computational Harmonic Analysis*, vol. 49, no. 3, pp. 1075–1106, Nov. 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S106352031830126X

[55] J. Lu, Q. Jiang, and L. Li, "Analysis of adaptive synchrosqueezing transform with a time-varying parameter," *Advances in Computational Mathematics*, vol. 46, 08 2020.

[56] L. Li, H. Y. Cai, and Q. Jiang. [Online]. Available: http://www.math.umsl.edu/~jiang/Jsoftware_SST.htm

[57] G. Yu, Z. Wang, and P. Zhao, "Multisynchrosqueezing Transform," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 7, pp. 5441–5455, Jul. 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8458385/

[58] F. Khan, A. Ghaffar, N. Khan, and S. H. Cho, "An Overview of Signal Processing Techniques for Remote Health Monitoring Using Impulse Radio UWB Transceiver," *Sensors*, vol. 20, no. 9, p. 2479, Apr. 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/9/2479

[59] E. Marchi, M. Cervetto, and C. Galarza. [Online]. Available: https://github.com/edgardomarchi/adaptivesswt

[60] Y. Liu, X. San Liang, and R. H. Weisberg, "Rectification of the Bias in the Wavelet Power Spectrum," *Journal of Atmospheric and Oceanic Technology*, vol. 24, no. 12, pp. 2093–2102, Dec. 2007. [Online]. Available: http://journals.ametsoc.org/doi/10.1175/2007JTECHO511.1

[61] F. Pukelsheim, *Proportional Representation: Apportionment Methods and Their Applications*, 2nd ed. Springer International Publishing, 2017, pp. 114–116.

[62] K. Schneider and M. Farge, "Wavelets: Mathematical Theory," in *Encyclopedia of Mathematical Physics*. Elsevier, 2006, pp. 426–438. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/B012512666200153X

[63] P. S. Foundation, "Python package index (pypi)," 2024, accessed: 30-Jan-2024. [Online]. Available: https://pypi.org/

[64] R. P. Ltd., "Micropython - raspberry pi documentation," accedido: 28 ene. 2024. [Online]. Available: https://www.raspberrypi.com/documentation/microcontrollers/micropython.html

[65] A. (Xilinx), "Pynq - python productivity for zynq," accedido: 28 ene. 2024. [Online]. Available: https://www.pynq.io/

[66] A. C. CERN, "Python packages for hpc," 2024, accessed: 30-Jan-2024. [Online]. Available: https://cern.ch/abpcomputing/guides/hpc_python/

[67] R. C. Princeton, "Python on the research computing clusters," 2024, accessed: 30-Jan-2024. [Online]. Available: https://researchcomputing.princeton.edu/support/knowledge-base/python

[68] H. G. NIH, "Python in hpc," 2024, accessed: 30-Jan-2024. [Online]. Available: https://hpc.nih.gov/training/handouts/171121_python_in_hpc.pdf

[69] P. S. Foundation, "Python/c api reference manual," 2024, accessed: 30-Jan-2024. [Online]. Available: https://docs.python.org/3/c-api/stable.html

[70] PYPL, "Pypl popularity of programming language index for january 2024," accedido: 28 ene. 2024. [Online]. Available: https://pypl.github.io/PYPL.html

[71] TIOBE, "Tiobe index for january 2024," accedido: 28 ene. 2024. [Online]. Available: https://www.tiobe.com/tiobe-index/

[72] G. Lee, R. Gommers, F. Waselewski, K. Wohlfahrt, and A. O'Leary, "PyWavelets: A Python package for wavelet analysis," *Journal of Open Source Software*, vol. 4, no. 36, p. 1237, Apr. 2019. [Online]. Available: http://joss.theoj.org/papers/10.21105/joss.01237

[73] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2

[74] T. Barrett, "History of ultrawideband (uwb) radar & communications: Pioneers and innovators," *Conference: Progress in Electromagnetics Symposium 2000 (PIERS2000)*, 07 2000.

[75] G. Ross, "Transmission and reception system for generating and receiving base-band pulse duration pulse signals without distortion for short base-band communication system," Apr. 17 1973, uS Patent 3,728,632.

[76] J. Ryckaert, C. Desset, A. Fort, M. Badaroglu, V. De Heyn, P. Wambacq, G. Van Der Plas, S. Donnay, B. Van Poucke, and B. Gyselinckx, "Ultra-wide-band transmitter for low-power wireless body area networks: design and evaluation," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 52, no. 12, pp. 2515–2525, Dec. 2005. [Online]. Available: http://ieeexplore.ieee.org/document/1556760/

[77] A. De Angelis, M. Dionigi, A. Moschitta, and P. Carbone, "A Low-Cost Ultra-Wideband Indoor Ranging System," *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 12, pp. 3935–3942, Dec. 2009. [Online]. Available: http://ieeexplore.ieee.org/document/5306095/

[78] E. J. Colli-Vignarelli, "Design of a Discrete-Component Impulse-Radio Ultra Wide-Band (IR-UWB) Testbed and Design of a very Low-Power IR-UWB Transmitter in CMOS Technology," *EPFL Infoscience*, Feb. 2012, publisher: Lausanne, EPFL. [Online]. Available: http://infoscience.epfl.ch/record/174674

[79] Q. Liu, Y. Wang, and A. E. Fathy, "A compact integrated 100 GS/s sampling module for UWB see through wall radar with fast refresh rate for dynamic real time imaging," in *2012 IEEE Radio and Wireless Symposium*. Santa Clara, CA, USA: IEEE, Jan. 2012, pp. 59–62. [Online]. Available: http://ieeexplore.ieee.org/document/6175295/

[80] *IEEE Standard for Local and metropolitan area networks–Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)*, Institute of Electrical and Electronics Engineers (IEEE), 2011. [Online]. Available: https://standards.ieee.org/standard/802_15_4-2011.html

[81] *P440 Ultra Wideband (UWB) Radio Transceiver User Manual*, Humatics, 2021. [Online]. Available: https://cdn2.hubspot.net/hubfs/4072898/320-0317F%20P440%20Data%20Sheet%20-%20User%20Guide-1.pdf

[82] "Tdsr uwb module," https://web.archive.org/web/20221222010141/https://tdsr-uwb.com/uwb-module/, accedido el: 29 de Septiembre de 2023.

[83] Novelda, "XM06 Datasheet," accessed: 2023-08-30. [Online]. Available: https://laonuri.techyneeti.com/wp-content/uploads/2019/02/X4M06_DATASHEET.pdf

[84] Novelda, "X4SIP02 Datasheet," accessed: 2023-08-24. [Online]. Available: https://github.com/novelda/Legacy-Documentation/blob/master/Datasheets/X4SIP02_X4_system_in_package_rev_b_preliminary.pdf

[85] Novelda, "XTMCU02 Datasheet," accessed: 2023-08-24. [Online]. Available: https://github.com/novelda/Legacy-Documentation/blob/master/Datasheets/XTMCU02_mcu_board_rev_b_preliminary.pdf

[86] Novelda, "XM05 Datasheet," accessed: 2023-08-24. [Online]. Available: https://github.com/novelda/Legacy-Documentation/blob/master/Datasheets/X4M05_radar_sensor_module_rev_c_preliminary.pdf

[87] Novelda, "XA02 Datasheet," accessed: 2023-08-24. [Online]. Available: https://github.com/novelda/Legacy-Documentation/blob/master/Datasheets/X4A02_antenna_board_rev_b_preliminary.pdf

[88] P. CIAA, "Computadora Industrial Abierta para aplicaciones de Alto Costo Computacional," 2023, accedido el 03 de Octubre de 2023. [Online]. Available: http://www.proyecto-ciaa.com.ar/devwiki/doku.php?id=desarrollo:ciaa_acc:ciaa_acc_inicio

[89] P. Gámez, E. Marchi, M. Cervetto, C. Giuffrida, G. Perez, A. Altieri, and C. Galarza, "A low-cost ultra-wideband test-bed for dielectric target detection," in *2017 XVII Workshop on Information Processing and Control (RPIC)*, 2017, pp. 1–6.

[90] M. B. A. Altieri; P. Gámez; E. Marchi; M. Cervetto and C. G. Galarza, "A low-cost ultra-wideband test-bed," in *XXXV Simpósio Brasileiro de Telecomunicações e Processamento de Sinais (SBrT 2017)*, 2017.

[91] T. Instruments, "TRF372017 300 MHz to 4.8GHz Quadrature Modulator with integrated wideband PLL/VCO," 2023, accessed: Oct. 4, 2023. [Online]. Available: https://www.ti.com/lit/gpn/trf372017

[92] A. Devices, "30 MHz to 6 GHz RF/IF Gain Block Data Sheet ADL5545," 2023, accessed: Oct. 4, 2023. [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/ADL5545.PDF

[93] pSemi Corporation, "50 ohm, HaRP™-enhanced, High Linearity RF Digital Step Attenuator PE43712," 2023, accessed: Oct. 4, 2023. [Online]. Available: https://www.psemi.com/pdf/datasheets/pe43712ds.pdf

[94] A. Devices, "LTC5586 - 6GHz High Linearity I/Q Demodulator with Wideband IF Amplifier," Analog Devices, Tech. Rep., 2018. [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/LTC5586.pdf

[95] T. Instruments, "ADC12DS105 Dual 12-Bit, 105 MSPS A/D Converter with Serial LVDS Outputs," Texas Instruments, Tech. Rep., 2013. [Online]. Available: https://www.ti.com/lit/ds/symlink/adc12ds105.pdf

[96] Isola, "FR408 - High Performance Laminate and Prepreg." [Online]. Available: https://www.isola-group.com/pcb-laminates-prepreg/fr408-laminate-and-prepreg/

[97] O. Park, "4 Layer Prototype Service." [Online]. Available: https://docs.oshpark.com/services/four-layer/

[98] Xilinx, "7 series selectio user guide." [Online]. Available: https://docs.xilinx.com/v/u/en-US/ug471_7Series_SelectIO

[99] A. W. Services, "Freertos - rtos (real time operating system) for embedded systems with internet of things extensions," accessed: Oct. 4, 2023. [Online]. Available: https://www.freertos.org/

[100] K. Shi, S. Schellenberger, C. Will, T. Steigleder, F. Michler, J. Fuchs, R. Weigel, C. Ostgathe, and A. Koelpin, "A dataset of radar-recorded heart sounds and vital signs including synchronised reference sensor signals," *Scientific Data*, vol. 7, no. 1, p. 50, Dec. 2020. [Online]. Available: http://www.nature.com/articles/s41597-020-0390-1

[101] T. G. Kolda and B. W. Bader, "Tensor Decompositions and Applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009.

[102] R. Albert and C. Galarza, "Spectrum estimation using frequency shifting and decimation," *IET Signal Processing*, vol. 14, no. 3, pp. 134–141, 2020. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/iet-spr.2019.0306

[103] J.-M. Papy, L. D. Lathauwer, and S. V. Huffel, "A Shift Invariance-Based Order-Selection Technique for Exponential Data Modelling," *IEEE Signal Processing Letters*, vol. 14, no. 7, pp. 473–476, 2007.

[104] S. Kolouri, S. R. Park, and G. K. Rohde, "The Radon Cumulative Distribution Transform and Its Application to Image Classification," *IEEE Transactions on Image Processing*, vol. 25, no. 2, pp. 920–934, 2016.

[105] M. He, Y. Nian, L. Xu, L. Qiao, and W. Wang, "Adaptive Separation of Respiratory and Heartbeat Signals among Multiple People Based on Empirical Wavelet Transform Using UWB Radar," *Sensors*, vol. 20(17), no. 4913, p. 17, 2020.

[106] M. Cervetto, E. Marchi, and C. G. Galarza, "A Fully Configurable SoC-Based IR-UWB Platform for Data Acquisition and Algorithm Testing," *IEEE Embedded Systems Letters*, vol. 13, no. 2, pp. 53–56, Jun. 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9099878/

Agradecimientos

"Muchas veces, a lo largo de un mismo día,
me doy cuenta que mi propia vida y sus logros
se han construido gracias al trabajo
de las personas que me rodean.
También comprendo, con cuanta seriedad
debo esforzarme para darles, en
correspondencia,
tanto como he recibido."

Albert Einstein

Agradezco enormemente a todas las personas e instituciones que han contribuido al desarrollo y culminación de esta tesis doctoral.

En primer lugar, agradezco al INTI por el espacio y la remuneración por el trabajo realizado y a FIUBA por la formación, el acompañamiento y la posibilidad de transmitir lo aprendido como docente. Agradezco también al CSC-CONICET por formar parte fundamental del proyecto y uno de los motores del mismo.

A mi directora de tesis, Cecilia G. Galarza, le extiendo mi más profundo agradecimiento por su invaluable guía, paciencia y dedicación. Particularmente nunca dejó de guiarme a pesar de pertenecer a una institución diferente y por ello tener que adaptarse a objetivos diversos a lo largo del desarrollo del proyecto.

A mi grupo de trabajo, amigos y colegas, gracias por su colaboración, apoyo constante y por crear un ambiente de trabajo enriquecedor y motivador. Sus ideas y esfuerzos han sido cruciales para el éxito de esta investigación. A todas y todos los investigadores que participaron de este proyecto macro, gracias por su aporte y por compartir siempre sus conocimientos y experiencias.

Finalmente, a mi familia, mi compañera de vida, mis padres y hermanos, les agradezco por su amor incondicional, la ayuda, el apoyo constante y por creer en mí en todo momento. Su comprensión y sacrificio han sido una fuente de inspiración y fortaleza.

A todos, mi más sincero agradecimiento.