Grado en Ingeniería en Telecomunicaciones

Trabajo Fin de Grado

Diseño, pruebas y simulación de un RADAR pasivo en la banda DVB-T2

Autor
Alejandro Manuel López Gómez

Directores
Carlos García de la Cueva
Javier Matanza Domingo

Madrid
Junio 2023

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Estudio, simulación y pruebas de un radar pasivo en

la banda DVB-T2

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el curso académico **2022/2023** es de mi autoría, original e inédito y no ha sido presentado con anterioridad a otros efectos. El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.

Fdo.: Alejandro Manuel López Gómez Fecha: 26/06/2023

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: Carlos García de la Cueva Fecha: 26/06/2023

Firmado por GARCIA DE LA CUEVA CARLOS - ***8797** el día 27/06/2023 con un certificado emitido por AC FNMT Usuarios

Fdo.: Javier Matanza Domingo Fecha: 26/06/2023

Firmado por MATANZA DOMINGO JAVIER - ***4397** el día 26/06/2023 con un certificado emitido por AC FNMT Usuarios

Grado en Ingeniería en Telecomunicaciones

Trabajo Fin de Grado

Diseño, pruebas y simulación de un RADAR pasivo en la banda DVB-T2

Autor
Alejandro Manuel López Gómez

Directores
Carlos García de la Cueva
Javier Matanza Domingo

Madrid
Junio 2023

A mi madre y a mi hermana

Índice general

Índice de figuras

Resumen ejecutivo del proyecto

Diseño, pruebas y simulación de un RADAR pasivo en la banda DVB-T2

AUTOR López Gómez, Alejandro Manuel.
DIRECTOR García de la Cueva, Carlos.
DIRECTOR Matanza Domingo, Javier.
ENTIDAD COLABORADORA ICAI – Universidad Pontificia Comillas.

Resumen

Introducción

Tradicionalmente, los radares activos emiten señales de radio y detectan los objetos en función del eco de las señales reflejadas por los objetos en su camino. Sin embargo, los radares pasivos, que aprovechan las señales existentes en el entorno sin emitir señales propias, han ganado interés en las últimas décadas debido a sus ventajas potenciales en términos de economía de energía, seguridad y capacidad de detección encubierta.

DVB-T (Digital Video Broadcasting - Terrestrial) es uno de los estándares más comunes de modulación digital para señales de televisión digital terrestre. Debido a su uso tan extendido, además de otros factores, las señales DVB-T son comúnmente empleadas como iluminadores de oportunidad en sistemas de RADAR pasivo.

Fruto de la popularidad del estándar, en 2006 el grupo DVB (responsable de la creación de DVB-T) decidió estudiar opciones para el diseño de un estándar DVB-T actualizado. Estos estudios culminaron con la creación de DVB-T2 (Digital Video

Broadcasting - Second Generation Terrestrial). La principal ventaja de este nuevo estándar respecto a su predecesor es un uso más eficiente del ancho de banda, en el siguiente apartado se profundizará más en sus diferencias.

Debido a la novedad de este estándar y su complejidad añadida no existe un trabajo tan extenso como para DVB-T en lo que a procesado RADAR se refiere. El principal objetivo de este trabajo es, a partir de una cadena de procesado RADAR para DVB-T diseñada previamente, realizar las modificaciones necesarias para trabajar con DVB-T2. Una vez obtenidos los algoritmos de DVB-T2 a partir de los de DVB-T, se diseñarán las fases de detección, tracking y predicción del objetivo. Finalmente una vez terminado el diseño, se realizarán simulaciones del proyecto final.

Como particularidad del diseño, se ha optado por emplear un único canal de recepción para minimizar los costes del sistema, lo que implica que será necesaria la separación de las señales de referencia, de eco y ruido, aumentando la complejidad del procesado. Este trabajo parte de un proyecto anterior sobre el que se realizarán diferentes modificaciones.

Metodología de trabajo

Este trabajo se centró en primer lugar en el estudio del análisis del trabajo previo. El objetivo de esta fase de estudio fue el aprendizaje de las fases de procesado RADAR, la disección del código planteado y el planteamiento de posibles mejoras sobre este. Una vez concluida esta fase del proyecto, se procedió a la implementación de las mejoras realizadas o al rediseño de aquellas partes que lo requiriesen, además de la adición de aquellos elementos de procesado previstos. Finalmente se procedió a la realización de simulaciones sobre las diferentes partes de la cadena de procesado y a la justificación de los valores escogidos aportando la evidencia pertinente.

Respecto al seguimiento del trabajo, se organizaron sesiones semanales de seguimiento cuyo objetivo fue presentar los avances realizados por el alumno y recibir retroalimentación sobre estos. A continuación en la siguiente tabla se muestra la planificación previa del proyecto.

	2022				2023				
OBJETIVO	SEPTIEMBRE	OCTUBRE	NOVIEMBRE	DICIEMBRE	ENERO	FEBRERO	MARZO	ABRIL	MAYO
Estudio y análisis RADAR pasivo DVB-T.									
Modificaciones para trabajar con DVB-T2.									
Diseño de las etapas de detección y tracking.									
Simulaciones cadena de procesado.									
Estudio de sensibilidad, ajustes y pruebas de campo.									

Figura 1: Esquema organización

Desafortunadamente se ha tenido que excluir la parte dedicada a hardware del proyecto, por lo que no se han realizado pruebas de campo o estudios de sensibilidad del hardware. En su lugar se ha optado por la adición a la cadena de procesado una nueva fase de especial interés en la tecnología RADAR, en concreto se trata de la fase de predicción de posiciones a partir del filtrado de Kalman.

Resultados

En esta sección se discutirá de forma breve los resultados obtenidos en el proyecto. Se comentarán las mejoras realizadas o cambios importantes realizados en las diferentes fases del procesado.

Función de ambigüedad cruzada

La función de ambigüedad cruzada (CAF) es una parte fundamental del procesado RADAR. A partir de las señales de referencia y eco, se realiza una operación de correlación. Se aplican progresivamente diferentes valores de retardo muestral y desplazamiento en frecuencia a la señal de referencia. El objetivo es lograr encontrar el par de valores retardo desplazamiento con un alto valor de correlación. Cuando se de esta situación se considera que se ha obtenido una detección [1].

A continuación se muestra el diagrama de bloques del algoritmo de Batch, una implementación práctica de esta función. Consiste en la separación de la señal a procesar en bloques de igual longitud con el objetivo de aumentar la velocidad de procesado. En este caso ha sido necesario añadir una primera fase de interpolación debido a la frecuencia de muestreo de la señal entrante y una fase de filtrado para mejor tratamiento del ruido.

El resultado de este algoritmo se observa en la misma figura, se trata de un mapa que relaciona distancia y velocidad biestáticas de correlación cuyos picos (simbolizados en color rojo) denotan posibles detecciones.

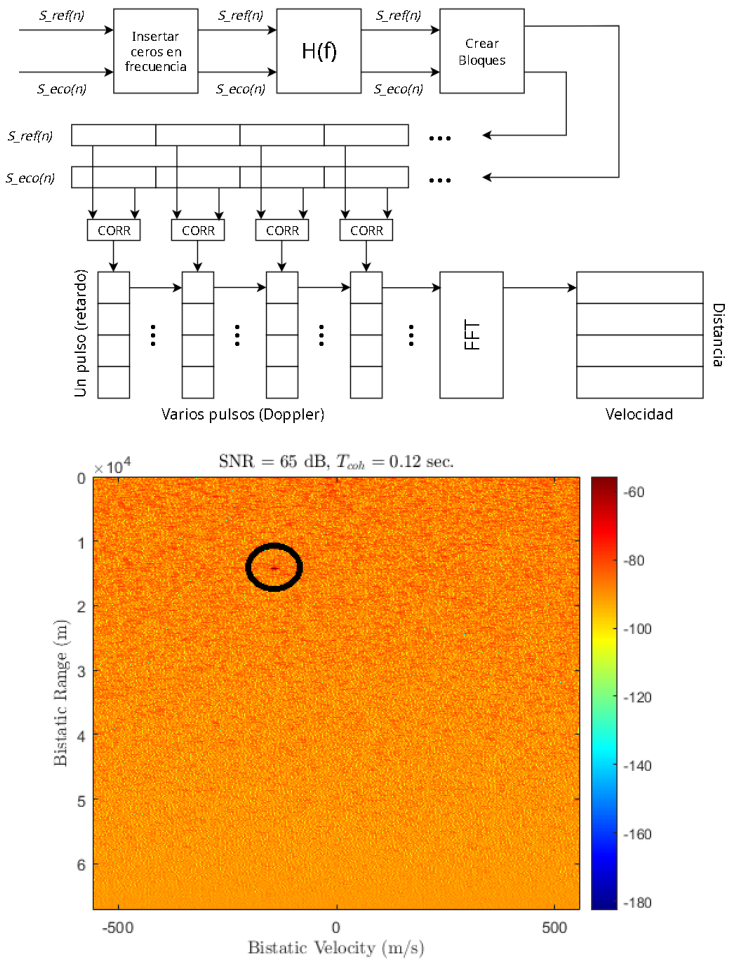


Figura 2: Algoritmo de Batch

Estimación gruesa de retardo de símbolo

El estimador grueso de retardo de símbolo ó estimador de símbolo grueso es un elemento fundamental previo al estimador de canal cuya función es solucionar cualquier problema inicial de sincronización temporal presente en la señal. El objetivo de esta fase del procesado es la sincronización temporal de la señal de entrada, eliminando posibles retardos muestrales que esta pudiese tener. La salida de esta fase de procesado es un índice muestral mediante el cuál se informa al estimador de canal donde comienza la señal DVB-T. Solamente se ejecuta una única vez para cada señal DVB-T.

Estimación de canal

En el diseño del receptor planteado solamente se contempla un único canal de recepción, por lo que las señales eco y referencia se encuentran mezcladas en una única señal recibida. Esta fase de procesado tiene la importante función de correctamente diferenciar y separar las señales anteriormente mencionadas.

El algoritmo diseñado de estimación de canal se muestra en el siguiente diagrama de bloques. En rojo y verde se muestran las memorias de los símbolos y respuestas al canal respectivamente. Las dos memorias utilizadas en este algoritmo permiten una correlación y promediado con respuestas al canal anteriores, aumentando la precisión de la estimación.

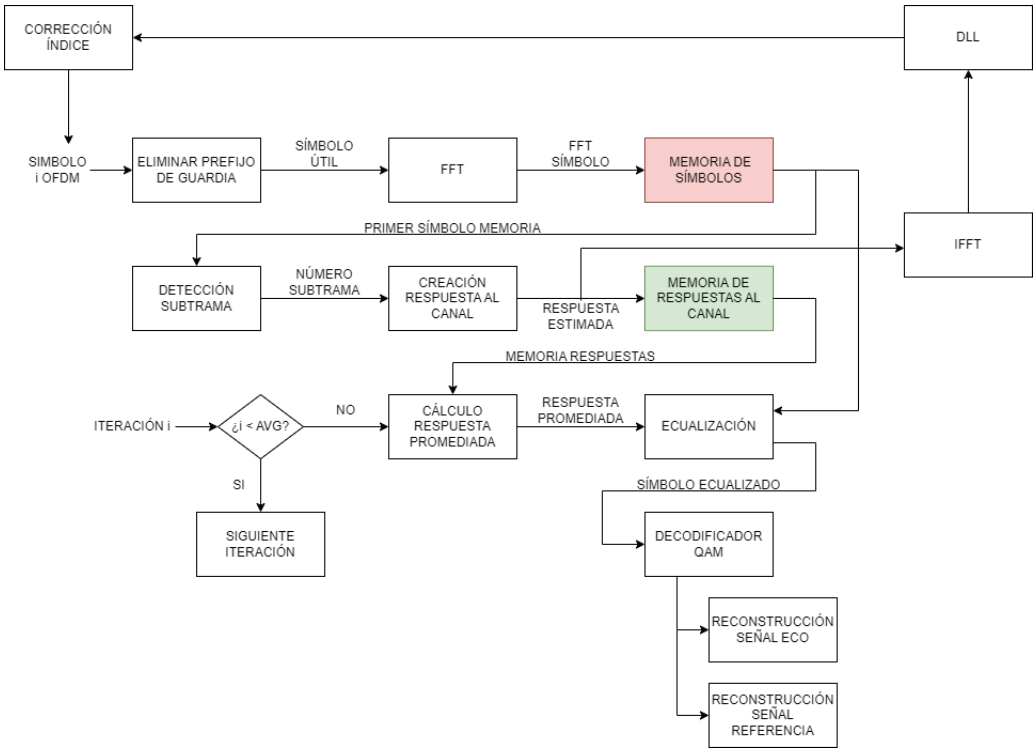


Figura 3: Algoritmo de estimación de canal

Detección y estimación

Esta fase es la responsable de determinar si un pico de correlación visto en la matriz resultado de la CAF se trata de una detección real. Esta decisión es tomada comparando el valor de este pico con un umbral previamente calculado a partir de celdas adyacentes del propio mapa de correlación. Si este umbral es sobrepasado, el pico de correlación bajo evaluación es considerado un objetivo válido. La lógica de este algoritmo se puede ver en el diagrama de bloques mostrado en la figura *4*.

La separación de este objetivo del resto del mapa generará una "nube de puntosçorrespondiente a un único objetivo (visible en la figura *5*). En este capítulo también se tratan dos métodos para refinar la detección a un único par de valores, siendo estos una aproximación polinómica y un centro de masas ponderado.

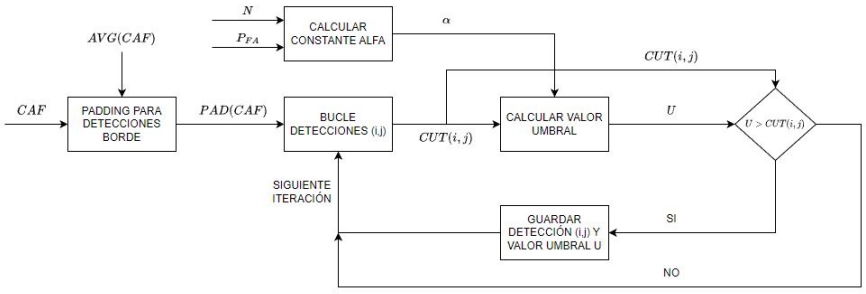


Figura 4: CA CFAR

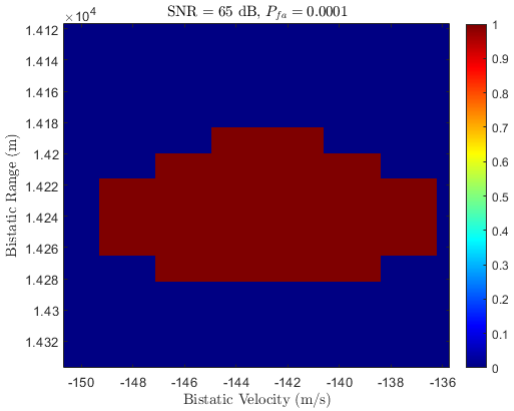


Figura 5: CA CFAR Grupo de positivos (una única detección)

Tracking y predicción

Tras realizarse la detección y estimación, el siguiente paso lógico es determinar como tratar múltiples salidas de este CFAR, es decir, ¿cómo se podría determinar si dos detecciones distintas corresponden o no al mismo objetivo o realidad física? Es necesario establecer una relación entre diferentes detecciones que en realidad muestren el movimiento de un único objetivo, esto además permitirá predecir la próxima posición donde este objetivo podría aparecer en el próximo ciclo.

Este seguimiento permite la creación de una trayectoria plausible en dimensiones de velocidad y distancia biestáticas comúnmente denominada en la literatura RADAR como *traza*. La creación de estas trazas será discutida en la primera parte del capítulo. Tras obtenerse esta traza, a partir de los datos de movimientos obtenidos se realizará la predicción de la próxima posición del objetivo, lo cual se expondrá en mayor detalle en la segunda parte de este capítulo.

Conclusiones

Como se comentó en la introducción, este trabajo es continuación de otro. Este trabajo ofrecía una propuesta para el algoritmo de Batch y la estimación de canal. A partir de este, se procedió a realiza mejoras mínimas en el algoritmo de Batch y se rediseñó por completo la estimación de canal. Una vez realizado este trabajo se diseñaron el estimador de símbolo grueso, el algoritmo CA-CFAR, la creación de trazas y agrupación de objetivos y finalmente la predicción de nuevas posiciones mediante filtrado de Kalman.

Referencias

Las referencias principales tomadas para este trabajo son diferentes trabajos de investigación en el ámbito de los radares pasivos, entre ellos destaca el doctor *Mateusz Malanowski*, cuya publicación *Signal Processing for Passive Bistatic RADAR* ha sido de gran ayuda en la realización de este trabajo. Otra referencia utilizada merecedora de mención es la documentación ofrecida por *Matlab* acerca de diferentes funciones y algoritmos.

Project executive summary

Design, testing and simulation of a passive RADAR in the DVB-T2 band

AUTHOR López Gómez, Alejandro Manuel.
DIRECTOR García de la Cueva, Carlos
DIRECTOR Matanza Domingo, Javier.
COLLABORATING ENTITY ICAI - Pontifical University Comillas.

Summary

Introduction

Traditionally, active radars emit radio signals and detect objects based on the echo of signals reflected by objects in their path. However, passive radars, which take advantage of existing signals in the environment without emitting signals of their own, have gained interest in recent decades due to their potential advantages in terms of energy economy, security and covert detection capability.

DVB-T (Digital Video Broadcasting - Terrestrial) is one of the most common digital modulation standards for digital terrestrial television signals. Due to its widespread use, in addition to other factors, DVB-T signals are commonly used as opportunity illuminators in passive RADAR systems.

As a result of the popularity of the standard, in 2006 the DVB group (responsible for the creation of DVB-T) decided to study options for the design of an updated DVB-T standard. These studies culminated in the creation of DVB-T2 (Digital Video Broadcasting - Second Generation Terrestrial). The main advantage of this new

standard over its predecessor is a more efficient use of bandwidth, the following section will go into more detail on its differences.

Due to the novelty of this standard and its added complexity, there is no work as extensive as for DVB-T as far as RADAR processing is concerned. The main objective of this work is, from a previously designed RADAR processing chain for DVB-T, to make the necessary modifications to work with DVB-T2. Once the DVB-T2 algorithms have been obtained from the DVB-T algorithms, the detection, tracking and target prediction phases will be designed. Finally, once the design is finished, simulations of the final project will be carried out.

As a particularity of the design, it has been chosen to use a single reception channel to minimize system costs, which implies that it will be necessary to separate the reference, echo and noise signals, increasing the complexity of the processing. This project is based on previous work which will be modified accordingly.

Working methodology

This work focused firstly on the study of the analysis of the previous work. The objective of this study phase was to learn the RADAR processing phases, dissect the proposed code and propose possible improvements to it. Once this phase of the project was completed, we proceeded to the implementation of the improvements made or to the redesign of those parts that required it, in addition to the addition of those processing elements foreseen. Finally, simulations were carried out on the different parts of the processing chain and the chosen values were justified by providing the pertinent evidence.

Regarding the follow-up of the work, weekly follow-up sessions were organized with the objective of presenting the progress made by the student and receiving feedback on it. The following table shows the previous planning of the project.

	2022				2023				
OBJETIVO	SEPTIEMBRE	OCTUBRE	NOVIEMBRE	DICIEMBRE	ENERO	FEBRERO	MARZO	ABRIL	MAYO
Estudio y análisis RADAR pasivo DVB-T.									
Modificaciones para trabajar con DVB-T2.									
Diseño de las etapas de detección y tracking.									
Simulaciones cadena de procesado.									
Estudio de sensibilidad, ajustes y pruebas de campo.									

Figura 6: Organization scheme

Unfortunately, the hardware part of the project had to be excluded, so no field tests or hardware sensitivity studies were performed. Instead we have chosen to add to the processing chain a new phase of special interest in RADAR technology, namely the position prediction phase based on Kalman filtering.

Results

This section will briefly discuss the results obtained in the project. The improvements or important changes made in the different phases of the processing will be discussed.

Cross ambiguity function

The cross-ambiguity function (CAF) is a fundamental part of RADAR processing. From the reference and echo signals, a correlation operation is performed. Different values of sample delay and frequency offset are progressively applied to the reference signal. The objective is to find the pair of delay-offset values with a high correlation value. When this situation occurs, it is considered that a detection has been obtained [1].

The block diagram of the Batch algorithm, a practical implementation of this function, is shown below. It consists of separating the signal to be processed into blocks of equal length in order to increase the processing speed. In this case it has been necessary to add a first interpolation phase due to the sampling frequency of the incoming signal and a filtering phase for better noise treatment.

The result of this algorithm can be seen in the same figure, it is a map relating distance and correlation bistatic velocity whose peaks (symbolized in red) denote possible detections.

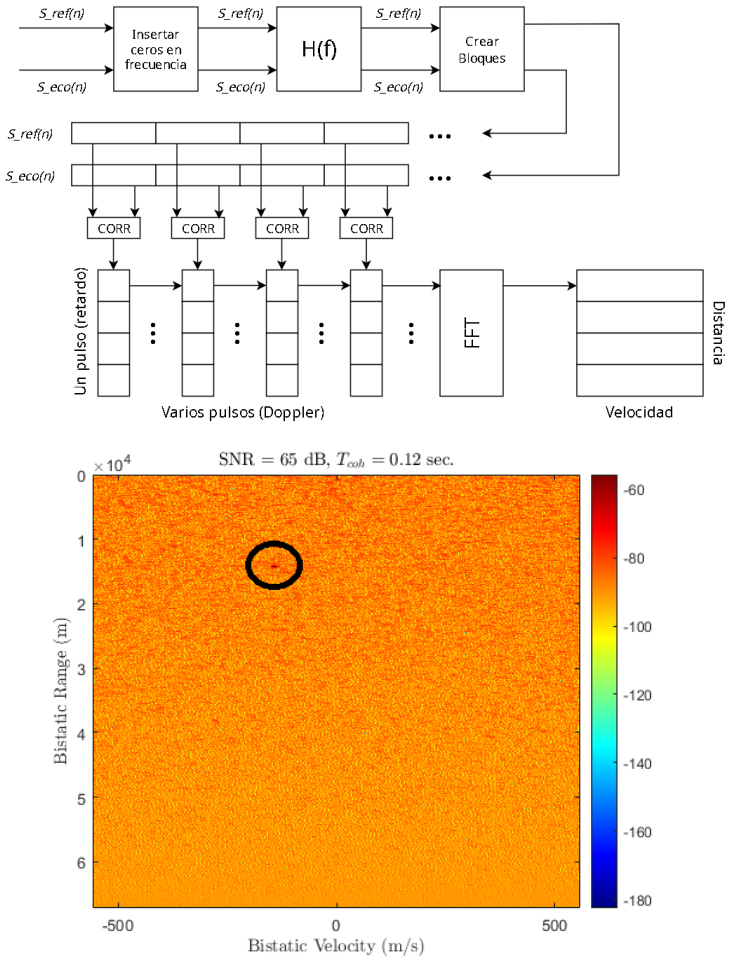


Figura 7: Batch algorithm

Coarse symbol estimation

The coarse symbol estimator is a fundamental element prior to the channel estimator whose function is to solve any initial time synchronization problems present in the signal. The objective of this processing phase is the time synchronization of the input signal, eliminating any possible sample delays it may have. The output of this processing phase is a sample index which informs the channel estimator where the DVB-T signal starts. It is only executed once for each DVB-T signal.

Channel estimation

In the design of the proposed receiver, only a single reception channel is considered, so that the echo and reference signals are mixed in a single received signal. This processing phase has the important function of correctly differentiating and separating the aforementioned signals.

The designed channel estimation algorithm is shown in the following block diagram. In red and green are shown the memories of the symbols and channel responses respectively. The two memories used in this algorithm allow correlation and averaging with previous channel responses, increasing the accuracy of the estimation.

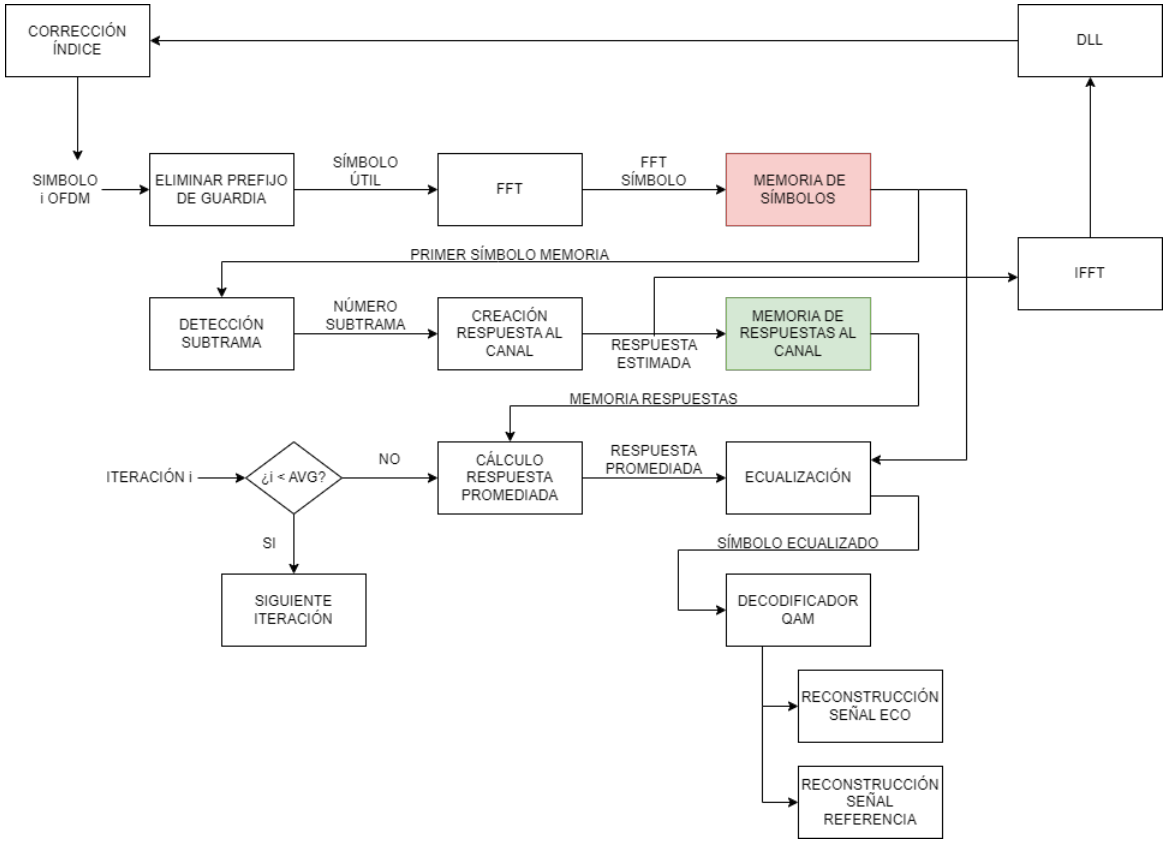
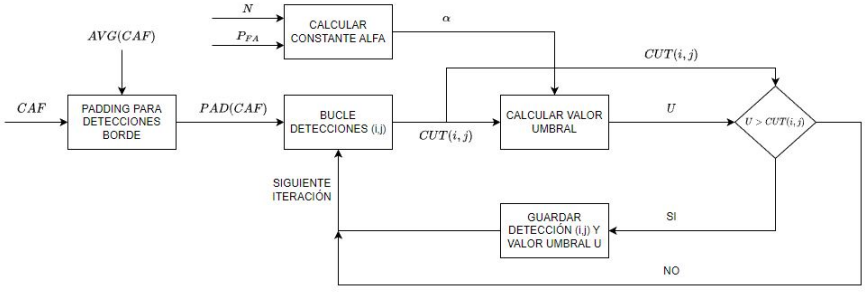


Figura 8: Channel estimation algorithm

Detection and estimation

This phase is responsible for determining whether a correlation peak seen in the CAF result matrix is a real detection. This decision is made by comparing the value of this peak with a threshold previously calculated from adjacent cells of the correlation map itself. If this threshold is exceeded, the correlation peak under evaluation is considered a valid target. The logic of this algorithm can be seen in the block diagram shown in the figure *4*.

Separating this target from the rest of the map will generate a "point cloudçorresponding to a single target (visible in figure *5*). This chapter also discusses two methods for refining the detection to a single pair of values, these being a polynomial approximation and a weighted center of mass.



.

Figura 9: CA CFAR

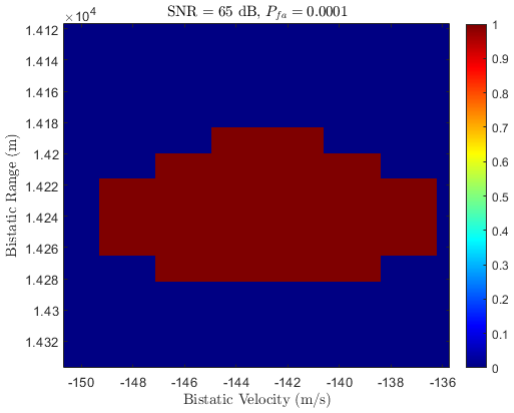


Figura 10: CA CFAR Group of positives (single detection)

Tracking and prediction

After the detection and estimation is done, the next logical step is to determine how to deal with multiple outputs of this CFAR, i.e., how could one determine whether or not two different detections correspond to the same target or physical reality? It is necessary to establish a relationship between different detections that actually show the movement of a single target, this will also allow to predict the next position where this target might appear in the next cycle.

This tracking allows the creation of a plausible trajectory in bistatic velocity and distance dimensions commonly referred to in RADAR literature as *trace*. The creation of these traces will be discussed in the first part of the chapter. After this trace is obtained, the prediction of the next target position will be made from the obtained motion data, which will be discussed in more detail in the second part of this chapter.

Conclusions

As discussed in the introduction, this paper is a continuation of another paper. This work offered a proposal for the Batch algorithm and channel estimation. From this, minimal improvements were made to the Batch algorithm and the channel estimation was completely redesigned. Once this work was done, the coarse symbol estimator, the CA-CFAR algorithm, the creation of traces and grouping of targets and finally the prediction of new positions by Kalman filtering were designed.

References

The main references taken for this work are different research works in the field of passive radars, among them is Dr. *Mateusz Malanowski*, whose publication *Signal Processing for Passive Bistatic RADAR* has been of great help in the realization of this work. Another reference worth mentioning is the documentation provided by *Matlab* about different functions and algorithms.

Capítulo 1

Introducción e ideas fundamentales del diseño

Los radares son sistemas ampliamente utilizados en una variedad de aplicaciones, desde la aviación y la navegación marítima hasta la meteorología y la seguridad. Desde el punto de vista académico, su bajo coste y dificultad añadida en procesado hacen de los radares pasivos un campo muy interesante en investigación. Cabe destacar que este trabajo se trata de una continuación que emplea como base el trabajo realizado el año anterior.

Previa lectura de los diferentes algoritmos e implementaciones realizas, a continuación se explicará muy brevemente las ideas más fundamentales del diseño. Entre ellas los elementos del estándar de señal empleado que afectan al diseño y una explicación a alto nivel de los principales bloques del procesado.

1.1. El estándar DVB-T y su implementación

El esquema de modulación de DVB-T (Digital Video Broadcasting - Terrestrial) se encuentra basado principalmente en transmisión OFDM (Orthogonal Frequency Division Multiplexing). En la literatura del estándar se define el concepto *frame OFDM*. Estos frames son la estructura básica en la que se organiza la transmisión de la señal DVB-T. Cada frame posee una duración establecida y consiste en 68 símbolos OFDM. Cuatro frames OFDM forman lo definido por el estándar como un *super frame OFDM*. Cada símbolo está formado por una parte útil de duración definida y un intervalo de guardia o prefijo circular previo al símbolo útil [2].

1.1.1. Estructura OFDM

Todo frame OFDM contiene portadoras de datos moduladas en QPSK, 16-QAM o 64-QAM. Las proporciones de las constelaciones creadas para cada esquema de modulación dependen de otro parámetro definido en el estándar denominado como α, el cual puede tomar los valores uno, dos y cuatro. Este parámetro es definido como la mínima distancia entre dos puntos de una misma constelación cuyos valores sean distintos dividida entre la mínima distancia entre dos puntos cualesquiera de la constelación. Además de datos transmitidos, un frame OFDM contiene los siguientes elementos [2].

- Portadoras pilotos dispersas. En función del frame OFDM los índices de las portadoras pueden variar, sin embargo estos índices varían según $[K_{min}; K_{max}]$, siendo $K_{min} = 0$ y $Kmax$ es definido por el modo de transmisión.

- Portadoras pilotos continuas. La posición de estas portadoras es fija y determinada por el estándar.

- Portadoras TPS (Transmission Parameter Signalling). Estas portadoras son empleadas para señalizar parámetros relacionados con el esquema de transmisión, la modulación... Su posición, al igual que los pilotos continuos es fija y definida por el estándar.

Estos elementos son empleados en sincronización de frames, sincronización en frecuencia, sincronización temporal, estimación de canal en el proceso de demodulación e identificación del modo de transmisión.

Se definen hasta tres modos de transmisión en el estándar DVB-T, siendo estos denominados como *2K, 4K y 8K*. Cada modo de transmisión define una serie de parámetros que modifican cómo la señal es modulada. En concreto se definen el número de portadoras continuas y dispersas, el valor K_{max} y el número de portadoras de datos disponibles.

1.1.2. Generación de pilotos

Los pilotos continuos y dispersos son generados a partir de una secuencia pseudoaleatoria PRBS (Pseudorandom Binary Sequence). En función de esta secuencia se establece la localización de las pilotos continuas y dispersas [2].

Respecto a la posición de las pilotos dispersas, para el símbolo OFDM de índice i se define mediante la siguiente expresión

$$k = K_{min} + 3 * (i \text{ mód } 4) + 12p <=> p \in N, p \geq 0, k \in [K_{min}; K_{max}] \tag{1.1}$$

Donde p es un valor entero que toma todos los posibles valores mayores o iguales que cero tal que el índice k (posición final del piloto disperso) se encuentre dentro del rango $[K_{min}; K_{max}]$.

Los posibles valores a obtener de esta ecuación para el valor k son $12p, 3+12p, 6+12p, 9+12p$. Cada uno de estos resultados forma un patrón de inserción que caracteriza a un tipo de símbolo OFDM, por tanto se pueden dar hasta cuatro tipos de símbolo en lo que a localización de pilotos dispersos se refiere. Se debe asegurar que k entra en el rango establecido por el estándar. El valor máximo de k es dado por el último de estos valores, $9+12p$. Por tanto se debe cumplir la siguiente expresión [3].

$$K_{max} \geq 9 + 12p \tag{1.2}$$

Despejando p de la anterior ecuación se extrae la siguiente conclusión. El valor de p es ahora conocido, en concreto el valor hallado es el valor óptimo para la modulación, pues abarca todo el rango permitido para índices de portadoras piloto dispersas.

$$p_{max} = \frac{K_{max} - 9}{12} \tag{1.3}$$

El valor de amplitud asignado a las portadoras tanto pilotos como dispersas se obtiene a partir de la secuencia PRBS y la siguiente expresión definida por el estándar [2].

$$Real(c_{m,i,k}) = \frac{4}{3} * 2 * (\frac{1}{2} - w_k) \tag{1.4}$$

$$Imag(c_{m,i,k}) = 0 \tag{1.5}$$

La amplitud de las portadoras TPS se obtiene mediante esta expresión [2].

$$Real(c_{m,i,k}) = 2 * (\frac{1}{2} - w_k) \tag{1.6}$$

$$Imag(c_{m,i,k}) = 0 \tag{1.7}$$

Donde m es el índice del frame OFDM, k es el índice de la portadora e i es el índice dentro de cada símbolo OFDM. Los pilotos solamente presentan una parte real cuyo valor es mayor que una portadora de datos común. No contienen parte imaginaria o compleja.

1.1.3. Selección de parámetros en el proyecto

Para la realización de diferentes pruebas y simulaciones, se han escogido los valores de transmisión y modulación más típicos en territorio español. La definición de estos parámetros puede encontrarse en el Anexo G.

1.2. Esquema general del diseño

En la siguiente figura se muestra el esquema de una cadena de procesado RADAR para modulaciones digitales.

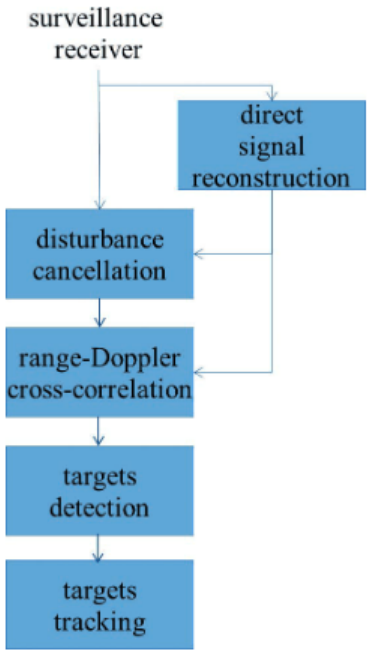


Figura 1.1: Esquema procesado RADAR pasivo [4]

Este esquema es básico para todo sistema de radar pasivo. El bloque dedicado a la reconstrucción de la señal debe realizarse teniendo en cuenta la modulación a emplear. Para el caso de DVB-T se debe tener en cuenta lo siguiente.

- **Eliminación y adición del prefijo circular propio de DVB-T.** Esta banda de guarda además de proteger ante interferencia entre símbolos permite identificar el inicio y fin de los mismos. Es eliminada en un primer paso ya

que no contiene información útil de la señal. Al finalizar la reconstrucción se volverá a añadir.

- **Estimación del canal y ecualización de símbolos.** La ecualización es una parte importante de la demodulación del estándar DVB-T. Mejora la precisión de la identificación de símbolos QAM (ó QPSK) en una decodificación posterior.

- **Demodulación de la señal.** En el estándar de DVB-T se recoge el uso de QPSK, 16-QAM y 64-QAM.

El procesado de la señal con finalidad RADAR es realizado por los tres últimos bloques, los cuales son independientes de la modulación o estándar de la señal.

- **Correlación cruzada o función de ambigüedad.** La correlación cruzada se calcula para un rango determinado de rangos y velocidades. Si el retardo y el Doppler aplicados a la señal de referencia coinciden con la señal de eco, en el mapa aparecerá un pico de correlación indicando el rango y la velocidad. Existen varias técnicas para implementar este bloque, todas basadas en la FFT.

- **Detección.** En este bloque son empleados los conocidos como algoritmos CFAR *(Constant False Alarm Rate)*. Esta técnica consiste en la estimación de un límite a partir de las características de la señal. Si el valor del pico de correlación supera este límite se considera una detección.

- **Tracking.** Los algoritmos de tracking en la mayoría de radares pasivos emplean el filtro de Kalman. La idea es la creación de regiones de predicción de movimiento para la siguiente observación. Cada predicción correcta (el objetivo es detectado dentro de la zona de predicción) disminuye la incertidumbre, reduciendo el tamaño de la siguiente región o área de predicción.

En apartados posteriores se desarrollará en profundidad la implementación práctica y diseño de cada uno de los bloques.

Capítulo 2

Función de ambigüedad (AF) y función de ambigüedad cruzada (CAF)

En este primer apartado de diseño, se tratarán dos conceptos fundamentales en el procesado RADAR. Siendo estos la función de ambigüedad (AF) y función de ambigüedad cruzada (CAF).

La función de ambigüedad (AF) es parte fundamental del entendimiento del procesado RADAR. Aunque esta no es utilizada en la cadena de procesado, es indispensable para verificar el comportamiento y viabilidad del estándar de modulación a emplear ante diferentes valores de retardo y desplazamiento en frecuencia. En la función de ambigüedad cruzada (CAF), en vez de tenerse en cuenta una única señal, se emplean la señal de referencia (señal recibida entre receptor e iluminador de oportunidad) y la señal de eco (señal proveniente del objetivo).

2.1. Función de ambigüedad (AF)

La función de ambigüedad (AF) se puede entender como la operación de correlación de una señal con una copia de esta misma afectada por un retardo temporal y desplazamiento en frecuencia. [5] Desde el punto de vista de implementación, su expresión matemática es la siguiente.

$$\psi(m,k) = \sum_{n=0}^{N-1} s_x(n) \cdot s_x^*(n-m) \cdot e^{-j\frac{2\pi}{N}kn} \tag{2.1}$$

2.1.1. Robustez del estándar DVB-T frente a la AF

En la siguiente figura se muestra el resultado de la función de ambigüedad. Los picos que se observan en los puntos de retardo y desplazamiento en frecuencia nulos corresponden a la primera iteración del algoritmo. Este resultado es esperable puesto que la mayor correlación entre señales se dará cuando estas sean iguales (es decir sin retardo ni desplazamientos aplicados).

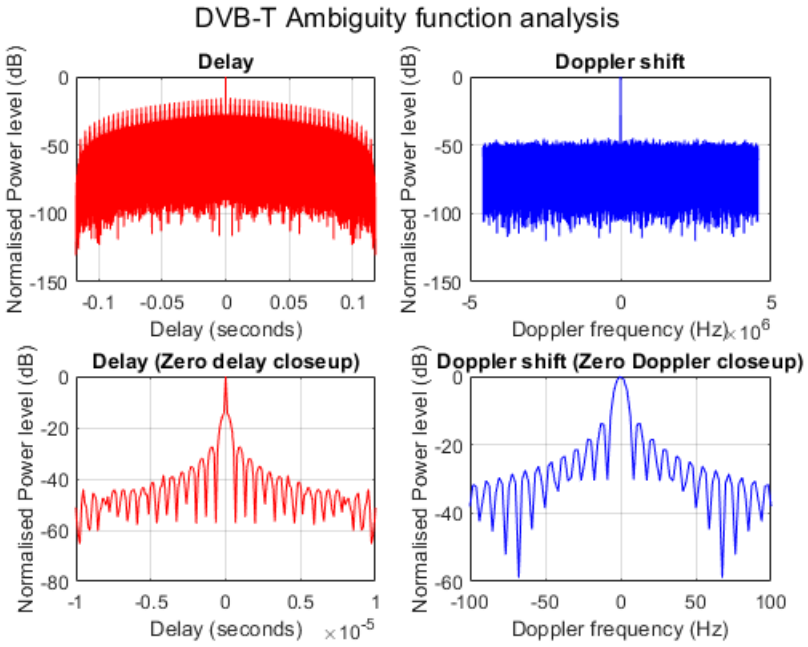


Figura 2.1: Análisis de robustez señal DVB-T

Los picos repetitivos observados en la señal roja correspondiente al dominio temporal son causados por la presencia de elementos repetitivos o periódicos en la señal analizada. Esto es congruente con los diferentes tipos de pilotos presentes en el estándar DVB-T explicado en apartados anteriores. Debido a que la separación de estos es constante ($1/T_u$ siendo T_u la duración de símbolo útil), las protuberancias observadas presentarán un patrón similar. Aunque estos pilotos sean de gran utilidad durante la demodulación de la señal en diversas operaciones mencionadas en apartados anteriores, la presencia de estos picos descritos anteriormente es un problema en lo que a procesado RADAR se refiere, puesto que podrían ser interpretados erróneamente como detecciones validas, dándose lo que se conoce como *falso positivo*.

2.1.2. Enventanado y su efecto en la AF

Aparte de analizar el comportamiento del estándar sin nigún tipo de operación previa aplicada, es conveniente verificar el comportamiento de la señal ante diferentes operaciones tales como enventanado en diferentes dominios e interpolación. En la figura *2.2* se muestra el resultado de realizar una operación de enventanado en el tiempo empleando un filtro de Chebyshev.

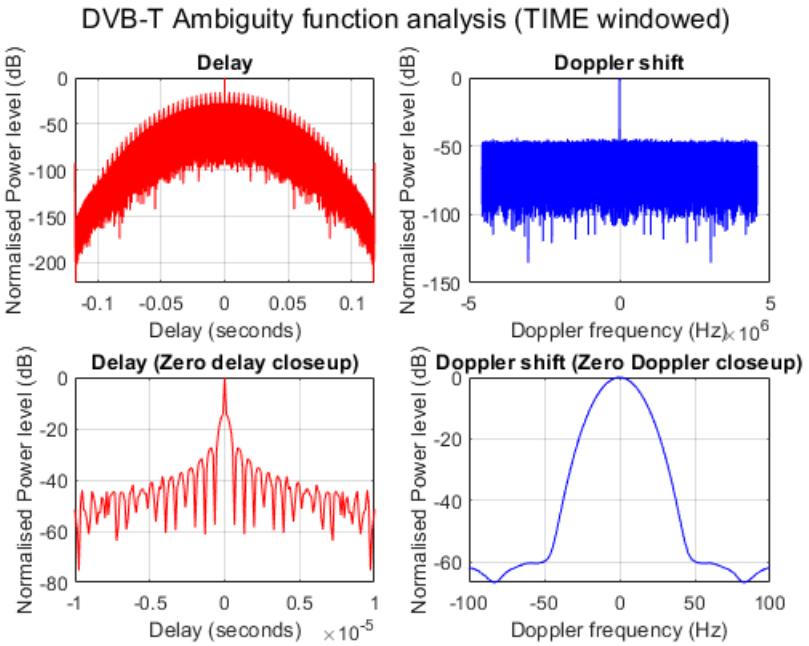


Figura 2.2: Análisis de robustez señal DVB-T (Enventanado en el tiempo)

La aplicación de la operación de enventanado en el dominio de la frecuencia demuestra una importante disminución de lóbulos secundarios en frecuencia, tal y como teóricamente es esperado. Sin embargo, esta operación provoca un ensanchamiento del lóbulo principal (fenómeno conocido como fuga espectral el cual será explicado posteriormente) que podría oscurecer posibles detecciones cercanas. Por tanto es necesario considerar un cierto margen en frecuencia (en el caso de la figura de aproximadamente 50 Hz) donde la sensibilidad de radar o sus capacidades de detección habituales se ven afectadas por una detección cercana.

En la siguiente figura (figura *2.3*) se muestra el resultado de realizar esta misma operación de enventanado pero en el dominio de la frecuencia (conformado de

la densidad espectral de potencia de la señal), se ha empleado el mismo filtro de Chebyshev.

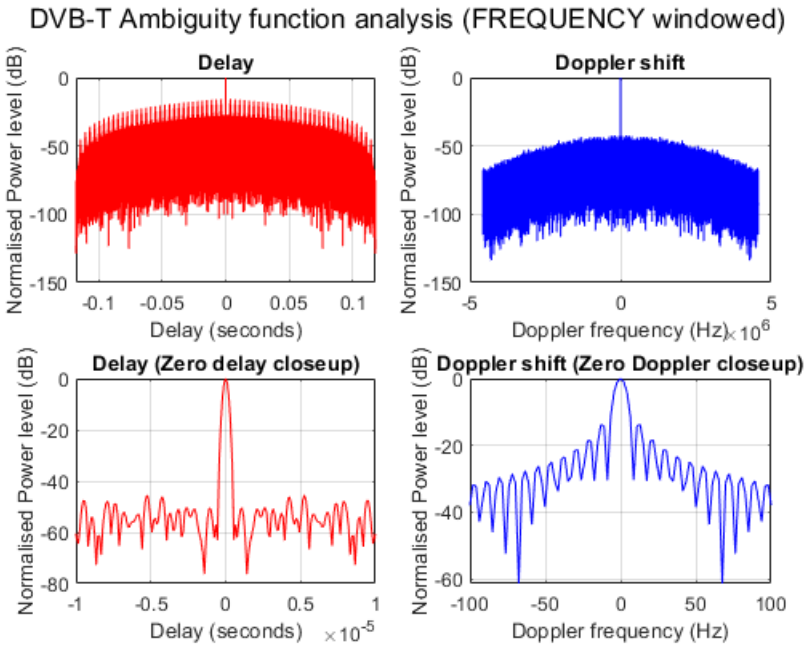


Figura 2.3: Análisis de robustez señal DVB-T (Enventanado en frecuencia)

El enventanado en frecuencia logra reducir los lóbulos secundarios en el dominio del tiempo de forma drástica sin provocar un ensanchamiento notable del haz principal. La selección de la ventana es un criterio de diseño que debe ser estudiado en función de los resultados que se busquen. En ejemplos anteriores se ha empleado la ventana de Chebyschev a modo de ejemplo, sin embargo, existe una multitud de ventanas a elegir. Los principales parámetros a tener en cuenta a la hora de escoger una ventana son los siguientes [6].

- Anchura del lóbulo principal. Este se define como el índice en frecuencia donde la potencia decae en $3dB$ respecto a la potencia de lóbulo principal.
- Atenuación de lóbulos secundarios. Este criterio se encuentra relacionado con un fenómeno denominado *spectral leakage*.

Spectral leakage, en español fuga espectral, se refiere a la dispersión de la energía de una señal fuera de su banda de frecuencia original. Este fenómeno ocurre cuando

una señal es muestreada en el dominio del tiempo y se le aplica una transformada de Fourier. Si la frecuencia de la señal no coincide exactamente con uno de los puntos de muestreo, se producirá un efecto de solapamiento o "fuga.[en] las frecuencias adyacentes, lo que resulta en una pérdida de precisión en la medición de la energía de la señal en su banda de frecuencia original.

Para minimizar la fuga espectral, se pueden utilizar técnicas de enventanado que reducen la energía de la señal fuera de su banda de frecuencia original, o se pueden aplicar métodos de interpolación para aumentar la precisión de la medición en frecuencias adyacentes.

En lo que a selección de ventana respecta, la selección perfecta sería aquella que proporcionase la menor fuga espectral posible, es decir un lóbulo principal lo más estrecho posible que no afecte a detecciones cercanas debido al fenómeno de oscurecimiento mencionado anteriormente, y la mayor reducción en lóbulos secundarios posibles que permita eliminar la posibilidad de falsos positivos (es decir confundir un lóbulo secundario de una detección válida como otra detección independiente).

Sin embargo, si existiese esta opción la elección de función de enventanado sería trivial. Una ventana no puede poseer todas las cualidades que hacen de esta una ventana ideal. Tómese por ejemplo la función de enventanado rectangular (señal azul en la figura *2.4*. Esta ventana presenta la mejor anchura de lóbulo principal de las analizadas, lo que ofrece mínima fuga espectral. La desventaja viene a la hora de analizar el SLL (Side Lobe Level) el cual es claramente superior que cualquier otra ventana. Lo contrario de esta ventana en el conjunto de ventanas analizadas en la figura sería la ventana de Chebyschev (señal verde en la figura). Esta función de enventanado logra el menor nivel de lóbulos secundarios de todas, sin embargo también ofrece la mayor anchura del lóbulo principal.

A cambio de un mayor nivel de lóbulos secundarios y por tanto una mayor cantidad de falsos positivos se puede obtener una anchura de lóbulo principal reducida, lográndose menor fuga espectral y mayor precisión en detecciones vecinas. Si el objetivo principal es reducir lo mas posible los lóbulos secundarios, se deberá sacrificar anchura de lóbulo principal y se perderá sensibilidad para otras detecciones cercanas. En resumen, la elección de la ventana debe ser realizada por el diseñador en función de los criterios que se consideren más importantes. Por ejemplo, si el objetivo es separar o identificar dos señales que se encuentran cercanas en frecuencia y son similares en potencia, se debería escoger la ventana rectangular [6].

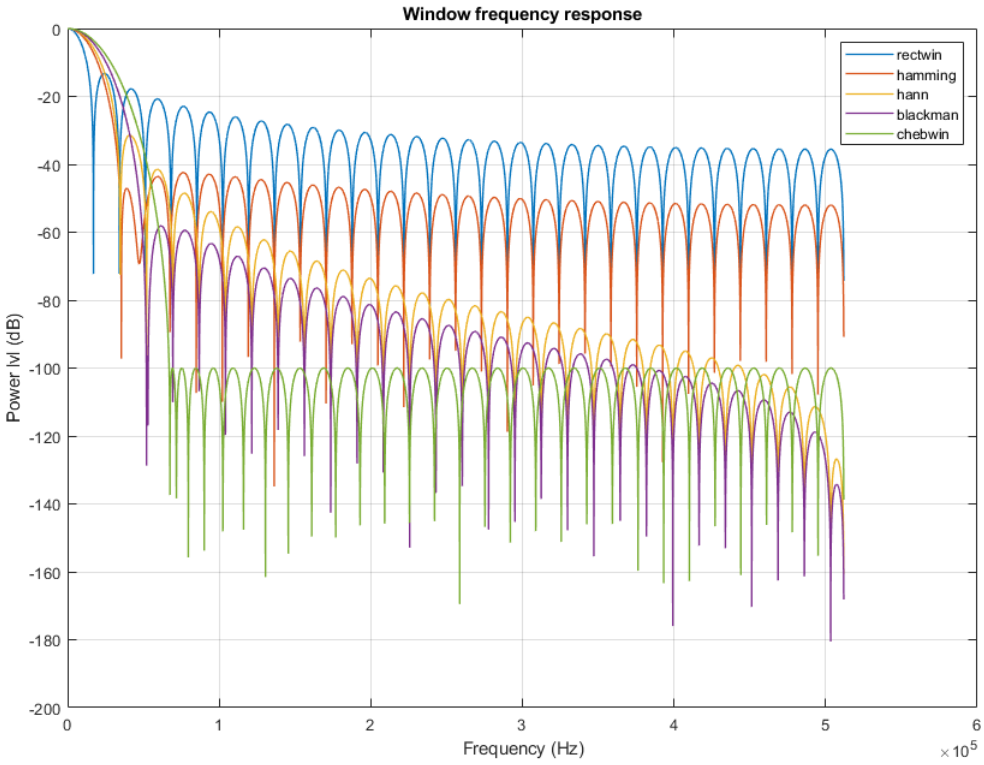


Figura 2.4: Respuesta en frecuencia de diferentes ventanas

Por otro lado, si el objetivo es separar o identificar dos señales de intensidades diferentes y frecuencias diferentes, la energía de una señal puede filtrarse a través de los lóbulos secundarios. En este caso, no importaría tener uno de los lóbulos principales más anchos y se sacrificaría resolución del lóbulo principal. Respecto al proyecto, se ha optado por emplear la ventana de Chebyschev, debido a que se considera más importante mitigar los lóbulos secundarios y se prioriza ofrecer detecciones fiables y poco afectadas por posibles fenómenos de fuga espectral.

2.2. Función de ambiguedad cruzada (CAF)

La función de ambigüedad cruzada (CAF) es una parte fundamental del procesado RADAR. A partir de las señales de referencia y eco, se realiza una operación de correlación. Se aplican progresivamente diferentes valores de retardo muestral y desplazamiento en frecuencia a la señal de referencia. El objetivo es lograr encontrar el par de valores retardo desplazamiento con un alto valor de correlación. Cuando se de esta situación se considera que se ha obtenido una detección. [1].

$$\psi(m,k) = \sum_{n=0}^{N-1} s_{eco}(n) \cdot s^*_{ref}(n-m) \cdot e^{-j\frac{2\pi}{N}kn} \tag{2.2}$$

Estos resultados deben ser almacenados de tal forma que permitan su visualización como un *mapa* que relacione velocidad y distancia (obtenidos de los valores de retardo y desplazamiento aplicados). La expresión matemática de esta operación es la siguiente.

2.2.1. Problemas de puesta en práctica

Idealmente, la implementación práctica de esta operación sería la ejecución directa de esta expresión, sin embargo, la realización de esta operación para señales de longitudes reales es de una complejidad computacional muy costosa, en concreto de $O(N^2)$.

Es necesario realizar, para cada valor de m y k una multiplicación compleja y una suma. Esto hace necesario realizar algún tipo de implementación diferente a la original con el objetivo de reducir a valores más realistas para el procesador RADAR el tiempo de ejecución del algoritmo. [7].

2.3. Algoritmo de Batch (procesado en bloques)

La implementación práctica escogida es el algoritmo de Batch. Este algoritmo es ampliamente conocido en aplicaciones de procesado RADAR. [8] Consiste en la separación en bloques de la señal y la aplicación de correlaciones a cada par de bloques de las señales eco y referencia de forma independiente y paralela. En la siguiente figura se puede observar el diagrama de bloques de este algoritmo.

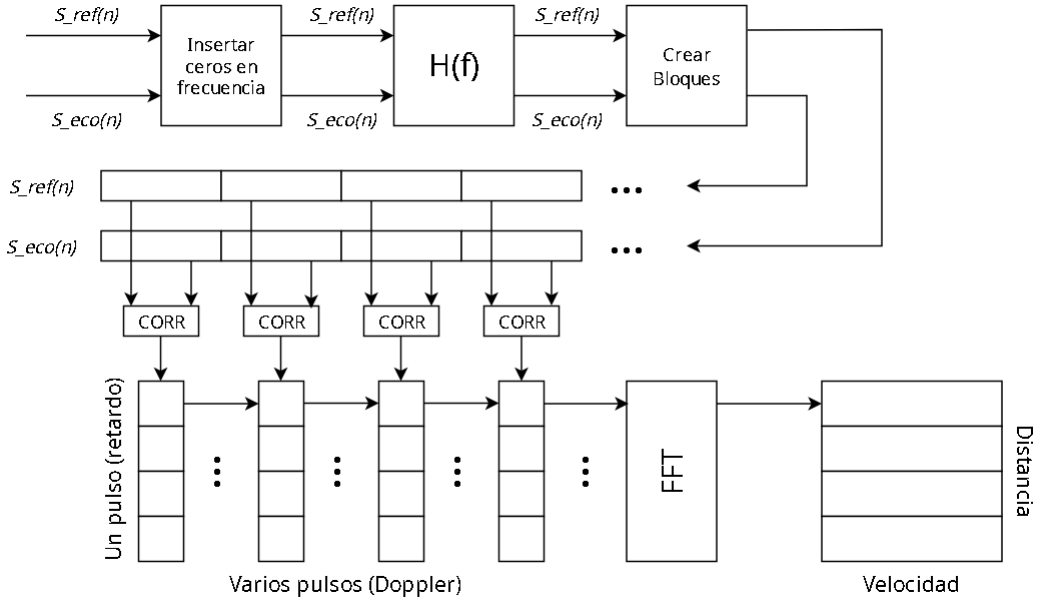


Figura 2.5: Diagrama de bloques. Algoritmo de Batch

En el proyecto realizado, la implementación de este algoritmo se diferencia de la habitual en los dos bloques iniciales (interpolación y filtrado) cuya finalidad se explica en el siguiente apartado.

El algoritmo original comienza con la separación de las señales eco y referencia en bloques de igual longitud. Se aplica una operación de correlación a cada par de bloques. Desde la terminología RADAR tradicional, se puede entender el producto de cada correlación como pulsos consecutivos, y la concatenación de diferentes resultados de esta correlación como un único pulso. Finalmente se aplica una FFT a cada pulso, obteniéndose este mapa de distancia y velocidad mencionado en la introducción de este apartado.

2.3.1. Demostración de la necesidad de la interpolación

El primer bloque del diagrama se puede observar que la operación realizada es la *inserción de ceros en frecuencia*. En la literatura de teoría de señal, a esta operación comúnmente se le denomina *interpolación*. En objetivo principal es lograr aumentar la frecuencia a la que es muestreada una cierta señal, obteniéndose una mayor resolución en la señal muestreada.

Para la demostración de esta operación, considérese el espectro en frecuencia de un ruido blanco *gaussiano* de media nula y cuya densidad espectral de potencia es σ_x^2. como el que se muestra en la siguiente figura (el valor de este parámetro es irrelevante para la demostración).

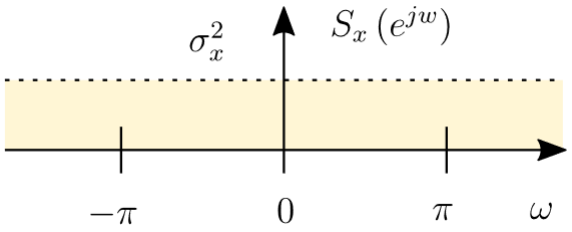


Figura 2.6: Espectro señal de ruido

El detalle fundamental a ojos de la demostración es la no limitación en banda de la señal, ya que un ruido blanco es aquel ruido presente en todas las frecuencias. Calculando su ITDFT (Inverse Time Discrete Fourier Transform) se obtiene el siguiente resultado.

$$Rx[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} Sx(e^{jw})e^{jw}dw = \frac{1}{2\pi}\int_{-\pi}^{\pi} \sigma_x^2 e^{jw}dw = ... = \sigma_x^2 sinc(n) \tag{2.3}$$

La función de auto correlación $Rx[n]$ de una señal de ruido blanco se trata de una función $sinc(n)$. Al realizar el muestreo digital de esta señal (se considera importante recordar que en muestreo de señales digitales los valores de n deben ser enteros) esta será muestreada en sus puntos nulos, a excepción de $n = 0$, ya que tras resolverse la indeterminación del cálculo se obtiene como valor la unidad. Este fenómeno puede ser observado en la figura *2.7*.

Este fenómeno es un problema a tener en cuenta durante el procesado RADAR. El muestreo poco preciso de una señal entrante provocará que aquellos lóbulos secundarios que deben ser atenuados mediante el bloque de filtrado posterior no sean atenuados, de forma que al muestrear de nuevo la señal con una frecuencia de muestreo distinta en cualquier otra etapa de procesado revelará estos lóbulos que han sido ocultados por un muestreo incorrecto.

La causa de este error en el muestreo es debido a utilizar un espaciado o paso de muestreo entero, la solución consiste en variar este espaciado a un numero

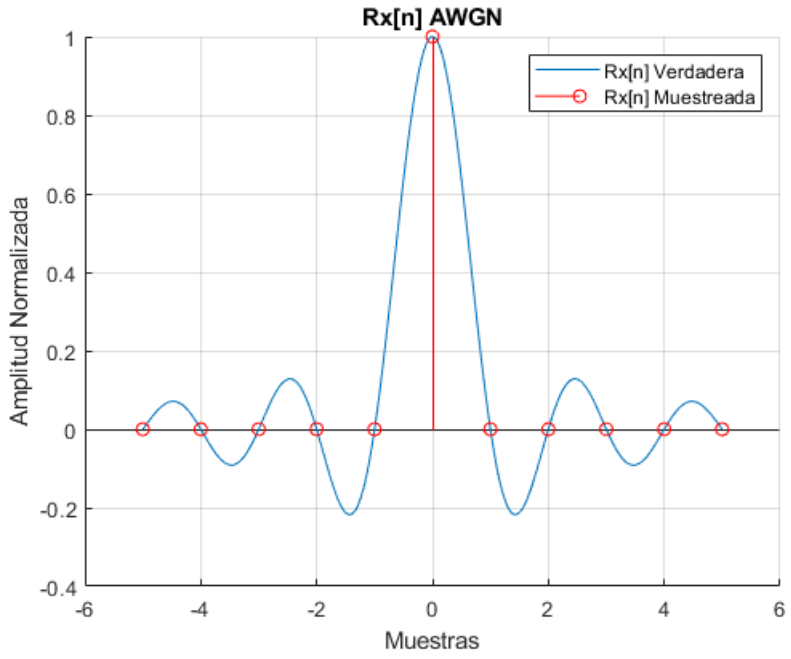


Figura 2.7: Rx[n] muestreada y real

racional, lo que implica cambiar la frecuencia de muestreo. Aplicando una interpolación de factor L, el espectro de la señal de ruido se verá limitado en banda.

Realizando de nuevo el cálculo de la función de auto correlación se obtiene lo siguiente.

$$Rx[n] = \frac{1}{2\pi}\int_{-\pi/L}^{\pi/L} Sx(e^{jw})e^{jw}dw = \frac{1}{2\pi}\int_{-\pi/L}^{\pi/L} \sigma_x^2 e^{jw}dw = ... = \frac{\sigma_x^2}{L}sinc(\frac{n}{L}) \qquad (2.4)$$

El resultado de aplicar dicha interpolación se muestra en la figura *2.8*. La amplitud de la señal se verá afectada por un factor de 1/L, sin embargo un enventanado posterior de amplitud L es empleado (entre otros factores) con el objetivo de contrarrestar esta reducción en potencia.

La conclusión fundamental a extraer de lo expuesto anteriormente es la importancia de la interpolación para resaltar o "sacar a la luz"los lóbulos secundarios de la señal en el dominio muestral, ya que sin esta operación hubieran sido pasados por alto. Las conclusiones son análogas para el dominio de la frecuencia debido a la simetría de la transformada de Fourier.

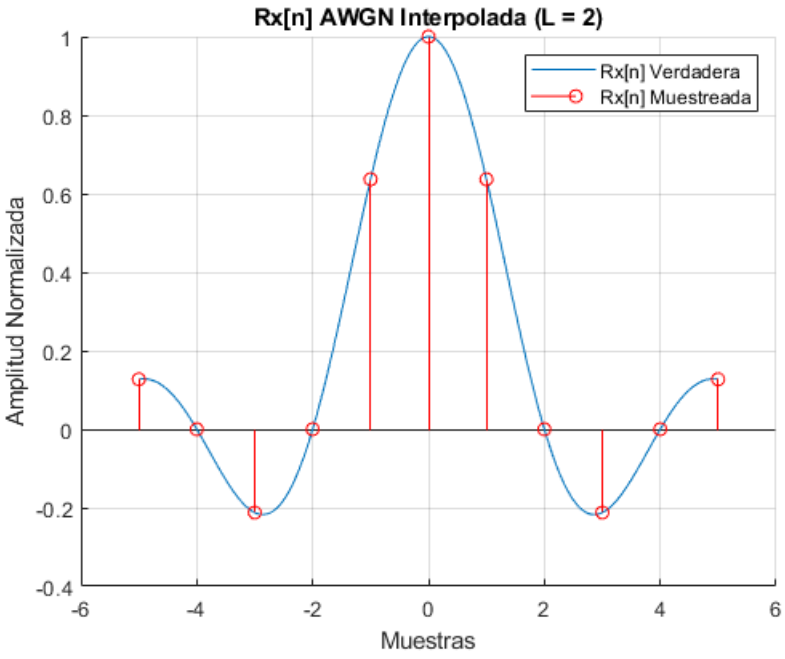


Figura 2.8: Rx[n] muestreada y real tras interpolación

2.3.2. Limitaciones teóricas del algoritmo

El algoritmo de Batch presenta una serie de limitaciones que deben ser tenidas en cuenta. Estas vienen dadas por factores como la frecuencia de muestreo o la frecuencia central de transmisión de la señal, pero sobre todo por un parámetro anteriormente mencionado, la longitud del bloque de señal. Las limitaciones se manifiestan en forma de umbrales de detección máximos, tanto en velocidad como en distancia. Los valores máximos detectables son los siguientes:

$$Vmax = \frac{fsc}{4Lf} \tag{2.5}$$

$$Rmax = \frac{cL}{2fs} \tag{2.6}$$

Deducción límite en velocidad

Partiendo del teorema de la frecuencia de muestreo de Nyquist [9], se sabe que para evitar problemas de aliasing, la frecuencia de muestreo aplicada debe mantener la siguiente relación respecto a la frecuencia máxima de la señal.

$$Fs > 2Fmax \tag{2.7}$$

Extrapolando este concepto al procesado RADAR, la variación máxima detectable en frecuencia (es decir la variación en frecuencia máxima de la señal que es posible detectar) será determinada por la frecuencia de muestreo escogida. Además, debido al empleo del algoritmo de Batch, el cuál realiza una separación en bloques de longitud L de las señales de eco y referencia entrantes, se debe tener en cuenta que la frecuencia de muestreo de cada bloque pasará a ser fs/L. Por tanto, la variación de frecuencia máxima detectable será la siguiente.

$$Fdmax < \frac{Fs}{2L} \tag{2.8}$$

Siendo Fdmax el desplazamiento en frecuencia máximo detectable. Ahora sería necesario transformar esta variación en frecuencia a un valor de velocidad en unidades más físicamente interpretables. Partiendo de la expresión para el cálculo de la variación Doppler de la literatura RADAR convencional [10]:

$$\Delta F = \frac{2\Delta V}{c} f_o \tag{2.9}$$

Siendo Δf la variación en frecuencia presente en la señal de eco, ΔV la variación en velocidad resultante, c la velocidad de la luz y f_o la frecuencia central de transmisión de la señal original. Conociendo Δf, que se corresponde con lo obtenido en (*2.8*) y despejando la expresión (*2.9*), la velocidad máxima detectable no ambigua detectable por el algoritmo de Batch será:

$$velocidad_{MAX} = \frac{cFs}{4Lf_o} \tag{2.10}$$

Deducción límite en distancia

El retardo máximo medible por el algoritmo de Batch es igual a la propia longitud del bloque. Este desplazamiento está especificado en muestras. En unidades de tiempo este retardo máximo será:

$$retardo_{MAX} = \frac{L}{Fs} \tag{2.11}$$

Partiendo de una expresión sencilla de un MRU (movimiento rectilíneo uniforme) como es $X = VT$, Siendo X la distancia recorrida, V la velocidad y T el tiempo empleado. Modificándola para el caso de procesado RADAR, teniendo en cuenta que solamente se está interesado en la mitad de la distancia recorrida y la velocidad es similar a la de la luz, se obtiene que:

$$X = \frac{cT}{2} \tag{2.12}$$

Para obtener la distancia máxima medible, conociendo el tiempo máximo que puede emplear la señal en realizar el trayecto *iluminador objetivo receptor* y sustituyendo en la expresión anterior se obtiene lo siguiente.

$$distancia_{MAX} = \frac{cL}{2Fs} \tag{2.13}$$

Conclusiones

Siendo L el tamaño de un bloque Batch. Este parámetro influenciará en el rendimiento de detección del algoritmo de la siguiente forma:

- A **mayor** tamaño de bloque Batch, **menor** resolución en velocidad y **mayor** resolución en distancia.
- A **menor** tamaño de bloque Batch, **mayor** resolución en velocidad y **menor** resolución en distancia.

2.3.3. Resultados de simulaciones

El resultado de la función de correlación cruzada ha sido descripto anteriormente como un mapa que relaciona las magnitudes de velocidad y distancia. En los resultados de la simulación observada en la figura *2.9*, la frecuencia central de la señal transmitida presenta un valor de $f = 300MHz$, la frecuencia de muestreo (establecida por el canal seleccionado para la transmisión de la señal DVB-T, véase el capítulo 2) es de $fs = 9{,}14MHz$ y la longitud de un bloque Batch cualquiera es $L = 4096$. Conocidos estos valores y empleando las expresiones de distancia y velocidades máximas de expresadas en las ecuaciones (*2.5*) y (*2.6*), los umbrales de detección máxima o sensibilidad del algoritmo Batch implementado para esta señal son los siguientes.

$$Velocidad_{max} = \frac{fsc}{4Lf} = 558{,}0357\ m/s \tag{2.14}$$

$$Distancia_{max} = \frac{cL}{2fs} = 67{,}1840\ km \tag{2.15}$$

El escenario planteado en la imagen revela una detección a una distancia aproximada de 1,5 km y cercana a los 150 m/s. Cabe destacar que este mapa de distancia velocidad no es el resultado final de la cadena de procesado RADAR. Tal y como ha sido explicado en el capítulo dos, posterior a la función de ambiguedad cruzada deben realizarse las fases de detección y seguimiento de objetivos.

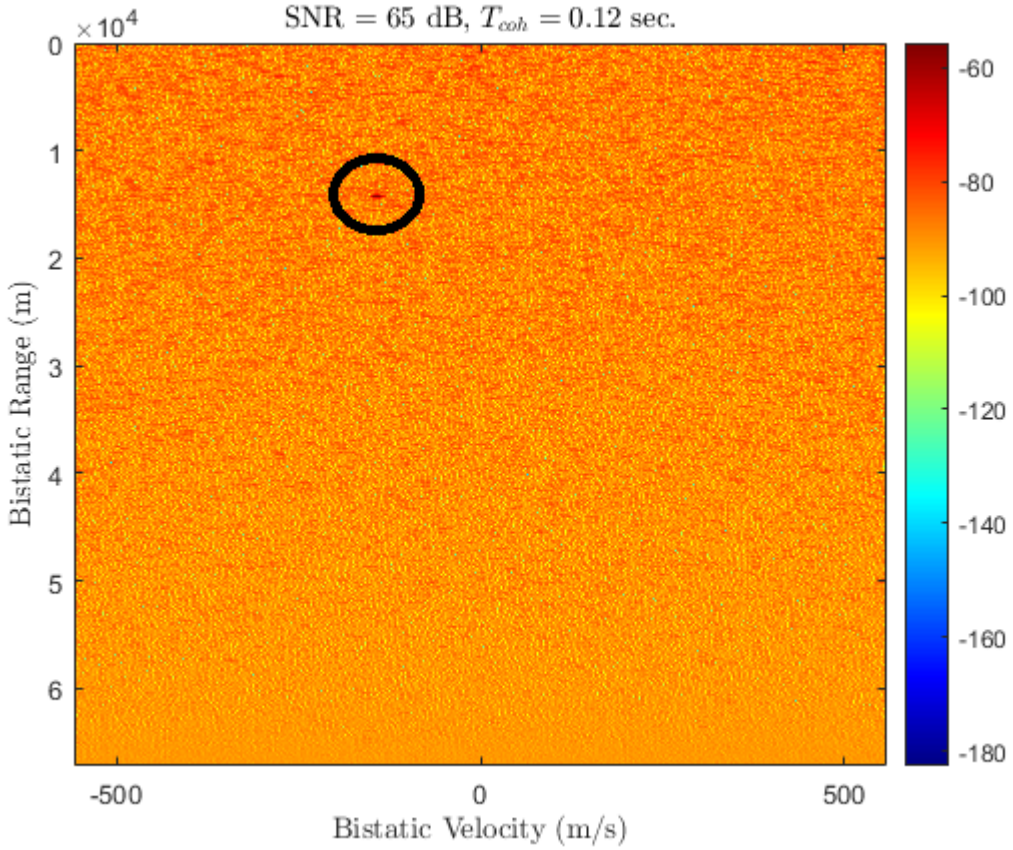


Figura 2.9: Mapa distancia velocidad

Capítulo 3

Estimación gruesa de retardo de símbolo

El estimador de símbolo grueso es un elemento fundamental previo al estimador de canal cuya función es solucionar cualquier problema inicial de sincronización temporal presente en la señal. El objetivo de esta fase del procesado es la sincronización temporal de la señal de entrada, eliminando posibles retardos muestrales que esta pudiese tener.

La salida de esta fase de procesado es un índice muestral mediante el cuál se informa al estimador de canal donde comienza la señal DVB-T. Solamente se ejecuta una única vez para cada señal DVB-T, a diferencia del estimador de sincronismo fino que se ejecutaría por cada símbolo OFDM de la señal y cuyo fundamento e implementación serán analizados en el capítulo posterior.

3.1. Diagrama de bloques y explicación

El algoritmo de estimación de símbolo grueso aprovecha la presencia de los prefijos cíclicos o de guardia al inicio de cada símbolo. La longitud de este prefijo es determinada por el estándar. Debido a sus características repetitivas y periódicas al inicio de cada símbolo, realizando una operación de auto correlación sobre la señal resaltará la localización en muestras de los prefijos de guardia. Se tendrán tantos picos de correlación como símbolos OFDM se encuentren presentes en la señal DVB-T [11].

La estimación de inicio de la señal podría tomarse directamente del primer pico de correlación (cuya posición coincidiría con el primer prefijo circular). Sin embargo,

si solamente es tenida en cuenta este primer prefijo cíclico la estimación perdería precisión. Como la longitud de símbolo OFDM total es conocida por el estimador sería conveniente utilizar únicamente este primer símbolo OFDM y su prefijo, pero si este símbolo presenta algún retardo temporal dentro del propio símbolo útil la estimación realizada será poco precisa. Con el objetivo de aumentar la precisión del estimador de símbolo grueso, se propone realizar un promediado con un número determinado de picos de auto correlación o prefijos cíclicos. Este número de símbolos a tomar es un parámetro de diseño y su impacto en la estimación será analizado posteriormente.

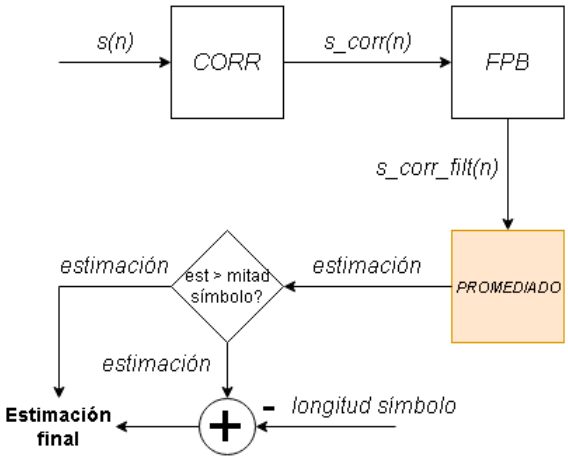


Figura 3.1: Algoritmo estimación de símbolo grueso

Estos picos de auto correlación previamente descritos pueden ser observados en la figura *3.2*, como ha sido mencionado anteriormente la distancia de estos picos es aproximada a la longitud total del símbolo OFDM. Se realiza un filtrado paso bajo para delimitar el número deseado de prefijos cíclicos a utilizar en la estimación de símbolo grueso (el criterio de la elección del valor será analizado posteriormente).

En esta misma figura se puede observar los dos siguientes pasos en el proceso, que consisten en la superposición y adición de los diferentes picos de correlación. Después de la adición se observan dos picos al inicio y al final de la señal creada. El último pico es causa de la propia superposición y no debe ser empleado en la estimación. Por tanto solamente se toman aquellos picos que se encuentren por debajo de la mitad de la longitud de símbolo total, puesto que cada símbolo OFDM solo presenta un único prefijo circular al inicio.

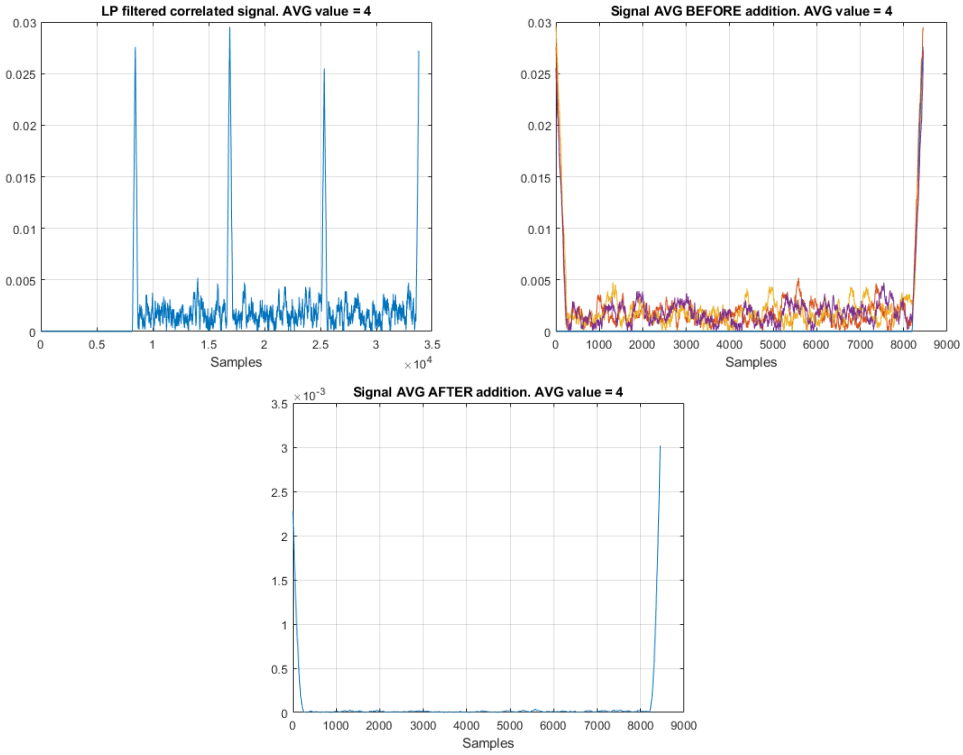


Figura 3.2: Estimación de símbolo grueso

En este caso debido a las dimensiones reducidas del código empleado se considera oportuno mostrar este a modo de esclarecer posibles dudas. La implementación realizada en Matlab del algoritmo de estimación de símbolo grueso se puede ver en el Anexo A.

3.2. Resultado de simulaciones

La cuestión fundamental acerca de la fase de estimación de símbolo grueso es la selección del valor de promediado apropiado. En la figura *3.5* se observa el resultado de simulaciones realizadas con diferentes valores de retardo aplicados. Se relaciona el error cuadrático medio en muestras con el valor de relación señal a ruido de la señal introducida para diferentes valores de promediado. La elección del valor de promediado debe realizarse seleccionando un valor máximo de error RMSE de muestras

(representado en la figura *3.5* mediante una línea negra discontinua) y relacionándose este con el valor de SNR esperado en la señal de entrada.

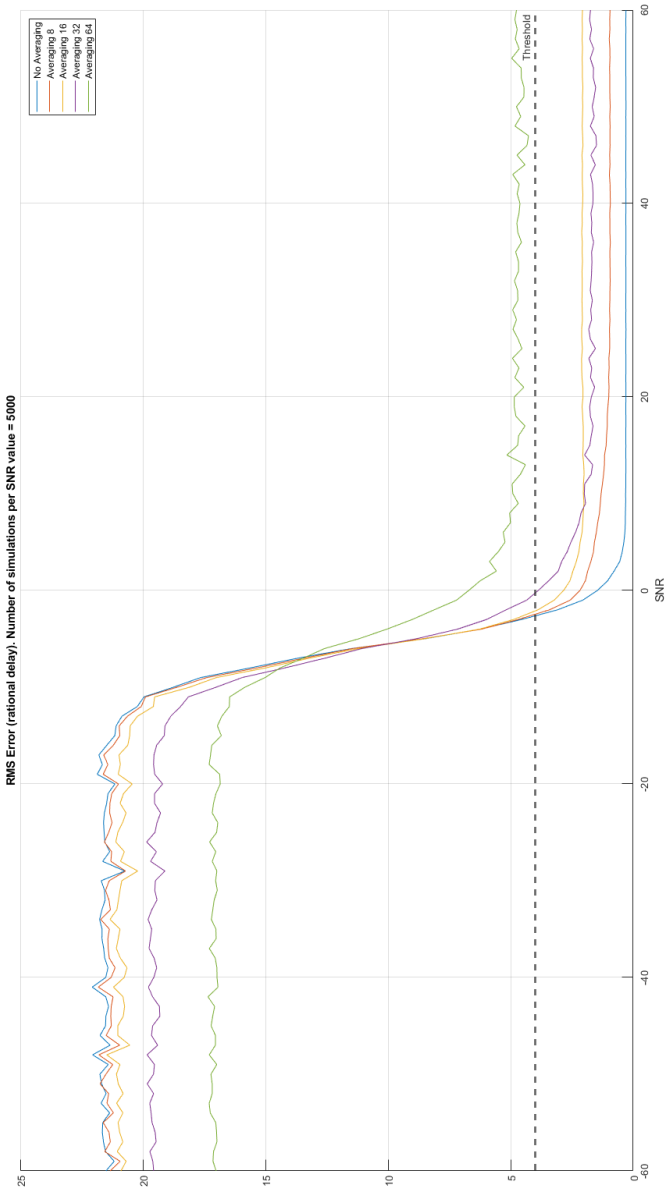


Figura 3.3: RMSE vs SNR valores de promediado

Capítulo 4

Estimación de canal

En el capítulo tres se ha tratado el algoritmo de ambigüedad cruzada, donde se ha podido observar como este precisa de dos señales de entrada, una señal proveniente del contacto con el objetivo (denominada como señal de eco) y otra señal obtenida del iluminador de oportunidad utilizado (señal de referencia). En el diseño del receptor planteado en la introducción al proyecto solamente se contempla un único canal de recepción, por lo que las señales eco y referencia se encuentran mezcladas en una única señal recibida. La siguiente fase de procesado que se presentará en este capítulo tiene la importante función de correctamente diferenciar y separar las señales anteriormente mencionadas.

4.1. Disección del estimador diseñado

El algoritmo diseñado de estimación de canal se muestra en el siguiente diagrama de bloques. En rojo y verde se muestran las memorias de los símbolos y respuestas al canal respectivamente. Las dos memorias utilizadas en este algoritmo permiten una correlación y promediado con respuestas al canal anteriores, aumentando la precisión de la estimación. A partir del índice de comienzo de señal obtenido en el estimador de símbolo grueso discutido en el capítulo anterior, en primer lugar se elimina el prefijo cíclico del símbolo a evaluar. Este prefijo será posteriormente añadido en la fase de reconstrucción de la señal.

A continuación se realiza la transformada de Fourier del símbolo útil (símbolo sin prefijo circular) con el objetivo de trabajar en el dominio de la frecuencia, puesto que las posiciones de los diferentes pilotos utilizados en la estimación son dadas en el dominio espectral. Este símbolo es almacenado en una memoria de símbolos (bloque

en rojo), con capacidad para un número de símbolos igual al valor de promediado escogido. Si embargo este método no considera los primeros $N-1$ símbolos en la primera estimación de la respuesta al canal ya que la memoria no ha terminado de llenarse, siendo N el valor de promediado escogido.

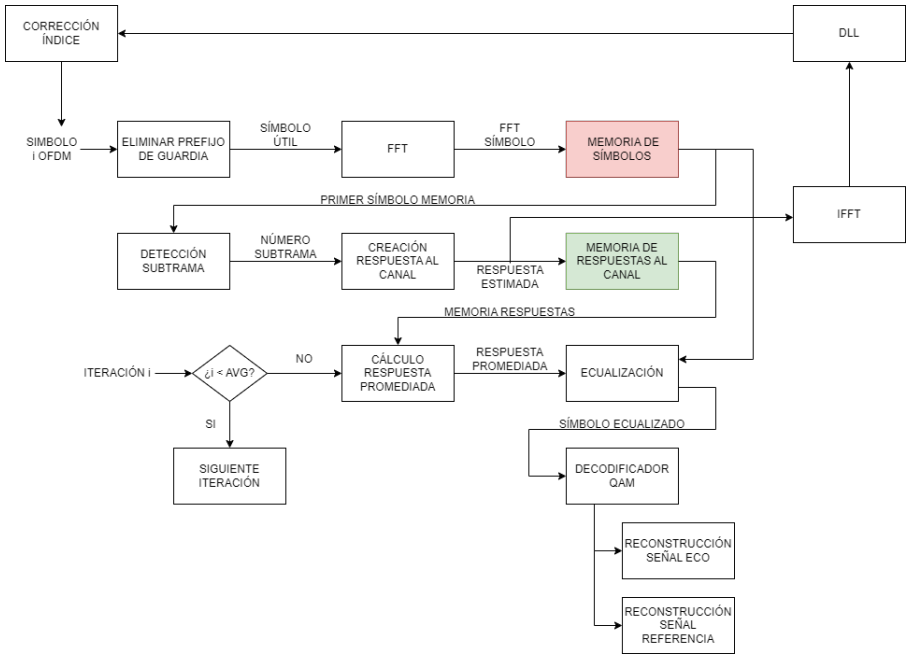


Figura 4.1: Algoritmo estimación de canal

4.1.1. Detección de subtrama

Despúes del almacenamiento del símbolo en la memoria, se extrae de esta el símbolo situado en la primera posición (análogo a una pila) y se realiza la detección de la subtrama presente en ese símbolo OFDM. En DVB-T se definen hasta cuatro tipos diferentes de subtrama OFDM, en lo que a términos del procesado se refieren, en cada una de ellas el posicionamiento de los pilotos dispersos es distinto. Es necesario previo a la estimación de la respuesta del canal determinar qué tipo de subtrama es para asociar correctamente los índices de estas portadoras dispersas. El funcionamiento de este algoritmo de detección se ilustra en el siguiente diagrama.

Consiste en probar para cada símbolo útil las cuatro posibles distribuciones de posiciones de subtrama OFDM. Se multiplican los valores esperados si el símbolo

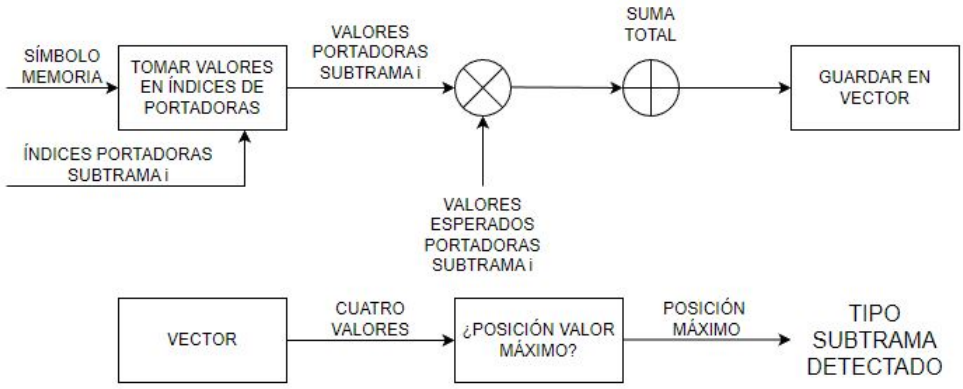


Figura 4.2: Detección de subtrama

fuese de una cierta subtrama por los valores en estos índices del símbolo, posteriormente se suman estos valores y se almacena este resultado. Tras realizar este proceso con cada uno de los cuatro tipos, simplemente se comprueba cuál de ellos es el valor máximo. La posición de este valor dentro del vector indica el tipo de subtrama.

4.1.2. Estimación de canal

Tras determinar el tipo de subtrama OFDM (y por tanto los índices de las portadoras dispersas a emplear), se realiza la estimación de la respuesta al canal. Los dos elementos fundamentales empleados para estimar esta respuesta son las portadoras piloto (cuyas posiciones son fijas y determinadas por el estándar y conocidas para todo símbolo OFDM) y las portadoras dispersas (cuyos grupo de índices ya han sido escogido). En la figura *4.3* se muestra el diagrama de bloques del algoritmo de estimación de canal.

En primer lugar, se multiplican los valores del símbolo OFDM en los índices de los diferentes pilotos (tanto continuos como dispersos) por el inverso de su valor esperado. Después de sumar (teniendo en cuenta el índice dentro de la respuesta al canal) se obtiene una primera respuesta al canal parcial, la cual se muestra en la figura *4.4*. Esta respuesta, aunque válida, es incompleta, como se puede ver en esta misma figura. Esto se debe a la generación de las posiciones en frecuencia o *bins* de los pilotos (que como es lógico no cubren todos los índices), fácilmente remediable a través de una operación de desplazamiento o *fftshift* en sintaxis de Matlab.

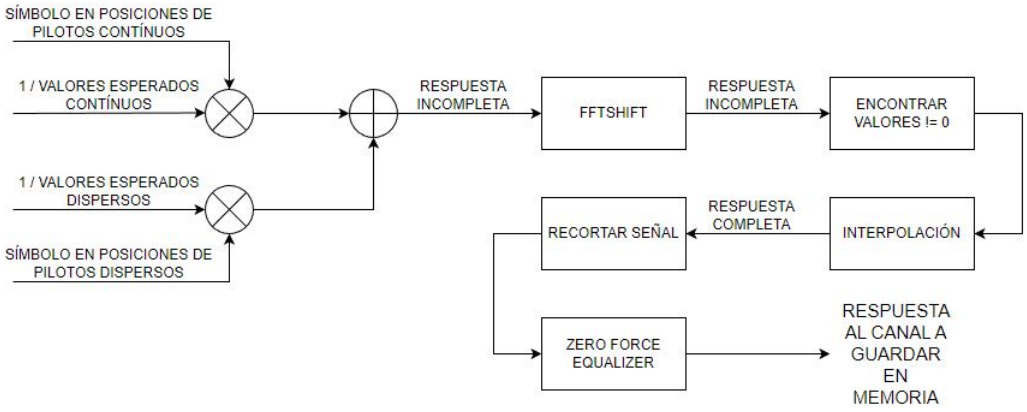


Figura 4.3: Algoritmo estimador de respuesta al canal

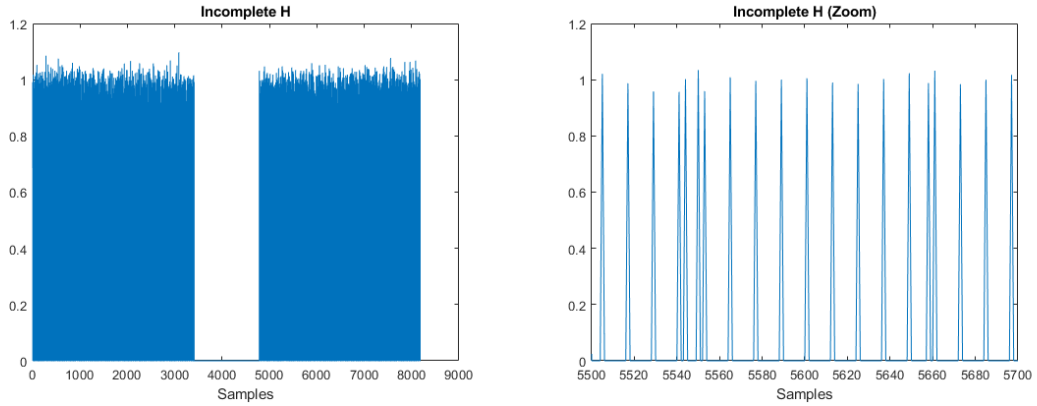


Figura 4.4: Respuesta al canal incompleta

Para completar la respuesta al canal se interpola la respuesta parcial para obtener una respuesta en todas los *bins* de frecuencia y no solamente en aquellas de los pilotos. Tras esta operación, tomando solamente aquellos valores de interés para la respuesta (la longitud de un símbolo útil) y teniendo la precaución de que ninguno de estos sea un valor nulo (aplicándose un Zero Force Equalizer), la respuesta obtenida posee el siguiente aspecto (figura *4.5*).

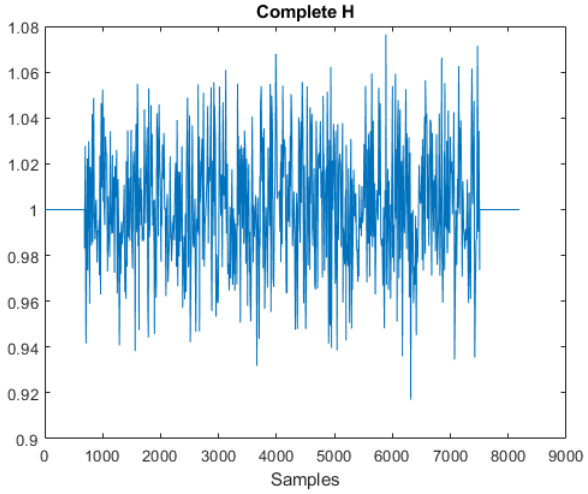


Figura 4.5: Respuesta al canal completa

Esta respuesta es luego almacenada en la memoria de respuestas al canal, que será utilizada para el cálculo de una respuesta promediada. Solamente se procede al cálculo de estas respuestas promediadas cuando la memoria ha logrado llenarse con un número AVG respuestas al canal intermedias. Este valor AVG se trata de un parámetro de diseño, determina el valor de promediado y el número de respuestas al canal intermedias que se promedian cada vez. La respuesta promediada se calcula según la siguiente expresión.

$$H_{AVG} = \frac{\sum_{n=1}^{n=AVG} H_{MEMORIA}}{AVG} \tag{4.1}$$

4.1.3. Sincronismo de símbolo fino. DLL

Después de realizar la estimación de canal del símbolo OFDM entrante, se procede a la siguiente fase denominada como sincronismo de símbolo fino. El sincronismo fino se refiere a la precisión y estabilidad de la sincronización de la señal o símbolo entrante, se enfoca en ajustar pequeñas variaciones en el tiempo. Mientras que el sincronismo grueso (estudiado en el capítulo anterior) se refiere a la alineación inicial de la señal en el tiempo. En este caso el algoritmo empleado para realizar este sincronismo fino se denomina *Delay Locked Loop (DLL)*, este es similar a a un *Phase Locked Loop (PLL)*, con la ausencia de un *Voltage Controlled Oscilator (VCO)* que es reemplazado por una línea de retardo [12].

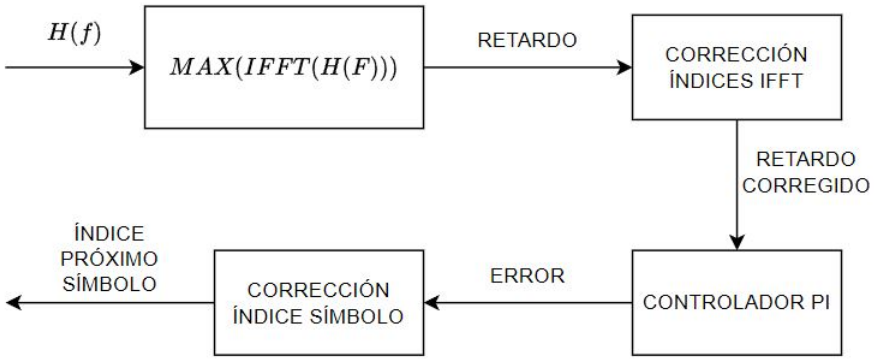


Figura 4.6: Esquema DLL

El *DLL* utiliza como entrada la estimación de canal realizada en esa iteración (no se emplea la respuesta promediada a partir de varias estimaciones puesto que arrastraría mayor error que si se realizase de forma individual) y se compone de un detector de retardo, una memoria donde se almacena la línea de retardo, un controlador Proporcional Integral (PI) realizado con un filtro IIR de segundo orden y un corrector acumulativo que finalmente indicará como salida del algoritmo el índice donde se tomará el siguiente símbolo OFDM en la próxima iteración.

En primer lugar se da el detector de retardo, que consiste en tomar el índice donde se encuentre el máximo valor de la $IFFT$ aplicada a la estimación de canal del símbolo OFDM. El resultado de esta transformada inversa es una delta en el dominio muestral cuyo índice indica el retardo observado en la señal. Este índice es posteriormente corregido aplicando una operación de desplazamiento *ifftshift* o bien teniendo en cuenta el desplazamiento sufrido por la señal y ajustando el índice obtenido.

La siguiente fase consta de un Proporcionador Integral (PI). Se trata de un tipo de controlador formado por un controlador proporcional (P) y un controlador inte-

gral (I). El objetivo principal de un PI es corregir retardos del sistema y obtener una línea de retardo que logre mantenerse lo más cercana a un retardo nulo.

El controlador proporcional (P) toma en cuenta el error actual entre la salida deseada y la salida real del sistema y genera una señal de control proporcional a ese error. La acción proporcional es directamente proporcional al error, lo que significa que cuanto mayor sea el retardo, mayor será la acción correctiva aplicada.

El controlador integral (I) tiene en cuenta tanto el retardo actual como los retardo pasados y acumula la integral de los errores a lo largo del tiempo. Genera una señal de control proporcional a la integral de los retardos acumulados. La acción integral ayuda a eliminar los retardos constantes o de estado estacionario que no pueden ser corregidos solo con la acción proporcional [13].

Finalmente se aplica la acción correctiva con el error estimado, teniéndose en cuenta para la próxima iteración del algoritmo. En la siguiente figura se puede observar la línea de error y su evolución a medida que el algoritmo procesa cada vez más símbolos entrantes. Se puede observar como la línea de error trata de mantenerse estable en un valor nulo de error y para ello cal comienzo del algoritmo aplica fuertes acciones correctivas para tratar de estabilizar el error (esto es visible entre los símbolos OFDM uno y veinte). Entre las muestras veinte y cuarenta la corrección decrece en intensidad para finalmente a partir de aproximadamente el símbolo número setenta lograse el enganche del *DLL* para el resto de los símbolos, donde el algoritmo ha determinado que ya no es necesario aplicar correcciones al índice de selección de los símbolos.

4.1.4. Decodificación QAM y reconstrucción

Una vez calculada esta respuesta promediada, se procede a la ecualización del símbolo, para finalmente decodificar y reconstruir este símbolo OFDM. Siendo X la transformada de Fourier del símbolo útil.

$$X_{EQ} = \frac{X}{H_{AVG}} \tag{4.2}$$

Tras obtenerse el símbolo ecualizado, ahora es necesario realizar la reconstrucción del mismo. Para ello se realizará en primer lugar la decodificación QAM y reconstrucción de este símbolo OFDM, para posteriormente construir las señales de eco y referencia. El esquema de demodulación y reconstrucción de símbolos es el siguiente.

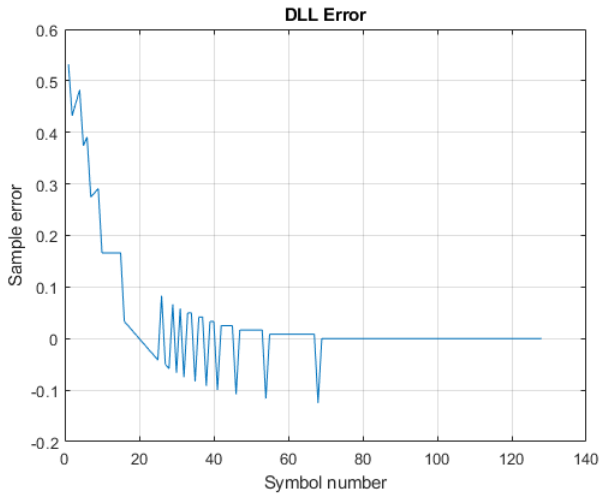


Figura 4.7: Línea de retardo DLL

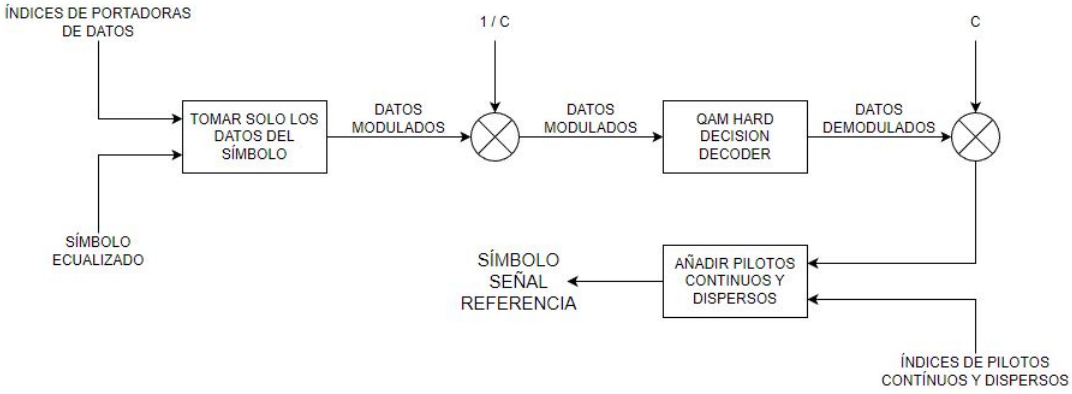


Figura 4.8: Decodificador QAM y reconstrucción de símbolo

En primer lugar, se toman del símbolo ecualizado solamente aquellos elementos que son datos modulados en QAM, esto es posible ya que estas posiciones son definidas por el estándar. A continuación se aplica un factor $1/C$ a estos datos, se trata de un factor de normalización (también definido por el estándar). En este caso al aplicarse el inverso del factor se está desnormalizando los datos para su poste-

rior demodulación. Tras esta des-normalización se procede a la decodificación de los símbolos QAM. Esta es realizada a través de un *Hard Decision Decoder*. El de este decisor puede encontrarse en el código del estimador de canal en el Anexo A.

Al detectarse un valor mayor o menor a $\pm\sqrt{M}$, este es corregido al máximo valor esperado en la demodulación, que es $\pm\sqrt{M}$. Es decir, este decisor solamente se centra en aquellos valores demasiado grandes o pequeños para poder ser resultado de la modulación QAM (de ahí que se conozca como *Hard Decisor*) [14]. Tras demodular y reconstruir los símbolos QAM, se vuelve a normalizar de nuevo los datos y finalmente se introducen los pilotos contínuos y dispersos, obteniéndose el símbolo OFDM reconstruido.

El último paso es la reconstrucción de las señales de eco y referencia, donde, entre otras operaciones, se añadirá el prefijo circular eliminado al inicio de la estimación. La reconstrucción de las señales se realiza para cada símbolo, y el algoritmo se ilustra en los siguientes diagramas de bloques.

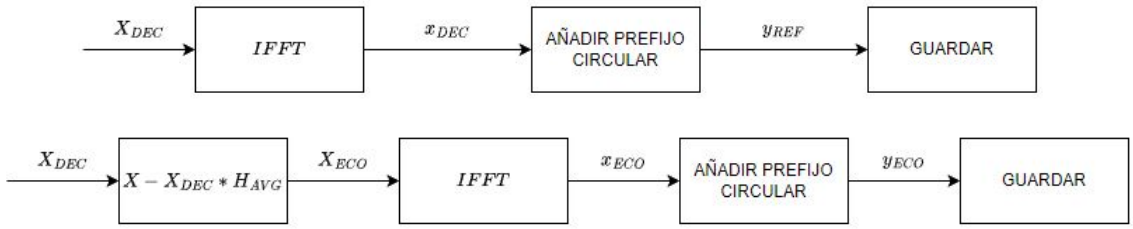


Figura 4.9: Reconstrucción señales eco y referencia

Del decodificador QAM se obtiene el símbolo de la señal de referencia, por tanto la reconstrucción de este para la creación de la señal de referencia es directa (IFFT y concatenación del prefijo de guardia). Sin embargo, para la señal de eco es necesaria una operación previa. Se puede entender el símbolo inicial como una del símbolo de la señal de eco y la señal de referencia, de forma que:

$$X = X_{ECO} + X_{REF} \tag{4.3}$$

Por tanto el símbolo de la señal de eco se puede obtener simplemente despejando esta ecuación. Tal que:

$$X_{ECO} = X - X_{REF} \tag{4.4}$$

El símbolo de referencia es estimable gracias a cálculos previos, conocida la respuesta al canal y el símbolo de referencia decodificado, este puede ser expresado de la siguiente forma:

$$X_{ECO} = X - X_{DEC} * H_{AVG} \tag{4.5}$$

Los siguientes pasos en la reconstrucción son análogos a el caso de la señal de referencia. Estos resultados intermedios son almacenados. Tras procesarse todos los símbolos estos se concatenarán en orden para forma las señales completas reconstruidas.

4.2. Comparación con el estimador previo

El trabajo anterior presentaba un estimador de canal radicalmente diferente al presentado anteriormente en este documento, las diferencias más fundamentales son las siguientes.

- El estimador anterior solamente utilizada las portadoras piloto para la estimación de canal. Lo que reduce la precisión del estimador.
- La forma de onda empleada en el trabajo anterior no se ajustaba en detalle al estándar DVB-T.
- No se realizaba un promediado intermedio en el cálculo de las estimaciones de canal, lo que de nuevo reduce la precisión de la estimación final.

En la siguiente figura se ofrece el error RMS de reconstrucción de la señal de referencia para diferentes valores de SNR en ambos estimadores, demostrando que los cambios realizados reducen en gran medida el error observado.

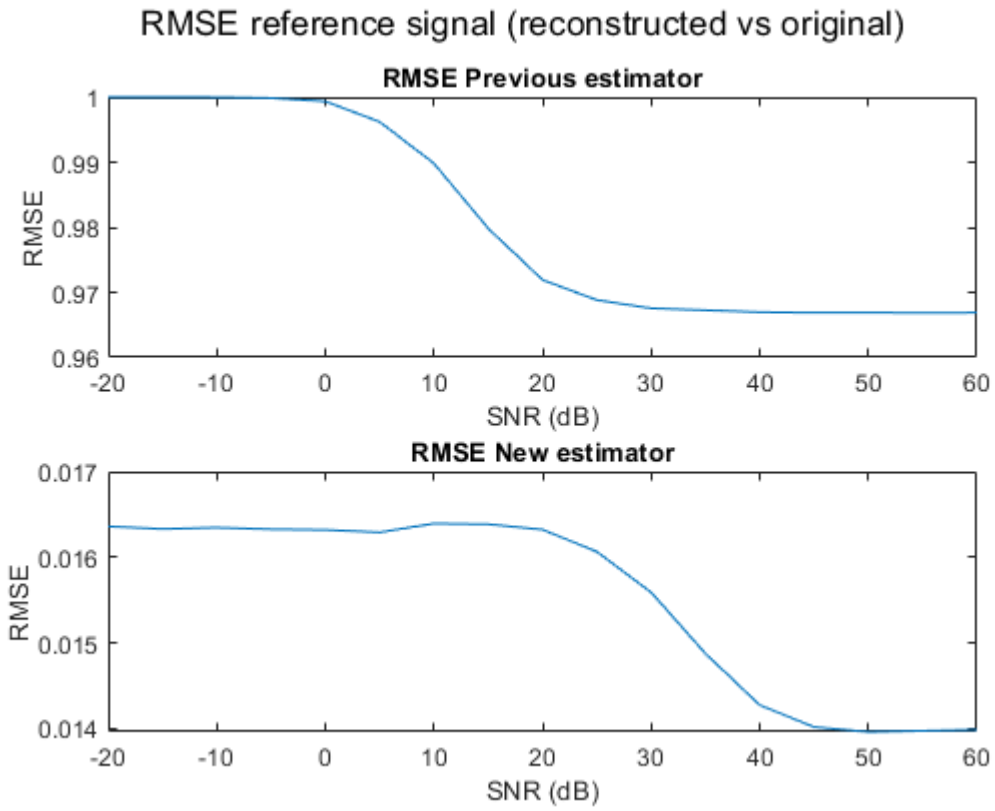


Figura 4.10: Comparación estimadores

Capítulo 5

Detección y estimación de parámetros

Resultado de capítulos anteriores se ha obtenido un mapa de rango y velocidad biestáticos con posibles detecciones de objetivos reales acompañadas de otros elementos como ruido y clutter. Es necesario realizar un procesado posterior de este mapa con el objetivo de discernir con cierto criterio entre aquellos objetivos confirmados como verídicos y otros picos de correlación que podrían resultar ruido u objetivos inválidos.

En este capítulo se tratará esta fase del procesado RADAR, la cual es la responsable de determinar si un pico de correlación visto en la matriz resultado de la CAF se trata de una detección real. Esta decisión es tomada comparando el valor de este pico con un umbral previamente calculado a partir de celdas adyacentes del propio mapa de correlación. En esto se centrarán los dos primeros apartados del capítulo [15].

Si este umbral es sobrepasado, el pico de correlación bajo evaluación es considerado un objetivo válido. La separación de este objetivo del resto del mapa generará una "nube de puntosçorrespondiente a un único objetivo, lo cuál hace que sin un procesado y refinamiento posterior la estimación del rango y la velocidad biestáticas sea arbitrario y erróneo. Posibles algoritmos que han sido implementados para el posterior refinado de esta nube serán analizados en los dos últimos apartados del capítulo [16].

5.1. CFAR

El algoritmo CFAR *(Constant False Alarm Rate)* posee diferentes variantes y opciones de implementación (véase [17]). En este caso dado que el mapa utilizado posee dos dimensiones (velocidad y distancia), se ha optado por emplear la variante del algoritmo CFAR conocida como Cell Averaging CFAR *CA-CFAR*.

El algoritmo CA-CFAR es capaz de detectar objetivos en presencia de ruido y clutter, ya que utiliza la media de las energías de las celdas adyacentes como referencia para determinar si hay un objetivo presente en la celda de interés (denominada en la literatura RADAR con *Cell Under Test (CUT)*). Además, el algoritmo es capaz de mantener una tasa de falsa alarma constante ajustando el umbral de detección en función del nivel de ruido y clutter en la señal. El número de celdas de referencia es tomado considerando un margen de guardia, que no es tenido en cuenta en la estimación debido a su cercanía a la posible detección. Si estas fueran tomadas en cuenta, el umbral crecería en valor, y es posible que una detección verdadera fuera catalogada como una detección inválida.

Lo explicado se puede observar a continuación en la figura *5.1*, donde se observan las celdas de referencia (de color verde) las celdas de guardia (en color azul) y finalmente la *CUT* coloreada en rojo. El área de búsqueda o de referencia para la creación del umbral (cuya expresión matemática es mostrada en la propia figura *5.1*) solamente considera estas celdas verdes.

La lógica seguida por el algoritmo implementado es la que se muestra en la figura 5.2. La parte fundamental de este algoritmo es un doble bucle que recorre la matriz tomando la *CUT* además de las celdas de guardia y referencia.

Inicialmente se toma la matriz de resultados de la *CAF* y se le aplica una operación de *padding*. Esto consiste en cubrir la matriz de detecciones con un valor para poder realizar de forma más correcta la evaluación de las detecciones situadas en los bordes de la matriz (es decir en las primeras y últimas filas y columnas de esta). En concreto se optó por emplear la media de la propia matriz *CAF* como valor de relleno. El motivo de esto es tratar de simular de la forma más aproximada el suelo de ruido presente, ya que esto permitirá calcular los umbrales de las estimaciones borde de forma más precisa.

Paralelo a este paso se realiza también el cálculo de la constante *alfa*. Esta cons-

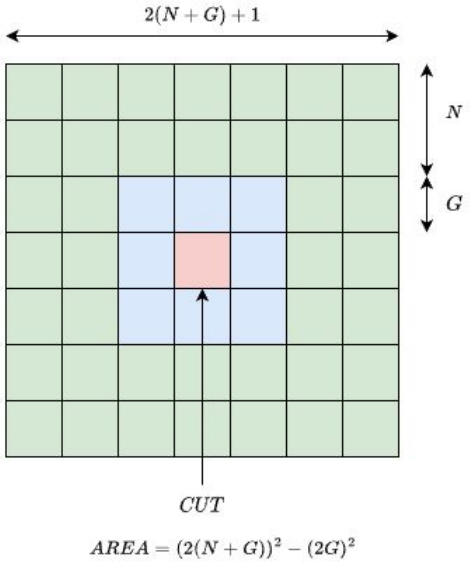


Figura 5.1: CA CFAR

tante es calculada a partir de una probabilidad de falsa alarma o falsa detección impuesta por el diseño y el número de celdas de referencia empleadas para el cálculo del umbral. La expresión matemática es la siguiente.

$$\alpha = N * (P_{FA}^{-1/N} - 1) \tag{5.1}$$

Donde N son el número de celdas de guardia y P_{FA} la probabilidad de falsa alarma [18]. Tras el cálculo de la constante α comienza el cálculo de los umbrales a partir de las celdas de referencia. La expresión matemática para esta operación es la siguiente.

$$train_{AVG}(i,j) = \frac{\sum_{i-N-G,j-N-G}^{i+N+G,j+N+G} Y(i,j) - 2Y(i,j) - \sum_{i-G,j-G}^{i+G,j+G} Y(i,j)}{(2(N+G))^2 - (2G)^2} \tag{5.2}$$

El umbral de detección es el resultado de la siguiente operación. Este valor se comparará con aquel de la celda *CUT*. Si el valor de la matriz CAF supera a este umbral, se considera una detección válida y su posición (i, j) es guardada para posterior análisis.

$$U = train_{AVG} * \alpha \tag{5.3}$$

El resultado o salida de este algoritmo es un mapa con una serie de grupos o *clusters* de detecciones. Esto se puede ver en la figura *5.3*, en este caso se trata de

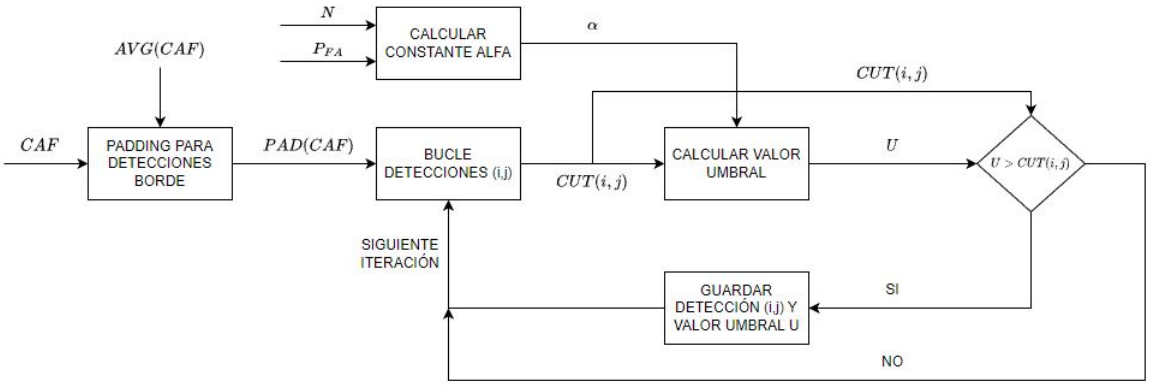


Figura 5.2: CA CFAR Algoritmo

una única detección. Gráficamente hablando, se puede entender este resultado como la aplicación de una *máscara* a la matriz resultante de la CAF [19]. En la figura *5.4* se pueden observar los umbrales empleados.

La característica forma de anillo de esta máscara es atribuible a el empleo de celdas de guardia. En estas figuras se ve claramente su función, pues deben evitar que *la detección se filtre a sí misma*, es decir, evitar que valores de detecciones positivas sean usados para realizar el cálculo del umbral. Si esto sucediese, este umbral aumentaría, corriéndose el riesgo de darse un falso positivo (la detección era real pero ha sido descartada por el algoritmo).

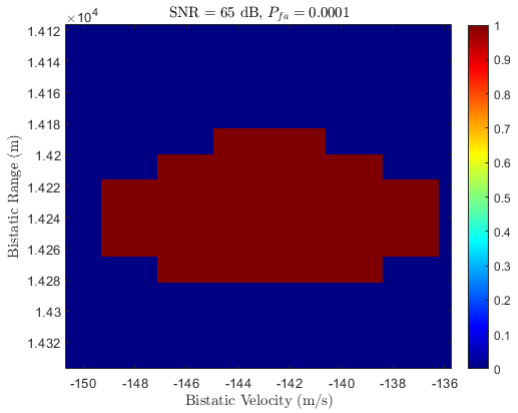


Figura 5.3: CA CFAR Grupo de positivos (una única detección)

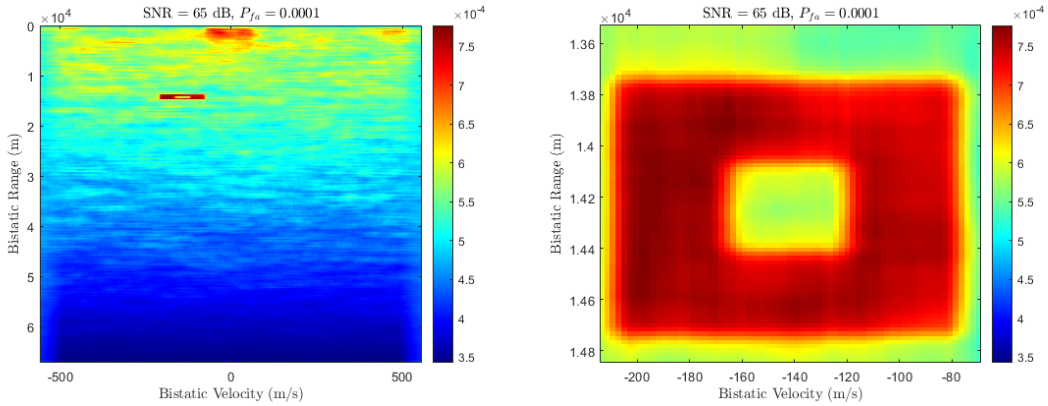


Figura 5.4: CA CFAR Umbral de detección

5.2. Selección del número de celdas de referencia y guardia

La selección razonable del número de celdas de evaluación o referencia y el número de celdas de guardia juega un papel extremadamente importante en la detección. Una mala elección podría llevar a casos de falsos positivos o negativos lo que provocaría un funcionamiento deficiente del RADAR. Los parámetros deben ser escogidos teniendo en cuenta el tipo de objetivos que se espera detectar, en concreto influyen elementos como la huella dejada por una detección positiva en la función de ambigüedad cruzada, el tiempo de computación, la sensibilidad de la detección...

Respecto al número de celdas de guardia que deben situarse alrededor de la celda *CUT*, el elemento más importante a tener en cuenta es la huella de las detecciones esperadas en la CAF. La detección realizada se puede entender como un lóbulo pronunciado sobre el suelo de ruido del mapa rango velocidad con un ancho concreto en ambas dimensiones. El criterio seguido para determinar el valor de G adecuado consiste en tomar el ancho mayor de los dos posibles valores.

En las siguientes figuras se puede observar los lóbulos de una detección realizada. En este punto es necesario establecer a qué nivel se debe tomar el ancho del haz. En este caso, siguiendo con la filosofía expuesta en la elección de la ventana, donde se prioriza un mayor ancho de ventana con el objetivo de lograr una mejor reducción de los lóbulos secundarios a cambio de perderse sensibilidad en la detección de objetivos cercanos, se ha optado por tomar el valor de ancho como la diferencia en celdas de los puntos donde el haz se encuentra en su anchura máxima (es decir, en la base del mismo). El ancho máximo observable en la figura *5.5* es de aproximadamente diez celdas, por lo que un valor mínimo de cinco celdas de guardia podría utilizarse

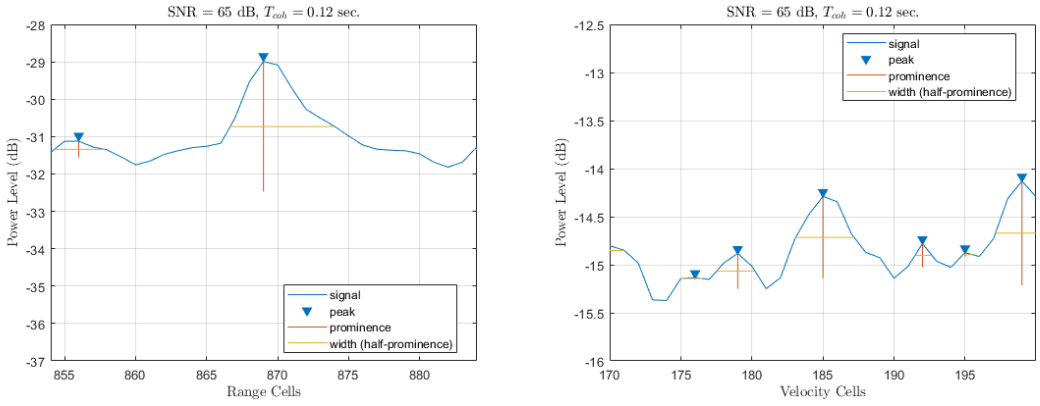


Figura 5.5: Ancho de detección

La selección del número de celdas de referencia ha sido realizada teniendo en cuenta la velocidad de procesado del algoritmo y la potencia de detección perdida. A un mayor valor de celdas de referencia a tener en cuenta para el cálculo de la energía media de ruido, mayor tiempo es necesario para completar el algoritmo de CFAR en su totalidad, sin embargo se da una menor pérdida de potencia de detección. Un número pequeño de celdas de referencia implica una menor precisión en el cálculo del umbral de detección del algoritmo CFAR aunque una menor complejidad.

Por tanto se debe encontrar un equilibrio entre velocidad de procesado y precisión de la estimación del umbral realizada (potencia de detección perdida). Analizando la siguiente figura se observa como una mayor probabilidad de falsa alarma (que según se ha visto anteriormente implica un umbral de detección menos exigente). Por lo que en función de la probabilidad de falsa alarma escogida, el valor máximo de pérdida de potencia de detección permitido y la máxima complejidad computacional determinarán el valor a escoger del número de celdas de referencia.

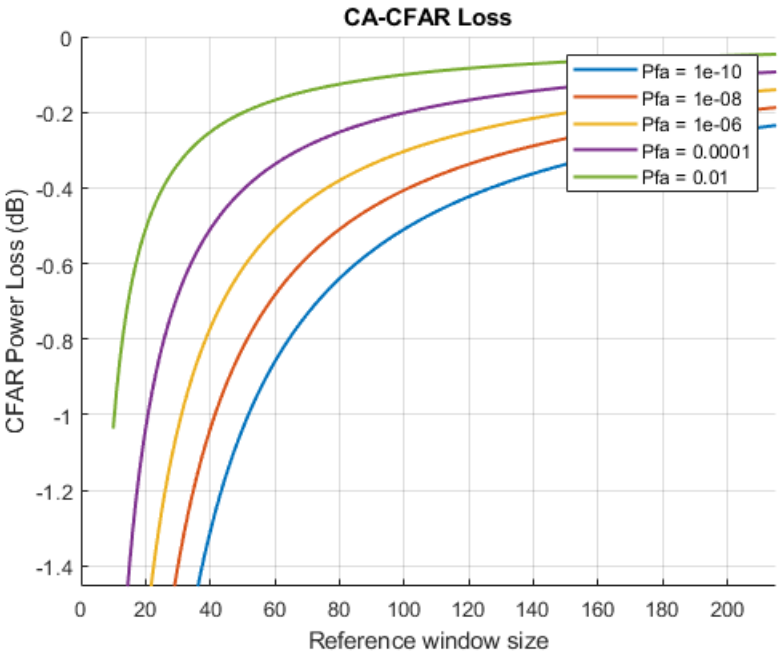


Figura 5.6: Pérdida de potencia de detección [15]

5.3. Estimación fina distancia - Doppler

Tras la aplicación del algoritmo CA-CFAR en dos dimensiones, se obtiene una agrupación de puntos relacionada a una única o varias detecciones válidas. Idealmente, para una única detección se espera un único par de valores biestáticos reales, sin embargo realmente se suelen obtener otras detecciones alrededor de este valor real (como se puede observar en la figura *5.3* en la sección anterior).

5.3.1. Aproximación polinómica

Es necesario traducir este grupo de detecciones a un único par de valores distancia-velocidad biestáticos. Un posible método consiste en aproximar los valores obtenidos a una curva polinómica de orden 2 (es decir, una parábola). Aunque la aproximación podría realizarse en un orden mayor, logrando una mejor aproximación de los valores obtenidos en la detección, el empleo de la parábola simplifica el cálculo y ofrece una única raíz o estimación final [20].

El algoritmo de estimación se muestra en el siguiente diagrama. Este algoritmo toma la salida del algoritmo de detección visto en la sección anterior y procede a agrupar los diferentes grupos de detecciones confirmadas.

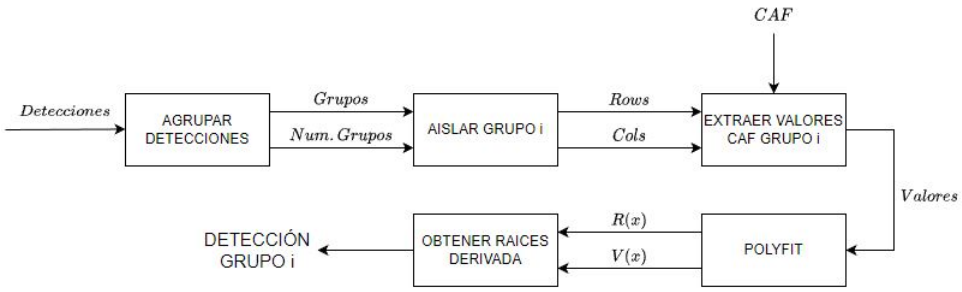


Figura 5.7: Agrupación y filtrado de detecciones

Tras realizar estas agrupaciones, conocido el número de grupos existentes se itera por todos estos aplicando las siguientes operaciones.

- En primer lugar, se aísla las detecciones del grupo a tratar. Esto se hace para evitar conflictos con otros grupos de detecciones en pasos posteriores. Se extraen las posiciones en forma de vectores fila y columna de este grupo.

- A continuación, se emplean estos valores de filas y columnas para extraer los valores de la CAF de interés.
- Estos valores son utilizados para crear una función polinómica de orden dos que aproxime a estos valores. Se crean dos polinomios, uno por cada dimensión.
- Finalmente, para obtener el máximo de estas funciones creadas (y por tanto el índice de detección definitivo) se obtienen la raíz del polinomio derivado, es decir, el índice del máximo valor de la función.

El resultado final son dos índices, uno para cada dimensión que indican la detección final que debe ser utilizada en el algoritmo de seguimiento. En las siguientes figuras se muestra la aplicación de esta idea en las dimensiones de velocidad y distancia.

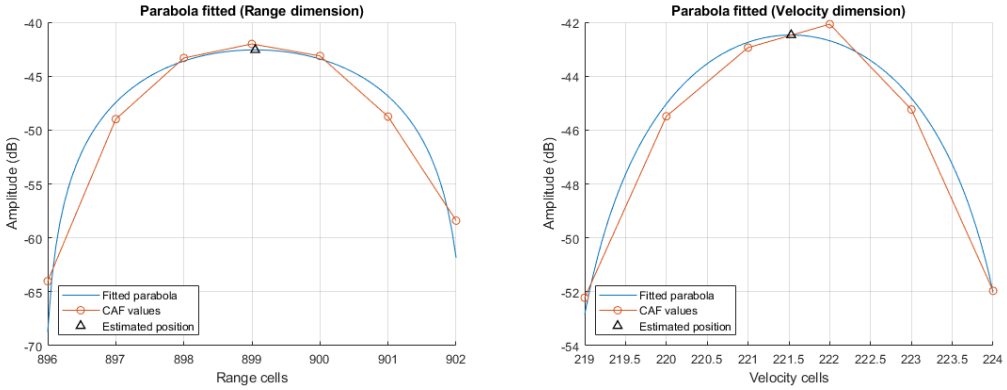


Figura 5.8: Estimación y aproximación de los valores

En este ejemplo se obtienen los índices o celdas 899 en la dimensión de la distancia y aproximadamente 222 en la dimensión de la velocidad. Traduciendo estos a valores de distancia y velocidad biestáticos empleando las ecuaciones (3.5) y (3.6) se puede concluir que el objetivo detectado se encuentra a aproximadamente una distancia de 14,20Km y se desplaza a una velocidad de 515Km/h.

Sin embargo, este método presenta incongruencias cuando se trata con agrupaciones de puntos muy próximas entre sí, además de que una aproximación polinómica requiere un mayor tiempo de operación y cuidado con los valores suministrados que otras operaciones más sencillas, lo que resulta inviable en simulaciones donde se trabaje con más de un objetivo. Aunque teóricamente este método es viable, carece de razones suficientes para sustituir al método que se presentará a continuación.

5.3.2. Centro de masas ponderado

Un forma más sencilla y directa es tomando los diferentes grupos de detecciones y ponderar las coordenadas obtenidas con el valor de la energía del mapa de rango-velocidad biestáticos. En la figura *5.9* se puede ver un esquema del algoritmo realizado.

En primer lugar se realiza una primera agrupación, similar al método de la aproximación polinómica. Sin embargo el objetivo no es el de discernir entre diferentes grupos, se busca realizar una primera separación con el objetivo de determinar la agrupación más pequeña de puntos correspondientes a una detección. Este valor, denominado como número mínimo de vecinos permitirá una mayor precisión en la posterior separación de grupos.

A continuación se computan las distancias entre todas las detecciones con el objetivo de encontrar el par de celdas individuales más lejanas. Este valor se empleará como la distancia máxima entre posibles detecciones pertenecientes a un mismo grupo. Otra posible interpretación es la distancia mínima entre dos agrupaciones o nubes de puntos que corresponden a un objetivo válido.

Los dos primeros pasos realizados tienen como objetivo ofrecer los valores de mínimo número de celdas vecinas y distancia máxima entre celdas pertenecientes a un mismo objetivo más acertados para la realización de *Density-Based Spatial Clustering of Applications with Noise (DBSCAN)*. Dicho algoritmo será explicado en mayor detalle en el siguiente capítulo. Este será el responsable de la división final de agrupaciones de detecciones.

Una vez que el algoritmo DBSCAN haya logrado separar las coordenadas de velocidad y rango biestáticos correspondientes a un único objetivo, se procede a realizar el cálculo del centro de masas ponderado, ofreciéndose finalmente un único par de valores de velocidad y distancia biestáticas por nube de detecciones u objetivo válido detectado.

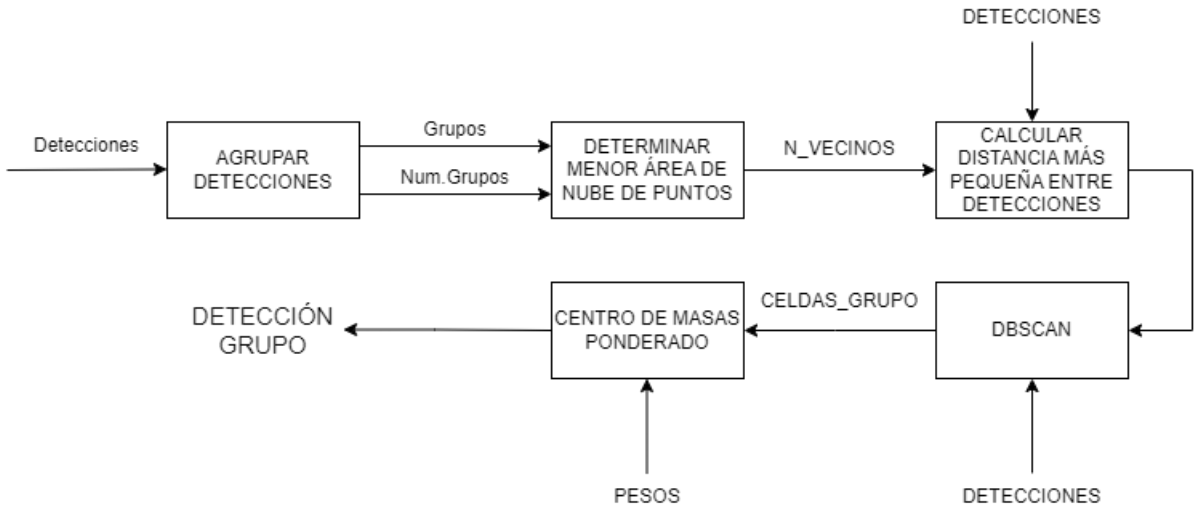


Figura 5.9: Centro de masas ponderado

Capítulo 6

Tracking y seguimiento

En el capítulo anterior se estudió el tratamiento de las detecciones individuales realizado por el algoritmo *CFAR*. El siguiente paso lógico es determinar como tratar múltiples salidas de este CFAR, es decir, ¿cómo se podría determinar si dos detecciones distintas corresponden o no al mismo objetivo o realidad física? Es necesario establecer una relación entre diferentes detecciones que en realidad muestren el movimiento de un único objetivo, esto además permitirá predecir la próxima posición donde este objetivo podría aparecer en el próximo ciclo. Este seguimiento permite la creación de una trayectoria plausible en dimensiones de velocidad y distancia biestáticas comúnmente denominada en la literatura RADAR como *traza*. La creación de estas trazas será discutida en la primera parte del capítulo. Tras obtenerse esta traza, a partir de los datos de movimientos obtenidos se realizará la predicción de la próxima posición del objetivo, lo cual se expondrá en mayor detalle en la segunda parte de este capítulo.

6.1. Creación de trazas

Se considera una traza a un conjunto de observaciones previas las cuales han sido catalogadas como pertenecientes a un objetivo o realidad física única, es decir, se deben construir tantas trazas como diferentes objetivos hayan sido detectados. Se puede interpretar la creación de trazas desde dos puntos de vista.

- Asignación lógica de observaciones. Agrupar aquellas observaciones que físicamente posean sentido. A partir de una distancia en celdas máxima permitida en el algoritmo se establece una zona alrededor de la última detección del objetivo donde su próxima detección es posible en el próximo ciclo.

- Algoritmos de clustering o agrupamiento. Emplear algoritmos de clustering para la agrupación de nubes de puntos. Posterior a la aplicación de algoritmos se podría verificar si físicamente las trazas creadas tienen sentido.

En este trabajo, se ha optado por una combinación de ambas opciones. A partir de un valor máximo de aceleración previamente establecido en la fase de diseño (este valor debe ser escogido en función de las características y naturaleza de los objetivos que se espera detectar) mediante el cual se calcula una distancia máxima teniendo en cuenta el CPI y las resoluciones en velocidad y distancia del algoritmo que luego es utilizada por algoritmo de agrupación o clustering. En concreto el algoritmo empleado es *Density-Based Spatial Clustering of Applications with Noise*. Se trata de un algoritmo de agrupamiento utilizado para unir puntos de datos cercanos en una región de alta densidad.

DBSCAN posee dos parámetros fundamentales, el radio de búsqueda y el número de vecinos mínimo. En este caso este radio de búsqueda es la distancia máxima calculada previamente, mientras que el número mínimo de vecinos establece la información o número de detecciones mínimas necesarias para inicializar una traza y catalogar las detecciones como pertenecientes a un único objetivo. Los puntos que quedan fuera de este radio se consideran como detecciones de objetivos nuevos de los cuales no se posee suficiente información para iniciar una traza y son tratados como observaciones individuales [21]. Aplicar este algoritmo al procesado RADAR posee bastante sentido debido a las siguientes razones.

- El radio de búsqueda tiene en cuenta la distancia máxima que el objetivo puede recorrer en un CPI, por tanto para aumentar la capacidad de asignación de trazas únicamente es necesario variar la aceleración máxima permitida.

- El número mínimo de vecinos permite establecer un tamaño mínimo de traza. En el trabajo realizado, se considerará como traza a una agrupación mínima de tres observaciones.

- Aquellas observaciones no incluidas en una traza no son eliminadas y pueden mantenerse en el conjunto de datos a la espera de más observaciones cercanas.

- Se podrían utilizar otros algoritmos como *k-means clustering*, sin embargo la diferencia fundamental con el empleado es la necesidad de conocer el número de clusteres a crear, lo que requeriría poseer inteligencia externa que permitiera previamente ajustar el algoritmo al número de trazas u objetivos esperado.

Además, respecto al algoritmo *k-means*, *DBSCAN* presenta una asociación más lógica de puntos obtenidos en un contexto RADAR. Es esperable que un objetivo siga un camino acorde a sus capacidades en términos de aceleración y maniobrabilidad, si este objetivo por ejemplo en dos observaciones independientes realizadas presentase valores carentes de sentido (como por ejemplo un desplazamiento y aceleraciones excesivos o maniobras imposibles) implicaría que la asociación de objetivos y creación de trazas no ha sido correctamente implementada. En la siguiente figura se observa un ejemplo visual de lo explicado.

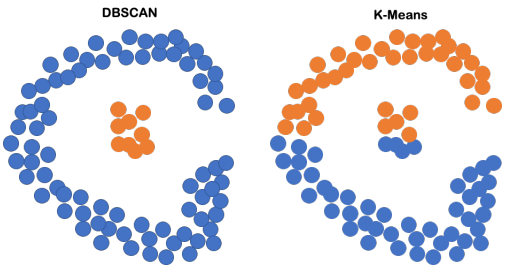


Figura 6.1: *DBSCAN* vs Kmeans [22]

En la figura 7.2 se muestra la creación de dos trazas, es decir dos objetivos diferentes, los cuales muestran unas distancias y velocidades similares pero moviéndose en posiciones opuestas respecto a la localización del receptor.

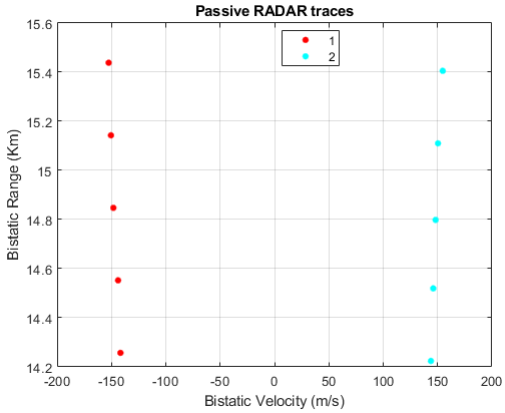


Figura 6.2: Trazas

6.2. Seguimiento. Filtro de Kalman

Un RADAR utiliza estos algoritmos de predicción, basados en las trazas previamente estudiadas, para estimar la posición futura de un objeto en movimiento. Entre los diferentes algoritmos de predicción que existen, en este proyecto se ha optado por emplear el filtro de Kalman [23]. El modelo de predicción de movimiento se puede condensar en la siguiente expresión.

$$x(k+1) = Fx(k) + u(k) \tag{6.1}$$

Siendo x el vector columna de posiciones con los valores de distancia, velocidad y aceleración del objetivo, k el índice de la posición a analizar, F es la matriz de transición de estados y u es el ruido en la predicción. Esta última matriz es determinante para la sensibilidad y precisión del modelo.

Las señales empleadas en el procesamiento RADAR son continuas, sin embargo solamente es posible procesar aquellas muestras que entre en el tiempo de un *CPI*, lo que obliga a separar la señal en bloques individuales de longitud en el tiempo T con o sin solapamiento. La formación de la matriz de transición previamente descrita debe tener en cuenta la duración temporal de estos bloques. Considerando que un movimiento acelerado *MRUA* se puede describir mediante las siguientes ecuaciones:

$$R(k+1) = R(k) + V(k)T + A(k)\frac{T^2}{2} \tag{6.2}$$

$$V(k+1) = V(k) + A(k)T \tag{6.3}$$

Colocando estas expresiones en forma de ecuación matricial, se obtiene el modelo de predicción detallado previamente descrito en la introducción de esta sección.

$$\begin{bmatrix} R(k+1) \\ V(k+1) \\ A(k+1) \end{bmatrix} = \begin{bmatrix} 1 & T & \frac{T^2}{2} \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R(k) \\ V(k) \\ A(k) \end{bmatrix} + u(k) \tag{6.4}$$

Este primer modelo ofrece una estimación adecuada del objetivo, sin embargo carece de la suficiente complejidad para realizar predicciones de objetivos con mayor capacidad de maniobra. Una posible forma de añadir irregularidad al modelo es mediante una mejor estimación del ruido y una matriz de transición con mayor capacidad de adaptabilidad al objetivo [23]. La evolución del modelo descrito anteriormente es la siguiente.

$$P(k+1) = FP(k)F' + Q \tag{6.5}$$

Donde P es una matriz de covarianza de estados que va obteniendo mayor cantidad de información a medida que se procesan más posiciones y Q es una matriz de ruido de predicción que tiene en cuenta la longitud temporal T. Las predicciones son realizadas a partir de la matriz de observación H cuyo valor es fijo y transforma la predicción en un vector. En este caso la matriz H es la siguiente. El valor concreto de esta matriz permite extraer del vector de posiciones conocidas la distancia y velocidad, excluyendo la aceleración.

$$H = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{6.6}$$

La predicción realizada con la posición entrante, la cuál será denominada como $z_n(k)$ es la siguiente.

$$z_n(k+1) = Hx(k+1) \tag{6.7}$$

Cuando se obtiene una nueva posición, se emplea para actualizar la matriz de covarianza P y crear un nuevo vector de innovación $v(k)$, el cual almacena las diferencias entre la predicción anterior y la predicción actual.

$$v(k+1) = z(k+1) - z_n(k+1) \tag{6.8}$$

Paralelamente, con la actualización de la matriz de covarianza de estados P se realiza también la actualización de la matriz de covarianza de predicciones S, esta se empleará posteriormente para obtener la ganancia del filtro que se debe aplicar para calcular la predicción definitiva.

$$S(k+1) = HP(K+1)H' + R \tag{6.9}$$

Siendo R una matriz que contiene las varianzas del cálculo de la distancia y la velocidad. A continuación se procede a calcular la ganancia del filtro K.

$$K = P(K+1)H'S^{-1}(k+1) \tag{6.10}$$

La predicción final se obtiene a partir de esta ganancia y del vector de innovación calculado previamente.

$$x_n = Fx(k) + Kv(k+1) \tag{6.11}$$

Matlab ofrece una librería que integra el filtrado Kalman [24] cuya implementación se puede encontrar en el Anexo H. El código trata de realizar una predicción en base al modelo creado y la longitud de bloque, esta predicción es luego corregida al entrar una nueva posición.

En la siguiente figura se puede observar de forma más visual el trabajo realizado por el filtro de Kalman. El algoritmo realiza el ciclo de predicción - detección - corrección. En primer lugar se produce una predicción inicial (simbolizadas por las aspas rojas en la figura). En el momento que entra la detección verdadera (puntos negros en la figura) se realiza la corrección respecto a la posición predicha y la real (círculo amarillo), realimentando el modelo.

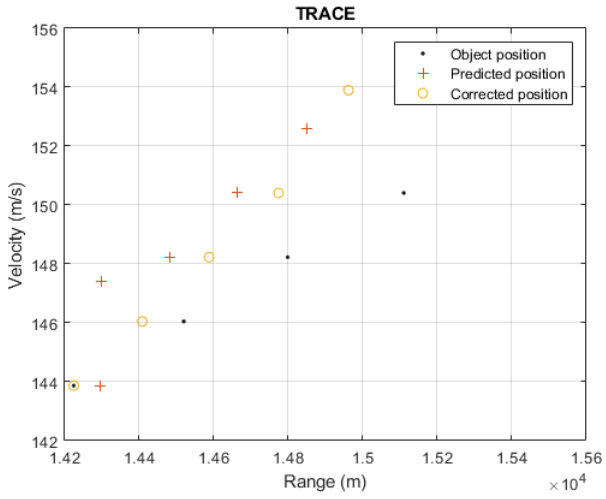


Figura 6.3: Traza con predicción

Otra forma de comprender el funcionamiento de este algoritmo según [25], para cada posición obtenida se crea una región o *association gate* donde podría darse la próxima detección del objetivo en el próximo ciclo del algoritmo (próximo CPI). Esta región es definida en base al vector de innovación *v(k)* y la matriz de covarianza de predicciones *S(k)*.

$$v'(k)S^{-1}(k)v(k) \leq \lambda \tag{6.12}$$

Donde λ es un parámetro escogido en la fase de diseño teniendo en cuenta las capacidades de los objetivos que se prevé detectar en la zona (velocidad máxima, capacidad de maniobra, tamaño...). Si la siguiente observación se encuentra dentro del área de búsqueda o región predeterminada, la covarianza en la próxima región disminuirá, por lo que esta será más exigente. Si sucediese lo contrario (la detección cae fuera de la región establecida), el algoritmo aumentará la cantidad de ruido o varianza introducida en el cálculo de la región, por lo que esta será más flexible.

Capítulo 7

Conclusión del trabajo

En este trabajo se han explorado técnicas de procesado RADAR ya conocidas, como pueden ser el algoritmo de Batch, la estimación de canal, el procesado y reconstrucción de la señal DVB-T, además de ofrecerse nuevas y diferentes aproximaciones al campo, como es la doble aplicación del algoritmo DBSCAN teniendo en cuenta la limitación física del algoritmo o la previa estimación del tamaño de la agrupación de puntos obtenidos del algoritmo CA-CFAR con el objetivo de una mejor separación de detecciones diferentes.

Como se comentó en la introducción, este trabajo es continuación de otro. Este trabajo ofrecía una propuesta para el algoritmo de Batch y la estimación de canal. A partir de este, se procedió a realiza mejoras mínimas en el algoritmo de Batch y se rediseñó por completo la estimación de canal. Una vez realizado este trabajo se diseñaron el estimador de símbolo grueso, el algoritmo CA-CFAR, la creación de trazas y agrupación de objetivos y finalmente la predicción de nuevas posiciones mediante filtrado de Kalman.

Como continuación de este trabajo una posibilidad sería la realización de pruebas de campo empleando hardware como la tarjeta ADALM PLUTO. Ya que en este trabajo solo se ofrecen resultados a partir de simulaciones cuyo ruido y condiciones son simuladas por código.

Apéndice A

Alineación del proyecto con los ODSs

Los ODSs, o Objetivos de Desarrollo Sostenible, son una serie de metas establecidas por las Naciones Unidas para abordar los desafíos globales y promover un desarrollo sostenible en todo el mundo. Constan de 17 objetivos interrelacionados.

ODS	**Descripción**	**Aplica**	**Justificación**
1	Fin de la pobreza	NO	
2	Hambre cero	NO	
3	Salud y bienestar	NO	
4	Educación de calidad	SI	Trabajo académico
5	Igualdad de género	NO	
6	Agua limpia y saneamiento	NO	
7	Energía asequible y no contaminante	NO	
8	Trabajo decente y crecimiento económico	SI	Trabajo de investigación y desarrollo
9	Industria, innovación e infraestructura	SI	Aportes a la industria
10	Reducción de las desigualdades	NO	
11	Ciudades y comunidades sostenibles	NO	
12	Producción y consumo responsables	NO	
13	Acción por el clima	NO	
14	Vida submarina	NO	
15	Vida de ecosistemas terrestres	NO	
16	Paz, justicia e instituciones sólidas	SI	Aplicable en Defensa
17	Alianzas para lograr los objetivos	NO	

Apéndice B

Código estimador de símbolo grueso

```
% Coarse Symbol estimation
s_x = conj([s_rx; zeros(Tu,1)]).*[zeros(Tu,1); s_rx];      % Correlation function
s_x = conv(s_x, ones(CP,1));                               % Signal smoothing
s_x = s_x(1:4*(Tu+CP));                                    % Adjust signal length
A = reshape(s_x,Tu+CP,length(s_x)/(Tu+CP));                % Matrix adjustement
estimate = sum(abs(A).^2,2);                               % Averaging

[~,peak_index] = max(estimate);                            % Obtain first estimation

if (peak_index > (Tu+CP)/2)                                % Correct fftshift
    peak_index = peak_index - (Tu+CP);
end
```

Apéndice C

Código estimador de canal

```
H = complex(zeros(1,Tu));
Y = zeros(Tu,1);

%Initialization phase
if (i < AVG)
    % Displace memory and add new symbol
    symb_memory(:,2:end) = symb_memory(:,1:end-1);
    symb_memory(:,1) = s_rx(round(peak_index)+CP+1:min(round(peak_index
        )+Tu+CP,length(s_rx)));
    % Symbol fft
    X = fft(symb_memory(:,1),Tu);

    % OFDM subframe type detection
    scatter_values = zeros(1,size(scatt_pilot_carriers,1));
    for j=1:size(scatt_pilot_carriers,1)
        scatter_values(j) = abs(sum(X(scatt_pilot_carriers_aux(j,:)+1)
            .*scatt_pilot_inf(j,:)'));
    end
    [~,subframe_number] = max(scatter_values);

    % Initial channel estimation (incomplete estimation)
    Y(cont_pilot_carriers_aux+1) = cont_pilot_inf;
    Y(scatt_pilot_carriers_aux(subframe_number,:)+1) = scatt_pilot_inf(
        subframe_number,:);
    H(cont_pilot_carriers_aux+1) = X(cont_pilot_carriers_aux+1)./Y(
        cont_pilot_carriers_aux+1);
    H(scatt_pilot_carriers_aux(subframe_number,:)+1) = X(
        scatt_pilot_carriers_aux(subframe_number,:)+1)./Y(
        scatt_pilot_carriers_aux(subframe_number,:)+1);
```

```
    % Complete channel estimation (fill null and zero gaps)
    % Shift to avoid bin allocation problems
    H = fftshift(H);
    % Find reference points for interpolation
    carriers = find(abs(H)~=0);
    % Interpolate the whole useful symbol length
    H = interp1(carriers,H(carriers),1:Tu,'linear','extrap');
    % Keep valid channel estimation
    H = H(min(carriers):max(carriers));

    % Accomodate length so its equal to symbol length
    H = [zeros(1,round(((Tu-length(H))/2)-1)), H];
    H = [H, zeros(1,Tu-length(H))];

    % Zero Force Equalizer
    H(H==0)=1;

    % DLL
    [~,phase_error] = max(ifft(H));
    phase_error = phase_error-1;

    if phase_error >= (Tu)/2
        phase_error = phase_error-Tu;
    end

    v(d) = (K2*phase_error + mem);
    mem(:) = v(d);
    v(d) = v(d) + K1*phase_error;
    peak_index = peak_index + Tu+CP + v(d);
    d = d+1;

    % Channel memory displacement and introduction of new estimation
    H_memory(:,2:end) = H_memory(:,1:end-1);
    H_memory(:,1) = H';

% Once filter is 75% full
else
    symb_memory(:,2:end) = symb_memory(:,1:end-1);
    symb_memory(:,1) = s_rx(round(peak_index)+CP+1:min(round(peak_index
        )+Tu+CP,length(s_rx)));
    X = fft(symb_memory(:,1),Tu);
    scatter_values = zeros(1,size(scatt_pilot_carriers,1));

    for j=1:size(scatt_pilot_carriers,1)
        scatter_values(j) = abs(sum(X(scatt_pilot_carriers_aux(j,:)+1)
            .*scatt_pilot_inf(j,:)'));
```

```
end

[~,subframe_number] = max(scatter_values);
Y(cont_pilot_carriers_aux+1) = cont_pilot_inf;
Y(scatt_pilot_carriers_aux(subframe_number,:)+1) = scatt_pilot_inf(
    subframe_number,:);

H(cont_pilot_carriers_aux+1) = X(cont_pilot_carriers_aux+1)./Y(
    cont_pilot_carriers_aux+1);
H(scatt_pilot_carriers_aux(subframe_number,:)+1) = X(
    scatt_pilot_carriers_aux(subframe_number,:)+1)./Y(
    scatt_pilot_carriers_aux(subframe_number,:)+1);

H = fftshift(H);
carriers = find(abs(H)~=0);
H = interp1(carriers,H(carriers),1:Tu,'linear','extrap');
H = H(min(carriers):max(carriers));
H = [zeros(1,round(((Tu-length(H))/2)-1)), H];
H = [H, zeros(1,Tu-length(H))];
H(H==0)=1;

[~,phase_error] = max(ifft(H));
phase_error = phase_error-1;

if phase_error >= (Tu)/2
    phase_error = phase_error-Tu;
end

v(d) = (K2*phase_error + mem);
mem(:) = v(d);
v(d) = v(d) + K1*phase_error;
peak_index = peak_index + Tu+CP + v(d);
d = d+1;

H_memory(:,2:end) = H_memory(:,1:end-1);
H_memory(:,1) = H';

% Calculate averaged response
Havg = sum(H_memory,2)./AVG;

% Equalization
Yeq = X./Havg;

% QAM decoder
Ydec = complex(zeros(Tu,1));
% Prepare signal structure
```

```
    data_carriers = zeros(Tu,1);
    data_carriers(data_carriers_indexes+1) = 1;
    data_carriers(cont_pilot_carriers_aux+1) = 0;
    data_carriers(scatt_pilot_carriers_aux(subframe_number,:)+1) = 0;
    data_carriers(tps_carriers+1) = 0;
    % Denormalise QAM values
    data_demmod = round(Yeq(logical(data_carriers))./c);

    % QAM Hard decision decoder
    data_demmod(real(data_demmod)>(sqrt(M))) = (sqrt(M)) + 1j*imag(
        data_demmod(real(data_demmod)>(sqrt(M))));
    data_demmod(real(data_demmod)<(-1*sqrt(M))) = (-1*sqrt(M)) + 1j*
        imag(data_demmod(real(data_demmod)<(-1*sqrt(M))));
    data_demmod(imag(data_demmod)>(sqrt(M))) = 1j*(sqrt(M)) + real(
        data_demmod(imag(data_demmod)>(sqrt(M))));
    data_demmod(imag(data_demmod)<(-1*sqrt(M))) = (-1j*sqrt(M)) + real(
        data_demmod(imag(data_demmod)<(-1*sqrt(M))));

    % Normalise values
    Ydec(logical(data_carriers)) = data_demmod*c;
    % Add continual and scatter pilots
    Ydec(cont_pilot_carriers_aux+1) = cont_pilot_inf;
    Ydec(scatt_pilot_carriers_aux(subframe_number,:)+1) =
        scatt_pilot_inf(subframe_number,:);

    % Reference signal reconstruction
    y_rx = ifft(Ydec);
    y_rx_cp = [y_rx(end-CP+1:end); y_rx];
    y_dp(:,n) = y_rx_cp;

    % Echo signal reconstruction
    Yobs = X-Ydec.*Havg;
    y_obs = ifft(Yobs);
    y_obs_cp = [y_obs(end-CP+1:end); y_obs];
    y_oc(:,n) = y_obs_cp;

    % Move n index to allocate reconstructed symbols
    n = n+1;
end
```

Apéndice D

Código algoritmo de Batch

```
%Adjust signal length to be divisible by number of blocks
s_rx = [s_rx(:); zeros(L-mod(length(s_rx),L),1)];
s_tx = [s_tx(:); zeros(length(s_rx)-length(s_tx),1)];
% Eliminate DC component
s_rx = s_rx - mean(s_rx);
s_tx = s_tx - mean(s_tx);
% Reshape signal into blocks of length L
s_rx = reshape(s_rx,L,length(s_rx)/L);
s_tx = reshape(s_tx,L,length(s_tx)/L);
%Number of blocks created
cols = size(s_rx,2);
Y = zeros(L,cols);
% FFT Length
Lfft_aux = 2^(nextpow2(2*L-1));
y_aux = zeros(Lfft_aux,1);

for i = 1:cols
    % Cross correlation
    y_aux = ifft(fft(s_rx(:,i),Lfft_aux).*conj(fft(s_tx(:,i),Lfft_aux))
        ,Lfft_aux);
    % Interpolation and frequency windowing
    y_aux = ifft(ifftshift([zeros(ceil((length(y_aux)*(factor_interp
        -1))/2),1) ; fftshift(fft(y_aux,length(y_aux))).*factor_interp.*
        chebwin(length(y_aux)) ; zeros(ceil((length(y_aux)*(
        factor_interp -1))/2),1)]),length(y_aux));
    Y(:,i) = y_aux(1:L);
end

% Time windowing
K = 2^(nextpow2(cols));
```

```
Y = [Y, zeros(L,K-cols)];
w = chebwin(cols);
w = w(:)/sum(w);
w = [w; zeros(K-cols,1)];
% Map preparation
Y = Y.*repmat(w.',L,1);
Y = fftshift(fft(Y,K,2),2);
```

Apéndice E

Código CA-CFAR

```
%CA-CFAR
for i = (N+G+1):(R-N-G)
    for j = (N+G+1):(D-N-G)
        % Calculate train average value from reference cells
        train_avg = (sum(sum(Y(i-N-G:i+N+G, j-N-G:j+N+G))) - 2*sum(Y(i,
            j))) - sum(sum(Y(i-G:i+G, j-G:j+G))))/((2*(N+G))^2-(2*G)^2);
        % Calculate threshold
        threshold = alpha * train_avg;
        % Compare detection threshold to CUT
        if Y(i,j) > threshold
            detections(i,j) = Y(i,j);
        end
    end
end

%Find smallest point cloud
[groups,num_groups] = bwlabel(detections);
minpnt = 0;
for i=1:num_groups
    if minpnt < length(find(groups==1))
        minpnt = length(find(groups==1));
    end
end

% DBSCAN
[range,velocity] = find(detections);
cells = [range,velocity];
distance = pdist2(cells,cells,'euc','Smallest',minpnt);
labels = dbscan(cells,max(distance,[],"all"),minpnt);
```

```
% Weighted center mass
for t=1:max(labels)
    trace = cells(labels==t,:);
    index = sub2ind(size(detections),trace(:,1),trace(:,2));
    weights = detections(index);
    point_cloud(ceil(sum(weights.*trace(:,1))/sum(weights)-N-G),ceil(
        sum(weights.*trace(:,2))/sum(weights)-N-G)) = s;
end
```

Apéndice F

Código creación de trazas y predicción

```
if nnz(point_cloud) ~= 0
    [range,velocity] = find(point_cloud);
    cells = [range,velocity];
    % Maximum euclidean distance calculation
    max_cell_distance = round((((max_acceleration*(CPI^2))/2)*2*fs)/3e8
        );
    max_cell_velocity = (max_acceleration*CPI)/2.1798;
                                % Fixed velocity resolution
    euc_distance = sqrt(max_cell_distance^2+max_cell_velocity^2);
    % DBSCAN
    traces = dbscan(cells,euc_distance,min_detections_for_trace);
end

for t=1:max(traces)
    if t ~= -1
        % Detection sorting
        kalman_trace = cells(traces==t,:);
        order = point_cloud(sub2ind(size(point_cloud),kalman_trace(:,1)
            ,kalman_trace(:,2)));
        kalman_trace = sortrows([order,kalman_trace]);
        kalman_trace(:,1) = [];
        % Kalman filter
        kalman_trace(:,1) = kalman_trace(:,1).*3e8/(2*fs);
        kalman_trace(:,2) = ((kalman_trace(:,2)-1)/2^nextpow2(cols)
            -0.5)*(fs/L)*3e8/(2*f);
        kalman(kalman_trace,CPI,max_acceleration,1,2.1798,SNR);
    end
end
```

Apéndice G

Parámetros DVB-T simulación

```
BW = 8;                          % Channel selection 5,6,7,8 (Mhz)
n_symb = 128;                    % Number of symbols
fs = (64/7)*1e6;                 % Sampling frequency (Hz)
tx_mode = '8K';                  % Transmitter mode '2K', '4K', '8K'
frame_offset = 0;                % Frame offset
guard = 1/32;                    % Guard interval length
T_symb = 8192;                   % Useful symbol time in samples
CP = guard*T_symb;               % Guard length in samples
mod_type = '64-QAM';             % Symbol modulation
M = 64;                          % Modulation order
alpha = 2;                       % Normalization factor
T = (n_symb*(T_symb + CP))/fs;   % Signal duration (in seconds)
```

Apéndice H

Código Kalman

```
T = CPI;                                % block length
q = 0.01;                               % power spectral density of process noise
sigmaR = 0.1;
sigmaV = 0.00001;
sigmaA = 0;

H = [1 0 0 ; 0 1 0];
F = [1 T T^2/2 ; 0 1 T ; 0 0 1];
Q = q*[T^5/20 T^4/8 T^3/6 ; T^4/8 T^3/3 T^2/2 ; T^3/6 T^2/2 T];
R = [sigmaR 0 ; 0 sigmaV];
P = [sigmaR ; sigmaV ; sigmaA];
initialState = [positions(1,1); positions(1,2); 0];

KF = trackingKF('MotionModel','Custom',...
     'StateTransitionModel',F,...
     'MeasurementModel',H,...
     'ProcessNoise',Q,...
     'MeasurementNoise',R,...
     'ControlModel',P,...
     'State',initialState);

for k = 1:size(positions,1)
     predictions(k,:) = predict(KF,T);
     corrections(k,:) = correct(KF,positions(k,:));
end
```

Bibliografía

[1] Mateusz Malanowski. *Signal Processing for Passive Bistatic Radar*. 685 Canton Street, Norwood, MA 02062: Artech House London, 2019, págs. 131-132. ISBN: 978-1-63081-662-9.

[2] DVB. *ETSI EN 300 744. Digital Video Broadcasting (DVB). Framing structure, channel coding and modulation for digital terrestrial television*. 650 Route des Lucioles, France: DVB, 2015, págs. 25-27. ISBN: 348-6-235620-001-7.

[3] DVB. *ETSI EN 300 744. Digital Video Broadcasting (DVB). Framing structure, channel coding and modulation for digital terrestrial television*. 650 Route des Lucioles, France: DVB, 2015, págs. 28-30. ISBN: 348-6-235620-001-7.

[4] K. E. Olsen H. Kuschel D. Cristallini. *Tutorial: Passive radar tutorial*. Norway: IEEE Aerospace y Electronic Systems Magazine, 2019, págs. 6-7. DOI: 10.1109/maes.2018.160146.

[5] Mateusz Malanowski. *Signal Processing for Passive Bistatic Radar*. 685 Canton Street, Norwood, MA 02062: Artech House London, 2019, págs. 37-38. ISBN: 978-1-63081-662-9.

[6] Frederic J. Harris. *On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform*. Norway: IEEE, 1978. DOI: 10.1109/PROC.1978.10837.

[7] Mateusz Malanowski. *Signal Processing for Passive Bistatic Radar*. 685 Canton Street, Norwood, MA 02062: Artech House London, 2019, pág. 134. ISBN: 978-1-63081-662-9.

[8] Mateusz Malanowski. *Signal Processing for Passive Bistatic Radar*. 685 Canton Street, Norwood, MA 02062: Artech House London, 2019, págs. 137-138. ISBN: 978-1-63081-662-9.

[9] Wikipedia. *Teorema de muestreo de Nyquist-Shannon*. URL: https://es.wikipedia.org/wiki/Teorema_de_muestreo_de_Nyquist-Shannon.

[10] Wikipedia. *Amplitude Modulation*. URL: https://en.wikipedia.org/wiki/Doppler_effect.

[11] DVB. *ETSI EN 300 744. Digital Video Broadcasting (DVB). Framing structure, channel coding and modulation for digital terrestrial television*. 650 Route des Lucioles, France: DVB, 2015, pág. 33. ISBN: 348-6-235620-001-7.

[12] Manuel Sierra Pérez. *Electrónica de comunicaciones*. Pearson Education, 2003, págs. 236-237. ISBN: 978-8-42053-674-3.

[13] Item. *Controlador PI*. URL: https://glossar.item24.com/es/indice-de-glosario/articulo/item//controlador-pi-2.html.

[14] Wikipedia. *QAM*. URL: https://es.wikipedia.org/wiki/Modulaci%C3%B3n_de_amplitud_en_cuadratura.

[15] Matlab. *CFAR Matlab*. URL: https://es.mathworks.com/help/phased/ug/constant-false-alarm-rate-cfar-detection.html.

[16] Mateusz Malanowski. *Signal Processing for Passive Bistatic Radar*. 685 Canton Street, Norwood, MA 02062: Artech House London, 2019, pág. 203. ISBN: 978-1-63081-662-9.

[17] Mateusz Malanowski. *Signal Processing for Passive Bistatic Radar*. 685 Canton Street, Norwood, MA 02062: Artech House London, 2019, págs. 207-219. ISBN: 978-1-63081-662-9.

[18] Matlab. *Constant False Alarm Rate (CFAR) Detection*. URL: https://www.mathworks.com/help/phased/ug/constant-false-alarm-rate-cfar-detection.html.

[19] Mateusz Malanowski. *Signal Processing for Passive Bistatic Radar*. 685 Canton Street, Norwood, MA 02062: Artech House London, 2019, pág. 221. ISBN: 978-1-63081-662-9.

[20] Mateusz Malanowski. *Signal Processing for Passive Bistatic Radar*. 685 Canton Street, Norwood, MA 02062: Artech House London, 2019, págs. 225-226. ISBN: 978-1-63081-662-9.

[21] Ester M. H.-P. Kriegel J. Sander X. Xiaowei. *A density-based algorithm for discovering clusters in large spatial databases with noise*. EEUU: Proceedings of the Second International Conference on Knowledge Discovery in Databases y Data Mining, 1996.

[22] RANJA SARKAR. *DBSCAN vs K-means*. URL: https://www.kaggle.com/code/ranja7/clustering-with-dbscan-compare-with-gmm.

[23] Mateusz Malanowski. *Signal Processing for Passive Bistatic Radar*. 685 Canton Street, Norwood, MA 02062: Artech House London, 2019, págs. 231-235. ISBN: 978-1-63081-662-9.

[24] Matlab. *Kalman*. URL: `https://es.mathworks.com/help/radar/ref/trackingkf.html`.

[25] Mateusz Malanowski. *Signal Processing for Passive Bistatic Radar*. 685 Canton Street, Norwood, MA 02062: Artech House London, 2019, págs. 236-237. ISBN: 978-1-63081-662-9.