eman ta zabal zazu

Universidad del País Vasco
Euskal Herriko Unibertsitatea

INFORMATIKA FAKULTATEA
FACULTAD DE INFORMÁTICA

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Ingeniería de Software

Sistema de mapeo de actividad humana en el mar

Mikel Oscoz Rivas

Dirección

Alfredo Goñi Sarriguren (EHU)

Jose Antonio Fernandes Salvador (AZTI)

10 de junio de 2025

Resumen

Este documento corresponde a la memoria del Trabajo de Fin de Grado titulado “Sistema de mapeo de actividad humana en el mar”, realizado en colaboración con el centro científico y tecnológico AZTI. El proyecto se enmarca dentro del ámbito de la teledetección aplicada a la detección de actividades humanas en el mar, como por ejemplo la instalación de infraestructuras offshore.

El trabajo se ha centrado en el análisis y comparación de imágenes satelitales SAR y ópticas para la detección de estructuras en el mar, con especial atención a las granjas eólicas marinas. Se ha desarrollado un sistema que descarga, procesa y detecta datos satelitales. Asimismo, se han evaluado diferentes tipos de datos y metodologías para identificar objetos flotantes, teniendo en cuenta las limitaciones y ventajas de cada tipo de sensor. El resultado es una propuesta metodológica reproducible y adaptable para el mapeo de estas infraestructuras desde el espacio.

Objetivos de Desarrollo Sostenible

Este Trabajo de Fin de Grado tiene una relación muy estrecha con el Objetivo de Desarrollo Sostenible número 14: Vida submarina, cuyo objetivo es conservar y utilizar de forma sostenible los océanos, los mares y los recursos marinos para un desarrollo sostenible.

Utilizando técnicas de teledetección a partir de imágenes satelitales, este trabajo se centra en el desarrollo de nuevas herramientas para detectar actividades humanas en el medio marino, como granjas eólicas. Poder monitorizar periódicamente estas estructuras permitirá un mayor control de las actividades que se realizan en el océano, algo clave para evitar impactos no deseados en los ecosistemas marinos, usar de forma más sostenible los recursos y avanzar hacia un desarrollo más respetuoso de los océanos.

Pero si a eso le sumamos que con este tipo de información se puede conocer de primera mano cuál es realmente el impacto de todas estas infraestructuras, es posible que autoridades y organizaciones medioambientales puedan tomar decisiones y actuar para minimizar el impacto que tiene en la biodiversidad marina. El uso de fuentes de datos abiertos de satélite, y en definitiva, de tecnologías accesibles y reutilizables, será crucial para que la observación de la Tierra sea accesible a todo el mundo, y por lo tanto, sea inclusiva.

En definitiva, este trabajo ofrece una base tecnológica que apoyará en la toma de decisiones hacia una gestión más sostenible y transparente en el entorno marino.

Índice de contenidos

Índice de contenidos v

Índice de figuras vii

Índice de tablas viii

1 Introducción 1
- 1.1. Problema . . . 1
- 1.2. Estado del arte . . . 2
- 1.3. Introducción al trabajo . . . 3

2 Planificación 5
- 2.1. Alcance . . . 5
  - 2.1.1. Objetivos . . . 5
- 2.2. Esquema de descomposición de tareas (EDT) . . . 6
  - 2.2.1. Tareas generales . . . 6
  - 2.2.2. Tareas específicas . . . 6
- 2.3. Planificación prevista . . . 9
  - 2.3.1. Diagrama de Gantt . . . 9
  - 2.3.2. Hitos . . . 10
- 2.4. Gestión del tiempo . . . 10
- 2.5. Riesgos . . . 11
- 2.6. Precedentes . . . 11
- 2.7. Tecnologías utilizadas . . . 12

3 Análisis de requisitos 15
- 3.1. Análisis de dominio . . . 15
  - 3.1.1. Descripción del dominio . . . 15
  - 3.1.2. Problemas y necesidades . . . 15
- 3.2. Análisis de funcionalidades . . . 16
  - 3.2.1. Funcionalidades principales . . . 16
  - 3.2.2. Funcionalidades secundarias . . . 17
  - 3.2.3. Requisitos no funcionales . . . 17
- 3.3. Requisitos técnicos . . . 17

3.3.1. Requisitos de Hardware . . . 18

4 Diseño **19**
4.1. Arquitectura del sistema . . . 19
4.1.1. Diagrama de arquitectura . . . 19
4.1.2. Componentes principales . . . 19
4.2. Diseño UML . . . 21
4.3. Arquitectura de los datos . . . 21
4.3.1. Modelo de datos . . . 22

5 Implementación **25**
5.1. Módulo de descarga . . . 25
5.2. Módulo de preprocesado . . . 29
5.2.1. Temporal Merging . . . 29
5.2.2. Eliminado de tierra . . . 31
5.2.3. Localización . . . 32
5.3. Módulo de detección . . . 33

6 Pruebas **37**
6.1. Pruebas realizadas . . . 37
6.2. Automatización . . . 39
6.3. Validar resultados . . . 39

7 Seguimiento **41**
7.1. Revisión del Alcance . . . 41
7.2. Gestión del tiempo . . . 42
7.2.1. Desviaciones de fechas . . . 42
7.2.2. Desviaciones en las dedicaciones . . . 42
7.3. Gestión de riesgos . . . 42
7.4. Desviaciones de tareas . . . 44

8 Conclusiones **45**
8.1. Conclusiones generales . . . 45
8.2. Limitaciones de OpenEO . . . 46
8.3. Nuevas competencias . . . 47
8.4. Aportaciones novedosas respecto a trabajos previos . . . 48
8.5. Integración en equipo de AZTI . . . 48

Bibliografía **49**

Bibliografía **50**

Índice de figuras

1.1. Comparación entre imagen SAR y Óptica 2
1.2. Diagrama de flujo del proyecto 4

2.1. Esquema de descomposición de tareas (EDT) 7
2.2. Diagrama de Gantt 9
2.3. Integración entre tecnologías 13

3.1. Casos de uso identificados 16

4.1. Diagrama de la arquitectura del sistema 20
4.2. UML de la arquitectura del sistema 21
4.3. Diagrama del flujo de datos 23

5.1. Ejemplo del menú del sistema 26
5.2. Ejemplo del visualizador de Copernicus 27
5.3. Ejemplo de un datacube 27
5.4. Conflicto entre tiles 29
5.5. Proceso de fusión temporal 30
5.6. Máscara de la Agencia Medioambiental Europea 33
5.7. Máscara hecha a mano 34
5.8. Comparación de las dos máscaras en la misma imagen 35
5.9. Resultado de la localización de granjas eólicas 36

6.1. Prueba de la zona de estudio con imágenes ópticas 38

8.1. Diferencias entre SAR y Optical (Fuente: Cervest) 46

Índice de tablas

2.1. Hitos del proyecto 10
2.2. Dedicación estimada de cada tarea 10

7.1. Desviaciones en los hitos del proyecto 42
7.2. Dedicación real de cada tarea 43

CAPÍTULO 1

Introducción

1.1. Problema

El aumento de las actividades humanas en el mar, como la acuicultura, los parques eólicos y los barcos, genera una gran presión sobre los ecosistemas marinos. La acuicultura, aunque proporciona una fuente sostenible de comida, puede ocasionar graves problemas medioambientales. Por ejemplo, la contaminación del agua puede dañar la calidad del agua local o favorecer la transmisión de enfermedades entre especies, aunque hay especies que pueden mejorar la calidad del agua (ej. bivalvos). También pueden causar degradación del hábitat causada por la construcción de instalaciones de acuicultura o fomentar la aparición de arrecifes artificiales.

Los parques eólicos son una fuente de energía renovable, pero pueden alterar los hábitats marinos durante su construcción y explotación (Galparsoro et al., 2022) [6]. El ruido generado durante la construcción puede interferir con la comunicación y navegación de los mamíferos. Además, la presencia de turbinas puede alterar los patrones migratorios de algunas aves y especies marinas. El tráfico marítimo también tiene un gran impacto en los ecosistemas marinos a través de la contaminación acústica. Además, los barcos pueden causar derrames de petróleo, lo que lleva a una grave contaminación del agua y daños a largo plazo en la vida marina.

Detectar y mapear estas actividades es esencial para la gestión de los recursos marinos y la planificación espacial (Coccoli, Caroline, et al. 2018) [2]. En la planificación espacial, las actividades marítimas que tradicionalmente serían reguladas dentro de sectores independientes se gestionan en el contexto de todo el ecosistema para maximizar el desarrollo minimizando al mismo tiempo el impacto humano. Los datos precisos y actualizados son cruciales para tomar decisiones informadas a lo largo del tiempo y el espacio sobre la conservación del medio ambiente. Sin embargo, recopilar datos precisos a gran escala es una tarea difícil porque los informes de las autoridades regionales y nacionales a menudo no distinguen entre áreas planificadas y áreas de actividad real.

El mapeo requerido para la planificación espacial o el seguimiento de actividades puede abordarse con tecnologías de teledetección, como las observaciones satelitales. Los satélites

equipados con radar de apertura sintética (SAR) y sensores ópticos pueden proporcionar datos consistentes desde una escala local hasta una global. Estos datos pueden utilizarse para desarrollar mapas de actividades humanas en el mar, lo que permite una mejor supervisión y gestión de los ecosistemas marinos.

1.2. Estado del arte

En los últimos años, la teledetección ha resultado ser una herramienta poderosa para el monitoreo de las actividades humanas en el mar. La teledetección es la adquisición de información sobre la superficie terrestre sin contacto directo, utilizando satélites normalmente. Esta tecnología emplea varios tipos de sensores para capturar datos a través de diferentes longitudes de onda del espectro electromagnético. Existen dos categorías principales de sensores de teledetección: pasivos y activos. Los sensores pasivos, como los sensores ópticos e infrarrojos, detectan la radiación natural que es emitida o reflejada por los objetos en la superficie terrestre. Los sensores activos, como el radar de apertura sintética (SAR), emiten sus propias señales y miden la retrodispersión desde la superficie terrestre. Las principales diferencias entre los dos tipos de teledetección son que las imágenes ópticas se capturan a color, es decir, cada imagen tiene tres bandas (RGB), y que ofrecen una mayor resolución espacial. Además, los satélites ópticos suelen tener una mayor resolución temporal, es decir, vuelven a pasar por la misma zona con más frecuencia. En cambio, las imágenes SAR solamente tienen dos bandas, VV y VH, por eso se ven en blanco y negro, y su resolución espacial es menor.

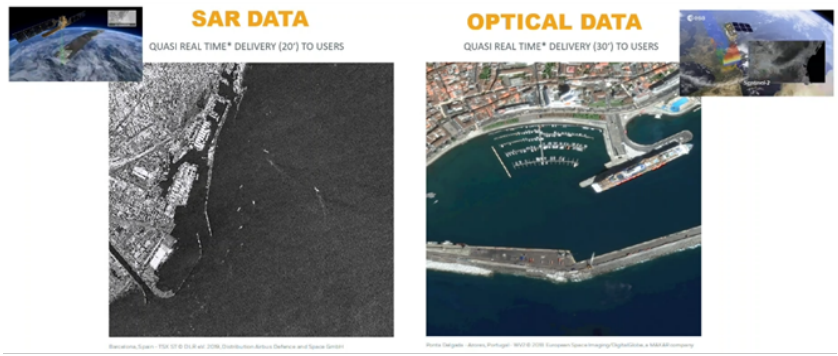

Figura 1.1: Comparación entre imagen SAR y Óptica

Los sensores SAR son particularmente valiosos por su capacidad de operar tanto de día como de noche y bajo cualquier condición meteorológica, lo que los hace ideales para aplicaciones como el monitoreo de actividades humanas en el mar. Estudios previos han utilizado datos SAR para detectar acuicultura en zonas costeras (Lekunberri et al., 2023) [14], utilizando su capacidad para capturar imágenes independientemente de las condiciones de luz o del clima. Otros estudios han conseguido utilizar las imágenes SAR para detectar derrames de aceite (Topouzelis, Konstantinos N., 2008) [12] demostrando así que estas imágenes no solo sirven para detectar objetos físicos flotantes.

Por otro lado, las imágenes satelitales ópticas han sido utilizadas para detectar barcos en

la superficie del mar (G. Yang, B et al., 2014) [5] (Corbane, Christina, et al., 2010) [3]. Estas imágenes son eficaces para capturar información visual detallada y han sido empleadas para validar los resultados de SAR y diferenciar entre tierra y mar.

Además de las aplicaciones mencionadas, investigaciones recientes han ampliado el uso de datos satelitales para detectar infraestructuras marinas como granjas eólicas marinas. Por ejemplo, se han desarrollado enfoques de aprendizaje automático espacial para detectar dinámicamente turbinas eólicas en el mar utilizando imágenes SAR (Xu et al., 2022) [15]. De forma complementaria, también se han empleado imágenes ópticas de Sentinel-2 para mapear granjas eólicas tanto terrestres como marinas en China (He et al., 2025) [11]. Asimismo, el análisis de series temporales de imágenes satelitales ha permitido evidenciar la proliferación de parques eólicos en el Mar del Norte y sus alrededores (Xu et al., 2020) [13]. Otros estudios como DeepOWT han desarrollado conjuntos de datos globales sobre turbinas eólicas marinas a partir de imágenes SAR y modelos de deep learning (Hoeser et al., 2022) [10], lo que demuestra el potencial de estas técnicas para la observación sistemática de infraestructuras marinas.

Aunque ambas tecnologías presentan ventajas claras, también tienen limitaciones. Las imágenes ópticas son dependientes de la iluminación y las condiciones atmosféricas, mientras que las imágenes SAR pueden presentar ruido y requieren mayor experiencia para su interpretación. Además, existen ya aplicaciones operativas que integran estas tecnologías, como el programa europeo Copernicus, que emplea imágenes SAR y ópticas para la vigilancia marítima, incluyendo la detección de barcos no identificados y el monitoreo de actividades ilegales. Aun así, persisten desafíos importantes, como la escasez de datos etiquetados para el entrenamiento de modelos, la confusión entre distintos tipos de objetos flotantes y la necesidad de mejorar la resolución temporal y espacial de las observaciones.

1.3. Introducción al trabajo

El objetivo de este trabajo es desarrollar un sistema de big data para extraer información sobre las actividades humanas en el mar a partir de observaciones satelitales, con el fin de crear un mapeo de la presión humana. Este sistema se basará en productos remotos disponibles gratuitamente en la plataforma europea Copernicus, utilizando datos ópticos. El trabajo se centrará en ampliar una prueba de concepto existente para la detección de acuicultura (Lekunberri et al., 2023) utilizando datos SAR, para incluir la detección de parques eólicos y barcos, y analizar el uso potencial de datos ópticos para mejorar la detección y la clasificación automática.

El primer paso del flujo de trabajo, representado en la Figura 1.2 es descargar los datos del satélite Sentinel-2 a nuestro espacio de trabajo de manera automatizada, en este caso los servidores de AZTI. Luego, los datos son preprocesados para eliminar toda la información innecesaria o ruido. Después se aplican unos filtros para detectar la actividad humana en las imágenes obtenidas. Por último, se utiliza un modelo preentrenado para clasificar lo que aparezca en las imágenes, que se puede tratar de actividad humana o ruido que no se haya eliminado en el procesado.

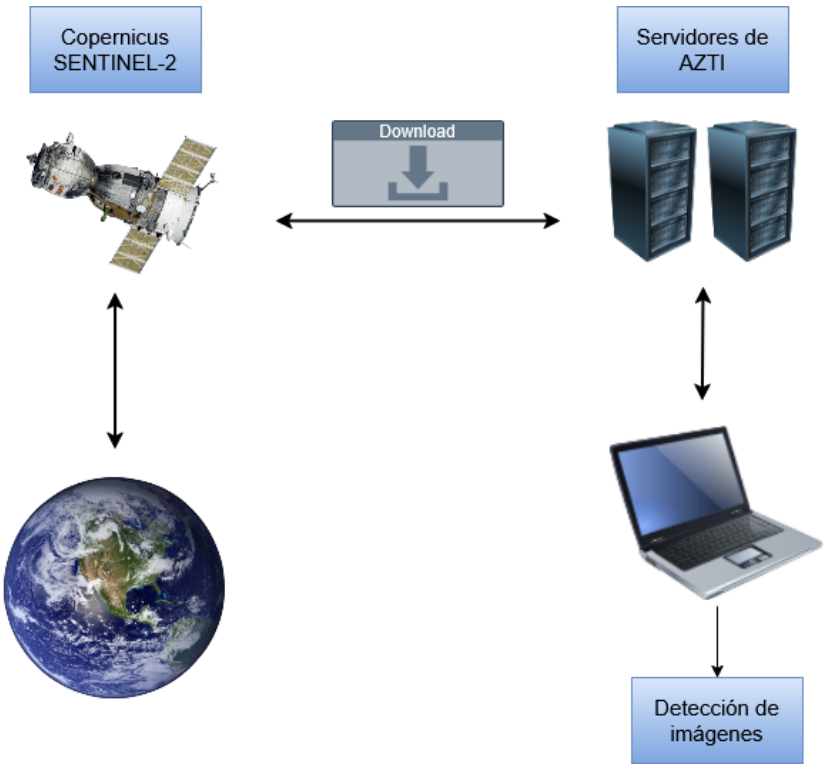

Figura 1.2: Diagrama de flujo del proyecto

CAPÍTULO 2

Planificación

En este apartado se muestra la planificación del Trabajo de Fin de Grado, definiendo el alcance, las tareas y la planificación inicial. También se analizarán los riesgos y, por último, se verán los precursores y las tecnologías utilizadas junto con su respectiva justificación.

2.1. Alcance

El objetivo principal de este proyecto es detectar y mapear actividad humana en el mar. Esto puede ser muy útil para biólogos marinos, pescadores y otros profesionales que estudian o trabajan en el mar. Además, puede resultar interesante para cualquier persona que trabaje con datos satelitales para aprender a trabajar con imágenes ópticas y los principios de la teledetección.

2.1.1. Objetivos

Aunque el objetivo principal es detectar la actividad humana en el mar, el Trabajo de Fin de Grado tiene algunos objetivos complementarios:

- **Aprender a trabajar con datos satelitales:** Los datos satelitales tienen una gran utilidad hoy en día, por lo que es importante aprender a trabajar con ellos, abarcando desde su descarga y preprocesamiento hasta su análisis y explotación. Otro objetivo del proyecto es saber cómo utilizar las herramientas del satélite *SENTINEL-2* de la plataforma *Copernicus* y entender su forma de captar las imágenes.
- **Crear un sistema de software para mapear actividad humana:** Para hacer un mapeo de actividad humana en el mar, es necesario crear un *software* que integre los distintos componentes de descarga de datos, preprocesado y detección. Cada componente es imprescindible para llevar a cabo el proyecto y debe comunicarse con los otros para utilizar datos compartidos y llevar a cabo el objetivo final.

- **Aplicar los conocimientos adquiridos en la universidad:** Un objetivo más personal es aplicar los conocimientos adquiridos durante la carrera en un caso real. En este proyecto se aplicarán los conceptos más importantes de la rama de *software* y de las asignaturas opcionales de cuarto relacionadas con Visión por Computador y Aprendizaje Automático.

2.2. Esquema de descomposición de tareas (EDT)

Las tareas generales para el TFG están detalladas en la figura 2.1, que se corresponde con el diagrama EDT visto en la asignatura *Gestión de Proyectos*. Cada tarea se descompone en subtareas que se explicarán en un apartado posterior. Para simplificar, se ha dividido en dos ramas con dos tareas generales cada una, dando un total de 4 tareas generales. Dentro de cada una de estas, están las tareas específicas.

2.2.1. Tareas generales

- **Estudio (E):** Esta tarea incluye la recopilación de información previa al inicio del trabajo. Esto puede abarcar la revisión de literatura, análisis de datos existentes y consulta con expertos en el campo.
- **Implementación (I):** Esta tarea abarca el desarrollo e implementación de la parte más técnica del trabajo. Incluye la programación, pruebas de software y la integración de diferentes componentes del sistema.
- **Documentación (D):** Esta tarea incluye la escritura de la memoria y la documentación del proyecto. Se detallarán los métodos utilizados, los resultados obtenidos y las conclusiones del trabajo.
- **Seguimiento (S):** Aquí se incluyen todas las actividades relacionadas con el control y seguimiento del proyecto. Esto puede incluir reuniones de seguimiento, evaluación de hitos y ajustes en la planificación según sea necesario.

2.2.2. Tareas específicas

Como se puede apreciar en la figura 2.1, el proyecto se ha dividido en dos ramas: desarrollo y gestión. En este apartado, se detallarán las tareas específicas de cada rama.

2.2.2.1. Desarrollo

- **Estudio (E):**
  - Revisión de Literatura (E.R): Antes de comenzar con el proyecto, es imprescindible analizar los trabajos anteriores que se han realizado en el mismo campo. Conocer las tecnologías que han utilizado o qué datos han elegido y por qué. De este modo, podemos definir el punto desde el que vamos a partir. Para este proyecto se ha

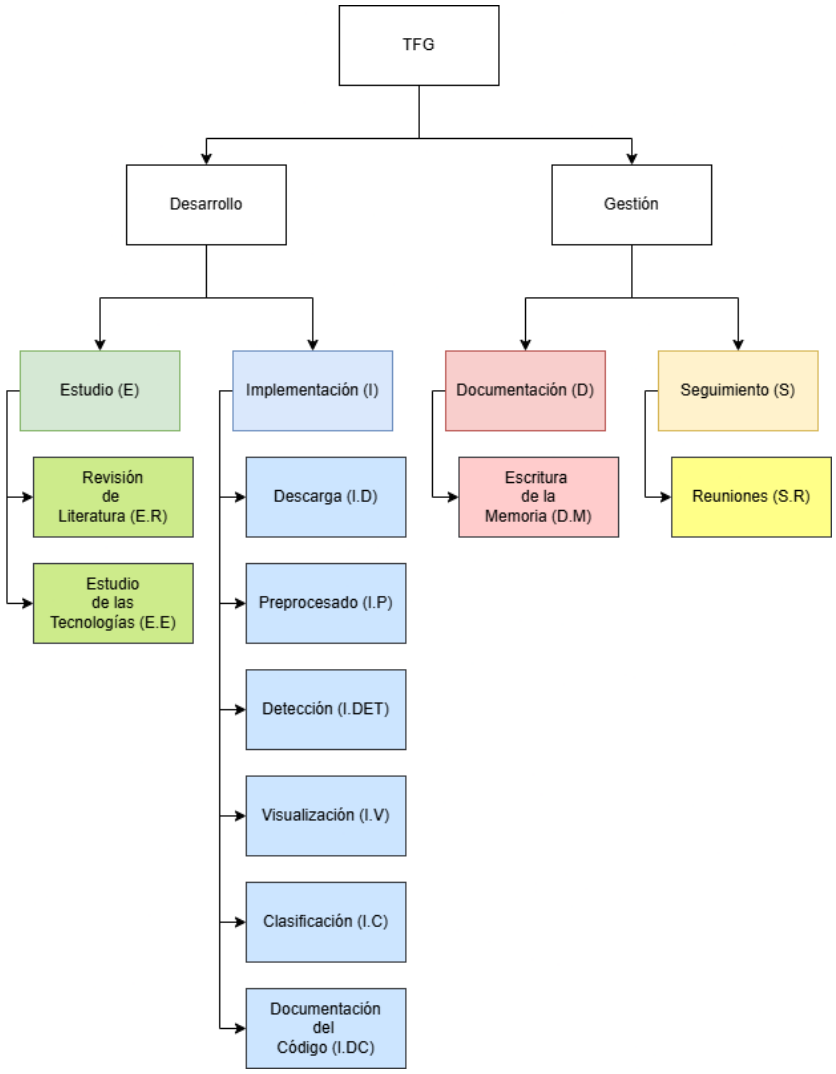

Figura 2.1: Esquema de descomposición de tareas (EDT)

utilizado como punto de partida el trabajo publicado en el artículo científico de (Lekunberri et al., 2023).

- Estudio de las Tecnologías (E.E): Este proyecto busca trabajar con datos satelitales, los cuales pueden ser complicados de utilizar si no se ha hecho con anterioridad. Por eso, antes de empezar es necesario hacer un estudio exhaustivo sobre las tecnologías y plataformas de datos satelitales. En este caso, hay que conocer la plataforma *Copernicus Browser*, para la visualización y búsqueda de datos y la librería de *Python OpenEO*, para su descarga.

- **Implementación (I):**

  - Descarga (I.D): La primera tarea en el apartado de implementación, es lograr la

descarga automatizada de datos mediante *scripts*. Estos datos, se almacenan en un sistema de directorios para ser utilizados más adelante por otros componentes del sistema.

- Preprocesado (I.P): Como los datos que se utilizan son imágenes, es preferible aplicar un preprocesado. Esto es una tarea básica en la Visión por Computador, porque las imágenes casi siempre tienen imperfecciones o ruido que puede dificultar trabajar con ellas.
- Detección (I.DET): Esta tarea, es el núcleo del proyecto, ya que aquí es donde se tratan las imágenes previamente procesadas y se aplica la detección de actividad humana, que es el objetivo principal del sistema.
- Visualización (I.V): Tras la detección, el sistema ha de proporcionar un *Output* visual, es decir, el mapeo. De este modo se pueden analizar los resultados de manera visual.
- Clasificación (I.C): Esta tarea se planteó como opcional en los parámetros del proyecto al inicio del TFG. Consiste en clasificar los distintos tipos de actividad humana previamente detectada mediante el uso de redes neuronales.
- Documentación del Código (I.DC): En la *Ingeniería del Software*, es importante mantener un código limpio y fácil de entender. Para lograr esto, otra tarea que se ha añadido es la de documentar el código, esto podría resultar muy útil por si alguien en algún futuro retomara el proyecto.

2.2.2.2. Gestión

Ahora, se especificarán las tareas de la rama de gestión. Como se puede apreciar, en esta rama hay menos tareas, esto es porque cada una de ellas es más significativa y van a ser llevadas a cabo durante todo el ciclo de vida del proyecto.

- **Documentación (D):**
  - Escritura de la Memoria (D.M): Esta tarea consiste en la escritura de la memoria del TFG. Esta es una tarea constante durante todo el proyecto y que va en paralelo al desarrollo. Esto es muy importante, ya que si se deja para el final, en la documentación no quedarán reflejados los detalles más significativos.
- **Seguimiento (S):**
  - Reuniones (S.R): Esta tarea sirve para llevar un seguimiento cercano y constante del proyecto. En el ciclo de vida de un proyecto, se han de realizar numerosas reuniones de control de calidad. No solo para el apartado técnico, sino que también para el apartado de escritura. En este caso las reuniones serán con el director del proyecto y con el responsable en la empresa, para discutir sobre el rumbo del proyecto.

2.3. Planificación prevista

En esta sección se detallará la planificación prevista del proyecto, es decir, la planificación realizada durante las primeras semanas del ciclo de vida del proyecto. Esta es una primera versión, y lo normal es que con el avance del proyecto se vaya ajustando si algunas tareas se cumplen dentro de periodos no establecidos. Estos desajustes serán detallados en un apartado posterior.

2.3.1. Diagrama de Gantt

	Tarea	Duración (Semanas)	Inicio	Fin
1	Proyecto completo	18	S 8	S 25
2	Estudio (E)	4	S 8	S 11
3	Revisión de Literatura (E.R)	4	S 8	S 11
4	Estudio de las Tecnologías (E.E)	3	S 8	S 10
5	Implementación (I)	13	S 10	S 23
6	Descarga (I.D)	4	S 10	S 13
7	Preprocesado (I.P)	5	S 12	S 16
8	Detección (I.DET)	5	S 15	S 19
9	Visualización (I.V)	2	S 19	S 20
10	Clasificación (I.C)	4	S 19	S 22
11	Documentación del Código (I.DC)	2	S 22	S 23
12	Documentación (D)	18	S 8	S 25
13	Escritura de la Memoria (D.M)	18	S 8	S 25
14	Seguimiento (S)	18	S 8	S 25
15	Reuniones (S.R)	18	S 8	S 25

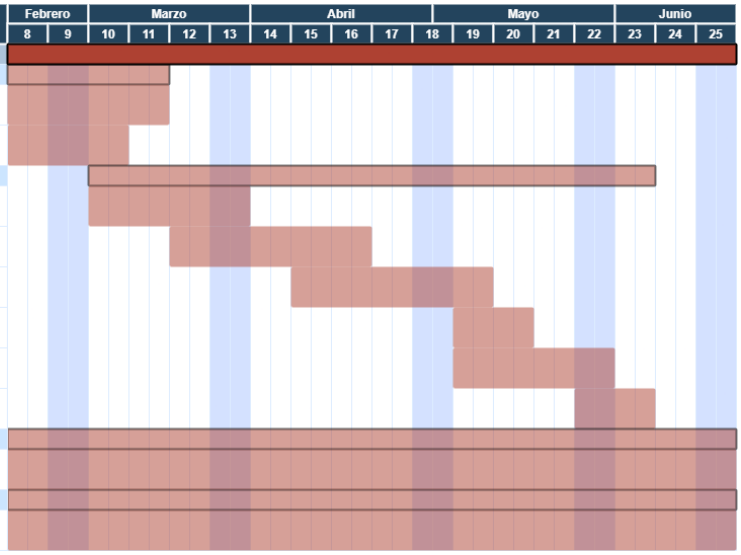

Figura 2.2: Diagrama de Gantt

En esta subsección se presenta el diagrama de Gantt (Figura 2.2), donde se muestra el inicio y el fin de cada tarea con el fin de visualizar el tiempo que se le va a distribuir a cada deber. Dado que la fecha de inicio del proyecto es el 17 de febrero y el fin el 16 de junio, son un total de 18 semanas, que van desde la 8ª semana del año hasta la 25ª. El eje x del diagrama son las semanas, y el eje y las tareas. Y el área roja indica el plazo de trabajo de dicha tarea.

A primera vista, se puede apreciar que las tareas de estudio se realizarán en las primeras semanas porque son necesarias para empezar con el proyecto. Dentro del apartado de implementación, se prevé que las tareas de descarga, preprocesado y detección sean las más largas, ya que se espera que surjan imprevistos durante el desarrollo. Por último, las tareas de documentación y seguimiento van a ser llevadas a cabo durante todo el ciclo de vida del proyecto.

2.3.2. Hitos

En la tabla 2.1 se encuentran los hitos del proyecto, en vez de asignar un hito a cada tarea, se han elegido las actividades y tareas más importantes y a cada una se le ha asignado una fecha de finalización. Esta fecha indica que, si para esa fecha la tarea no se ha completado, se han de hacer modificaciones en la planificación.

Hito	Fecha
Inicio del proyecto	17-02-2025
Descarga (I.D)	02-04-2025
Preprocesado (I.P)	25-04-2025
Detección (I.DET)	16-05-2025
Fin Implementación	06-06-2025
Fin memoria	08-06-2025

Tabla 2.1: Hitos del proyecto

2.4. Gestión del tiempo

En esta sección se muestra la dedicación estimada para cada tarea, que se puede ver en la tabla 2.2. El total aproximado es de 310 horas, que suena factible teniendo en cuenta que un TFG tiene una duración estimada de 300 horas.

TAREA	DEDICACIÓN ESTIMADA
Estudio (E)	
Revisión de Literatura (E.R)	15h
Estudio de las tecnologias (E.E)	10h
Subtotal	25h
Implementación (I)	
Descarga (I.D)	35h
Preprocesado (I..P)	40h
Detección (I.DET)	50h
Visualización (I.V)	10h
Clasificación (I.C)	40h
Documentación del código (I.DC)	10h
Subtotal	185h
Documentación (D)	
Escritura de la Memoria (D.M)	80h
Subtotal	80h
Seguimiento (S)	
Reuniones (S.R)	20h
Subtotal	20h
Total	310h

Tabla 2.2: Dedicación estimada de cada tarea

2.5. Riesgos

Ahora se detallarán y gestionarán los riesgos, este es un aspecto vital para el desarrollo de un proyecto ya que un riesgo inesperado puede retrasar la entrega si no se toma acción inmediatamente.

- **Riesgo 1 - Estudios:** Este proyecto va a ser realizado durante el segundo cuatrimestre del curso 24-25, dado que al mismo tiempo se estarán cursando tres asignaturas, se va a tener que compaginar el proyecto con los estudios. Lo que puede ocasionar grandes desviaciones en la planificación como por ejemplo en las semanas de horario agrupado. La prevención de este riesgo es hacer la planificación teniendo en cuenta la carga de trabajo de las asignaturas que se estarán cursando y ver si tienen examen en la semana de horario agrupado.
- **Riesgo 2 - Tecnologías desconocidas y cambiantes:** Las tecnologías que se van a utilizar en este proyecto son totalmente desconocidas, por lo que hacer una planificación sin saber si su adaptación va a ser fácil o no es un riesgo a tener en cuenta. Además se depende de algunas tecnologías que pueden tener cambios durante el desarrollo del proyecto. Para prevenir este riesgo se asignará un mayor tiempo al estudio inicial.
- **Riesgo 3 - Planificación inadecuada:** Uno de los riesgos que se ha de tener en cuenta en cualquier proyecto de cualquier ámbito es el riesgo de seguir una planificación inadecuada. Como se menciona en el punto anterior, la planificación tiene el riesgo de ser inadecuada debido al desconocimiento desde el que se ha hecho. Es decir, al no haber trabajado nunca en un proyecto similar, las horas y periodos de trabajo asignados a cada tarea pueden ser erróneos. La prevención en este caso es consultar con el responsable dentro de la empresa, que tiene un gran conocimiento en el sector, para que confirme si la planificación es realista o no.

2.6. Precedentes

Como se ha mencionado anteriormente, este trabajo parte del realizado por Xabier Lekunberri, el cual se puede ver en su respectivo artículo científico (Lekunberri et al., 2023) escrito en 2023. En este artículo, Xabier explica cómo hizo un *Software* de detección de acuicultura con imágenes *SAR*.

Las imágenes *SAR* a diferencia de las ópticas, son captadas por radares que emiten sus propias microondas para captar imágenes. Y por este motivo, pueden sacar imágenes tanto de día como de noche sin verse afectados por las condiciones meteorológicas como las nubes. Por otro lado, tienen el inconveniente de que las imágenes resultantes son en blanco y negro y que por lo tanto son menos aptas para la visualización.

Al tener este trabajo tan similar como precedente, algunas de las técnicas se reutilizarán para este proyecto. Y como los problemas que se encontrarán a lo largo del proyecto serán parecidos, también se consultarán algunos de los métodos empleados para solucionarlos. Sin embargo, el proyecto final será un producto completamente nuevo e independiente.

2.7. Tecnologías utilizadas

En esta sección se detallarán las tecnologías, programas, *librerías* e interfaces de programación de aplicaciones (API) que se han utilizado para el proyecto y se justificará su elección. En la figura 2.3 se muestra cómo se relacionan las tecnologías.

- **Python:** Python es un lenguaje de programación interpretado de alto nivel conocido por su simplicidad y legibilidad. Su extensa biblioteca estándar y su comunidad activa lo convierten en una opción versátil para una amplia gama de aplicaciones, desde el desarrollo web hasta el análisis de datos y el aprendizaje automático. En este proyecto, Python ha sido utilizado para la creación de scripts, procesamiento de datos y el desarrollo del proyecto de mapeo humano que coordinaba todas las diferentes partes del mismo.
- **Visual Studio Code:** Visual Studio Code es un editor de código fuente desarrollado por Microsoft. Es conocido por su flexibilidad, soporte para múltiples lenguajes de programación y una amplia gama de extensiones que facilitan el desarrollo de software. En este proyecto, Visual Studio Code ha sido utilizado para múltiples cosas, al principio se crearon algunos Jupyter Notebooks con el fin de hacer algunos experimentos con las librerías y los datos. Más adelante, se desarrolló el proyecto en un sistema de archivos con Visual Studio.
- **Copernicus Browser:** Copernicus Browser es una plataforma en línea que proporciona acceso a datos de observación de la Tierra recopilados por el programa Copernicus de la Unión Europea. Esta herramienta es muy útil para obtener imágenes satelitales y otros datos geoespaciales que son cruciales para el análisis y la toma de decisiones en proyectos de gestión ambiental y del territorio. Esta herramienta fue vital para visualizar el área de estudio antes de descargar los datos. El navegador en sí, tiene muchas funciones útiles, como la posibilidad de dibujar un polígono alrededor del área de interés o poder elegir el satélite, el rango de tiempo y la cobertura de nubes para una imagen más precisa.
- **OpenEO:** OpenEO es una API y un framework de código abierto que permite procesar y analizar datos de observación de la Tierra de varios proveedores de back-end, incluido el programa Copernicus. Proporciona una serie de funciones para realizar consultas y descargar datos de los satélites. En este proyecto, OpenEO ha sido utilizado para descargar datos satelitales desde el *backend*.
- **GeoJson:** GeoJson es un formato de datos basado en JSON diseñado para representar datos geoespaciales. Es ampliamente utilizado para intercambiar información geográfica en aplicaciones web debido a su simplicidad y compatibilidad con numerosos sistemas y bibliotecas GIS. En este proyecto, GeoJson ha sido utilizado para convertir coordenadas físicas en objetos json, los cuales posteriormente fueron utilizados para descargar los datos de esos puntos exactos.
- **QGis:** QGis es un sistema de información geográfica (GIS) de código abierto que permite a los usuarios visualizar, editar y analizar datos geoespaciales. Su capacidad para manejar una variedad de formatos de datos y su extensa gama de herramientas de análisis lo

convierten en una herramienta esencial para proyectos que requieren la gestión de información geográfica. Este programa fue muy útil para visualizar todas las imágenes de satélite.

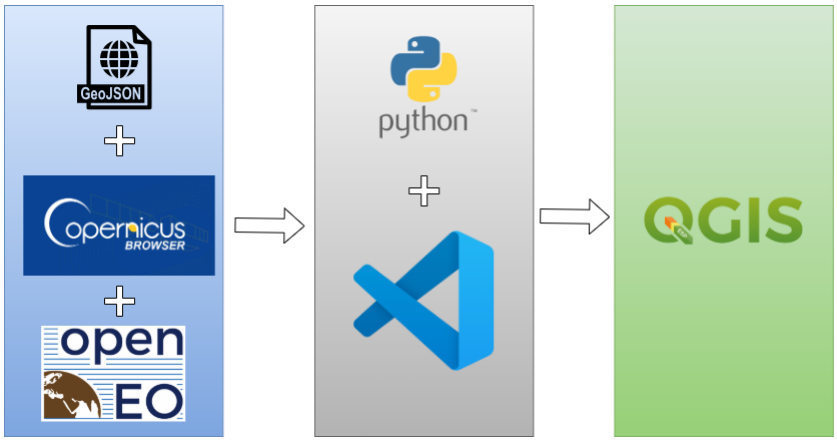

Figura 2.3: Integración entre tecnologías

CAPÍTULO 3

Análisis de requisitos

3.1. Análisis de dominio

El proyecto se centra en el desarrollo de un software para la detección de la actividad humana en el mar. El objetivo es proporcionar una herramienta que permita a los usuarios elegir un área de interés y un margen de fechas, y poder detectar y visualizar la actividad humana.

3.1.1. Descripción del dominio

Encontrar información sobre granjas eólicas es una tarea complicada, ya que la mayoría de los países no hacen pública esta información, y en el caso de hacerla es muy probable que se encuentre desactualizada. Para obtener información sobre el paradero de las granjas eólicas, muchas veces es necesario contactar con las autoridades del país al que pertenecen y ellos pueden elegir si proporcionar la información o no. Este sistema busca abordar estos desafíos proporcionando una solución accesible y fácil de usar que permita a los usuarios detectar granjas eólicas, u otros tipos de actividad humana en un futuro, de manera más eficiente.

3.1.2. Problemas y necesidades

Los principales problemas identificados en el dominio son:

- **Falta de información:** Las personas o instituciones no tienen ni pueden obtener el conocimiento exacto de la actividad humana en el mar, ya sea por la dificultad de su obtención o por ser un entorno muy cambiante.
- **Procesos manuales:** Si alguien quiere detectar la actividad humana en el mar analizando imágenes satelitales, va a tener que hacerlo manualmente, ya que en la actualidad no hay ningún software que permita automatizar esta tarea.

Las necesidades identificadas son:

- **Automatización de la detección de la actividad humana:** Los usuarios no pueden tener información actualizada sobre la actividad humana en el mar, ya que de querer hacerlo, tendrían que estar descargando manualmente nuevos datos constantemente y en muchas zonas no está disponible.

- **Mapeo de la actividad humana:** Los usuarios no pueden marcar la actividad humana en un mapa interactivo, destacando las partes importantes.

3.2. Análisis de funcionalidades

Para abordar los problemas y necesidades identificados, se han definido las siguientes funcionalidades o casos de uso (figura 3.1):

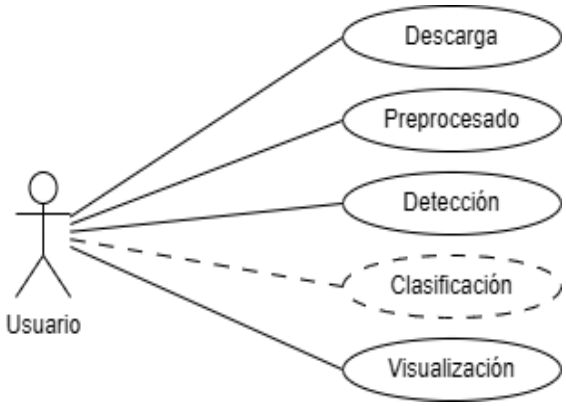

Figura 3.1: Casos de uso identificados

3.2.1. Funcionalidades principales

Todas las funcionalidades se ejecutan desde la interfaz de inicio, entre las funcionalidades principales y obligatorias para el sistema se encuentran las siguientes:

- **Descarga:** Esta funcionalidad descarga todos los datos que cumplan todos los parámetros definidos por el usuario. Primero, se le pregunta al usuario qué tipo de datos quiere descargar, **Optical** (Sentinel 2) o **SAR** (Sentinel 1). Después se pregunta de dónde se quieren descargar los datos, de la zona de estudio (Mar del norte) o si se desea definir una zona manualmente, en este caso se solicitarán las coordenadas para la definición de la zona. Seguidamente se pide al usuario que introduzca el rango de fechas del que se quieren descargar las fotos. Y tras comprobar que las fechas sean correctas, se termina pidiendo la nubosidad, un porcentaje del 1 al 100 (en el caso de que las imágenes sean ópticas). Con todos estos parámetros, se inicia el proceso de descarga automática, que tiene una duración variable en función de si se ha definido una zona manualmente o se ha elegido la zona de estudio y la amplitud temporal requerida. Tras descargar las imágenes, estas quedan guardadas en el directorio **data**.

- **Preprocesado:** Este caso de uso, se encarga de preprocesar todos los datos recopilados durante el proceso de descarga. Aquí no se le da ninguna opción al usuario, ya que todas las imágenes deben seguir el mismo proceso. Primero, se hace la fusión temporal de todas las imágenes de una misma área para eliminar todo el ruido y después se hace el eliminado de tierra. Las imágenes quedan guardadas en el directorio **no-land-data**. Si el usuario intenta ejecutar esta funcionalidad sin haber descargado previamente los datos, saltará un mensaje de error pidiendo que se descarguen los datos primero.

- **Detección:** Esta funcionalidad se encarga de hacer el proceso de detección de actividad humana, que es el objetivo principal del sistema. En este caso de uso tampoco se da ninguna opción a elegir, porque otra vez, todas las imágenes siguen el mismo proceso de detección. Al iniciar esta función, se recogen todos los datos del directorio *no-land-data* creado en la funcionalidad de preprocesado y se les aplica la función de detección. Tras aplicarla, los datos se guardan en el directorio **filtered-data**. En esta funcionalidad, como en la anterior, saltará un aviso si el usuario intenta iniciar el proceso de detección sin imágenes preprocesadas.

- **Visualización:** Este caso de uso sirve para visualizar los resultados obtenidos en un mapa interactivo. Cuando el usuario elige esta funcionalidad, todas las imágenes almacenadas en **filtered-data** se colocan en un mapa del mundo interactivo, donde todos los objetos detectados se muestran en un color rojo intenso. Como en las anteriores funcionalidades, si se intenta ejecutar sin los datos filtrados, la función no se llevará a cabo.

3.2.2. Funcionalidades secundarias

- **Clasificación:** Esta funcionalidad es la que se encarga de distinguir la actividad humana una vez haya sido detectada. Mediante una red neuronal entrenada, se clasificarían los elementos de cada foto entre: ruido o granja eólica. En un futuro, se podría extender para distinguir entre otros tipos de actividad humana como acuicultura o barcos.

3.2.3. Requisitos no funcionales

- **Usabilidad:** La interfaz de usuario será en la terminal ya que se trata de un software científico. Deberá ser intuitiva y fácil de usar.

- **Rendimiento:** El sistema debe ser capaz de manejar grandes volúmenes de datos sin afectar al rendimiento.

3.3. Requisitos técnicos

Para asegurar que la aplicación cumpla con los requisitos funcionales y no funcionales, se han definido los siguientes requisitos técnicos:

3.3.1. Requisitos de Hardware

- **Memoria RAM:** Un ordenador con al menos 8GB de RAM y 4 núcleos de CPU para manejar grandes datos. Dependiendo del tamaño de la solicitud, se puede necesitar más capacidad de RAM.
- **Almacenamiento:** Una memoria o disco duro con capacidad suficiente para almacenar datos pesados.

CAPÍTULO 4

Diseño

En este capítulo se detalla la arquitectura del sistema, es decir, cómo están organizados todos los componentes y cómo se comunican entre ellos. También se explicará de qué manera están guardados los datos y cómo es el sistema de directorios.

4.1. Arquitectura del sistema

En este apartado se ve cómo es la estructura del sistema primero en general y después entrando en detalle en cada apartado.

4.1.1. Diagrama de arquitectura

En la figura 4.1 se muestra la arquitectura general del sistema, el cual está compuesto por cinco módulos principales: Descarga, Preprocesado, Detección, Clasificación y Visualización. Todos ellos están coordinados desde un módulo principal denominado "*MAIN*".

Cada componente se ejecuta de forma secuencial y depende del resultado del anterior, estableciendo un flujo de datos en cascada. Por ejemplo, el módulo de Preprocesado solo se activa tras una descarga de datos, de la misma forma, el módulo de Detección no puede iniciarse si no existen datos preprocesados. Este diseño asegura una estructura modular pero interdependiente. Cabe destacar que el módulo de Clasificación es opcional. Su ejecución depende de si el usuario desea aplicar un modelo de clasificación sobre los resultados de la detección. La omisión de este módulo no afecta a la funcionalidad general del sistema.

4.1.2. Componentes principales

A continuación, se describen los cinco módulos del sistema, destacando su función dentro del flujo de ejecución, sus dependencias y sus particularidades.

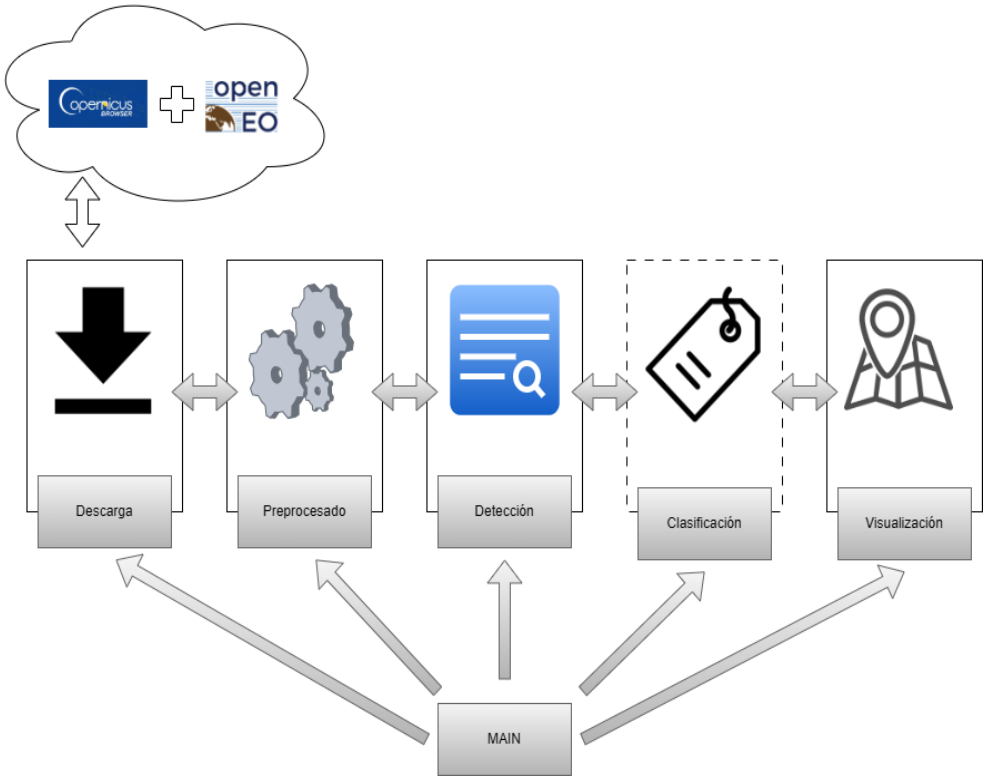

Figura 4.1: Diagrama de la arquitectura del sistema

- **Módulo de Descarga:** Este módulo interactúa con la API de OpenEO para recuperar imágenes satelitales del proyecto Copernicus según los parámetros definidos por el usuario. Internamente, realiza validaciones sobre fechas, ubicación geográfica y niveles de nubosidad. El sistema guarda los datos brutos en el directorio `data/`. Este componente fue diseñado para ser extensible, permitiendo incluir futuras fuentes de datos con mínimas modificaciones.
- **Módulo de Preprocesado:** Este módulo automatiza tareas necesarias para preparar los datos antes del análisis. Entre ellas se incluye la fusión temporal y el eliminado de tierra. Se utilizan librerías como `rasterio` y `numpy`. Sus salidas se almacenan en el directorio `no-land-data/`, listo para la detección.
- **Módulo de Detección:** Implementa el algoritmo de detección de actividad humana, a partir de las imágenes preprocesadas. Los resultados se almacenan en `filtered-data/`.
- **Módulo de Clasificación:** Este módulo, si se activa, aplica un modelo de clasificación sobre los objetos detectados para clasificar el tipo de actividad. Utiliza un clasificador entrenado previamente, cargado desde un archivo local.
- **Módulo de Visualización:** Este módulo permite representar los resultados en un mapa interactivo usando herramientas como `Folium`. Crea un archivo `html` que el usuario pue-

de abrir con cualquier navegador y explorar libremente. Este componente se comunica con el directorio filtered-data/ para acceder a los resultados ya procesados.

4.2. Diseño UML

En esta sección se muestra la arquitectura del sistema en **UML**. Entrando en detalles técnicos, cada componente es un archivo **Python** formado por una lista de funciones. Cada archivo tiene una función *main* que llama de manera ordenada a todas las funciones del mismo archivo. Y cada uno de los archivos es gestionado por el archivo *main*. Donde el usuario tiene una interfaz en la terminal que le permite elegir qué funcionalidad ejecutar. En la figura 4.2 se muestra este diagrama.

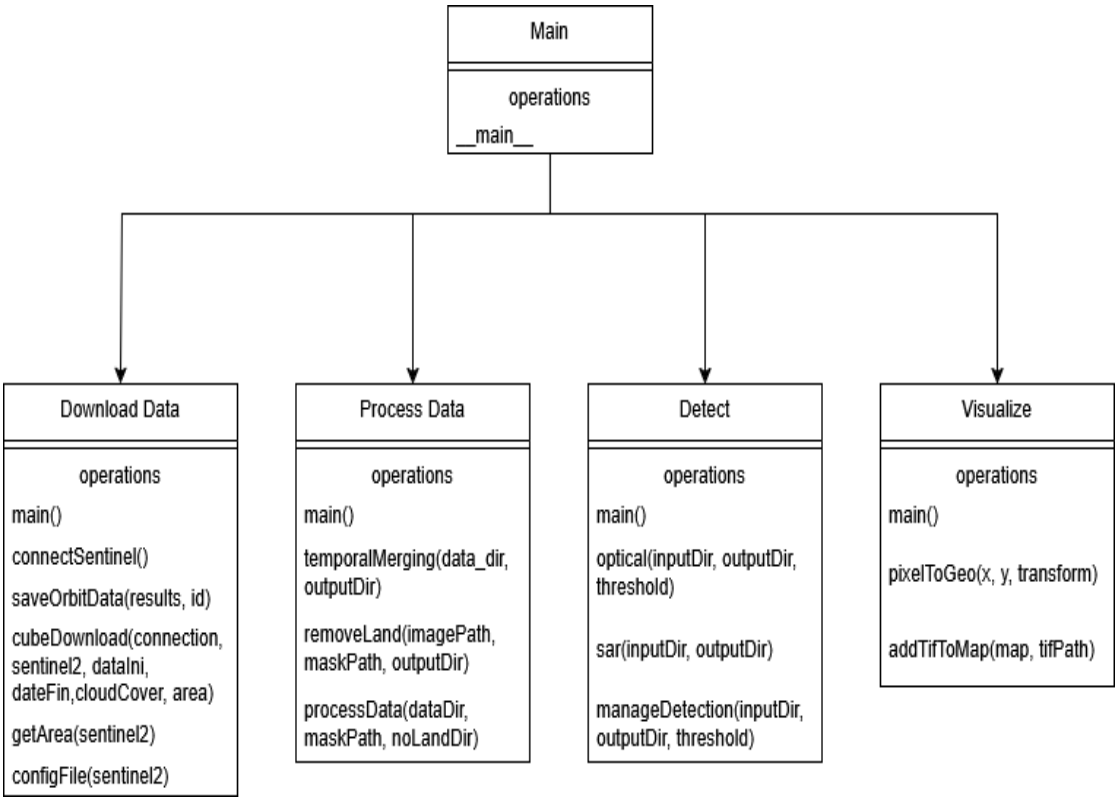

Figura 4.2: UML de la arquitectura del sistema

4.3. Arquitectura de los datos

En este apartado se detalla cómo están organizados y estructurados los datos que gestiona el sistema a lo largo de su ciclo de vida. Se describe tanto el sistema de directorios utilizado como las transformaciones que se realizan en cada etapa del flujo, desde la descarga inicial hasta la visualización de los resultados. Este análisis es fundamental para comprender cómo el sistema manipula grandes volúmenes de información satelital.

4.3.1. Modelo de datos

Los datos utilizados por el sistema provienen de imágenes satelitales descargadas en formato *.TIFF*. El modelo de datos del sistema se basa en una estructura de carpetas jerárquica que organiza los datos según su nivel de procesamiento:

- data/: Contiene los datos brutos descargados desde el proveedor (imágenes completas con metadatos).
- merged-data/: Almacena las imágenes tras aplicar el preprocesado (fusión temporal).
- no-land-data/: Almacena las imágenes tras aplicar el eliminado de tierra.
- filtered-data/: Incluye únicamente las imágenes que han pasado por el módulo de detección.

Cada imagen se identifica mediante un nombre de archivo que incluye la fecha de cuando la imagen fue tomada. En el caso de la carpeta de data/, cuando se haya elegido la opción de descargar el área del Mar del Norte, dentro de este directorio, habrá 11 subdirectorios numerados. Ya que esta zona se dividió en 11 áreas de menor tamaño. En caso de haber definido un área manualmente, solo habrá un subdirectorio. En la figura 4.3 se muestra un esquema conceptual que resume este apartado.

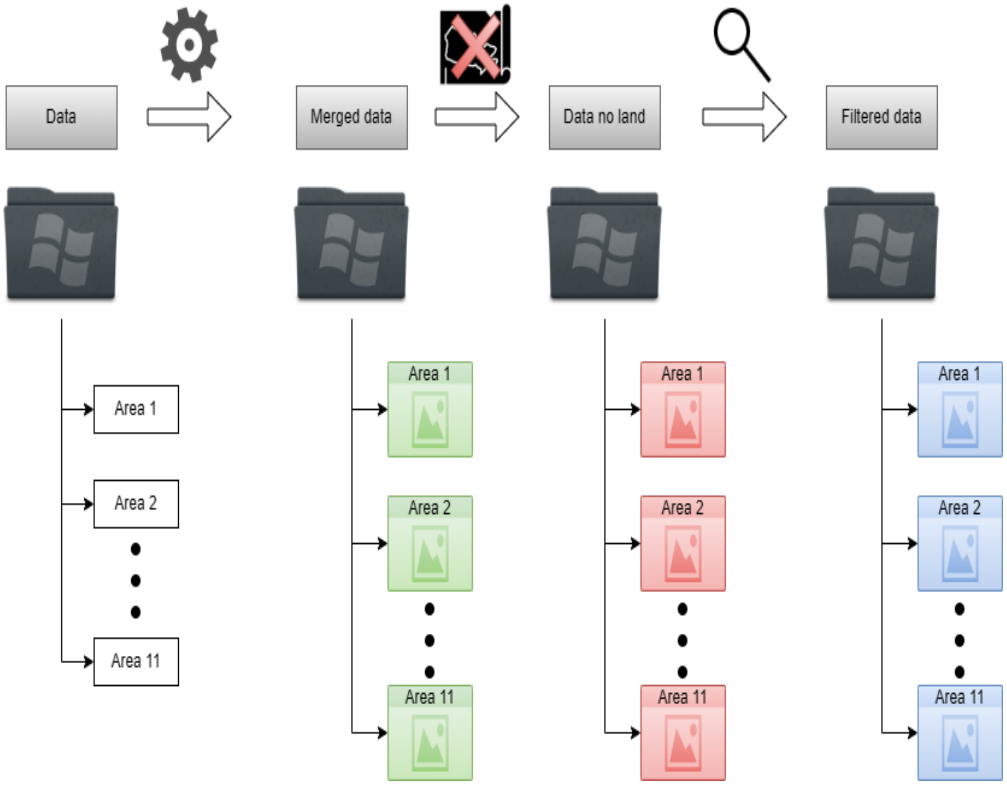

Figura 4.3: Diagrama del flujo de datos

CAPÍTULO 5

Implementación

En este capítulo, se desarrollará cómo se ha implementado cada parte del proyecto de manera cronológica, qué problemas han ido apareciendo y cómo se ha solucionado cada uno.

El sistema se ejecuta desde una interfaz en la terminal que presenta al usuario un menú con distintas opciones. Tras seleccionar una opción, se ejecuta la funcionalidad correspondiente y se vuelve al menú principal hasta que se elige la opción de salir. En la figura 5.1 se puede ver un ejemplo de la interfaz en la terminal.

5.1. Módulo de descarga

La primera tarea fue encontrar una manera de descargar datos de los satélites Copernicus Sentinel-2 mediante código. El navegador web de Copernicus tiene una opción que sirve para buscar imágenes en un área y período de tiempo designados. Después de especificar esos parámetros, muestra todas las imágenes tomadas bajo esas circunstancias y se pueden elegir las imágenes deseadas. En la figura 5.2 se puede ver un ejemplo. Esto es muy útil para hacer pruebas visuales y ver si hay datos disponibles.

La idea es aplicar esta misma lógica de la web de Copernicus en el código, partiendo de la base de que el usuario final introducirá un rango de fechas y un área para que el sistema encuentre todas las imágenes entre las fechas en esa área (al igual que se hace en la web). La mejor opción parece ser utilizar una función que descargue todas las imágenes del área en el rango de fechas. Aquí es donde entra en juego OpenEO [7]. Como se menciona en la planificación, esta API nos permite hacer consultas y descargar datos desde el backend.

Tras leer la documentación de OpenEO y establecer una conexión con el backend [9], la mejor solución encontrada fue definir primero un cubo de datos (*datacube*) utilizando la función **Load Collection** [4]. Un cubo de datos es simplemente un objeto que contiene los parámetros de uno o varios datos, en este caso: rango de fechas, coordenadas, bandas y cobertura de nubes. Las fechas y coordenadas significan exactamente lo que el nombre indica; las bandas son los

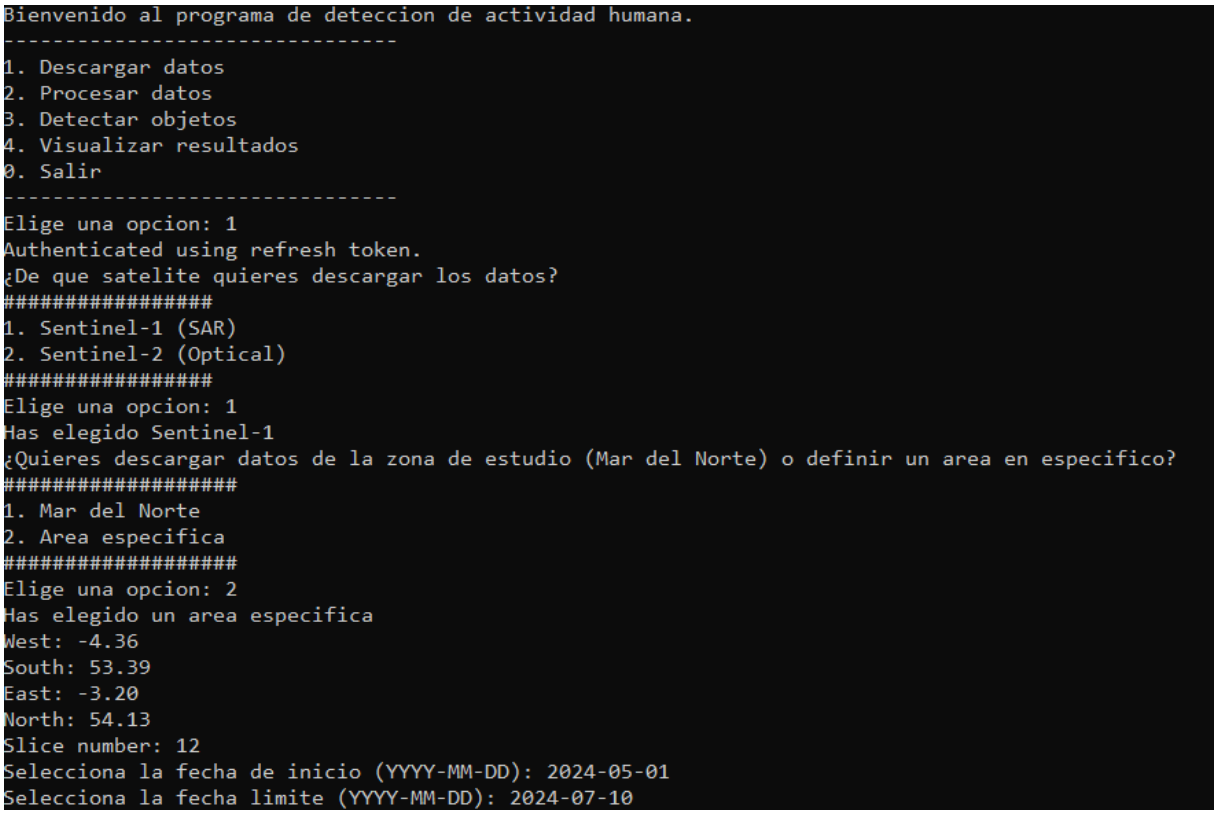

Figura 5.1: Ejemplo del menú del sistema

distintos canales que cada satélite tiene, y la cobertura de nubes es la nubosidad máxima que permitimos en la imagen. La zona de estudio principal del proyecto es el Mar del Norte, cubriendo la costa de los Países Bajos y Dinamarca (alrededor de 81.000 km²). Se ha elegido esta zona ya que abundan las granjas eólicas. En cuanto a las fechas, sería recomendable no utilizar rangos menores a 2022, ya que no se sabe cuándo se construyeron algunas granjas eólicas y, si utilizamos una fecha mínima muy lejana, podríamos estar descargando imágenes con información desactualizada. En las bandas especificamos qué canales de la fotografía queremos obtener. En el caso del satélite **Sentinel-2**, que toma fotos a color "*true color*", las bandas seleccionadas fueron las B04, B03 y B02, que son Rojo, Verde y Azul, respectivamente. En la cobertura de nubes se ha utilizado una nubosidad de entre el 10 % y el 15 %. Por otro lado, para el satélite **Sentinel-1**, se utilizaron las bandas VH y VV. Tras definir el cubo con estos datos, queda algo como lo que se ve en la figura 5.3.

Como se ha mencionado anteriormente, la función **Load Collection** devuelve un cubo de datos, donde se referencian todas las imágenes que cumplen las condiciones especificadas. Después de definir el cubo, existe la opción de descargarlo mediante OpenEO con la función **Download**. Aquí surgió el primer error, ya que el área seleccionada era demasiado grande. Una solución típica para este problema suele ser usar trabajos por lotes (*jobs*), pero el área era tan grande que esta opción tampoco era viable.

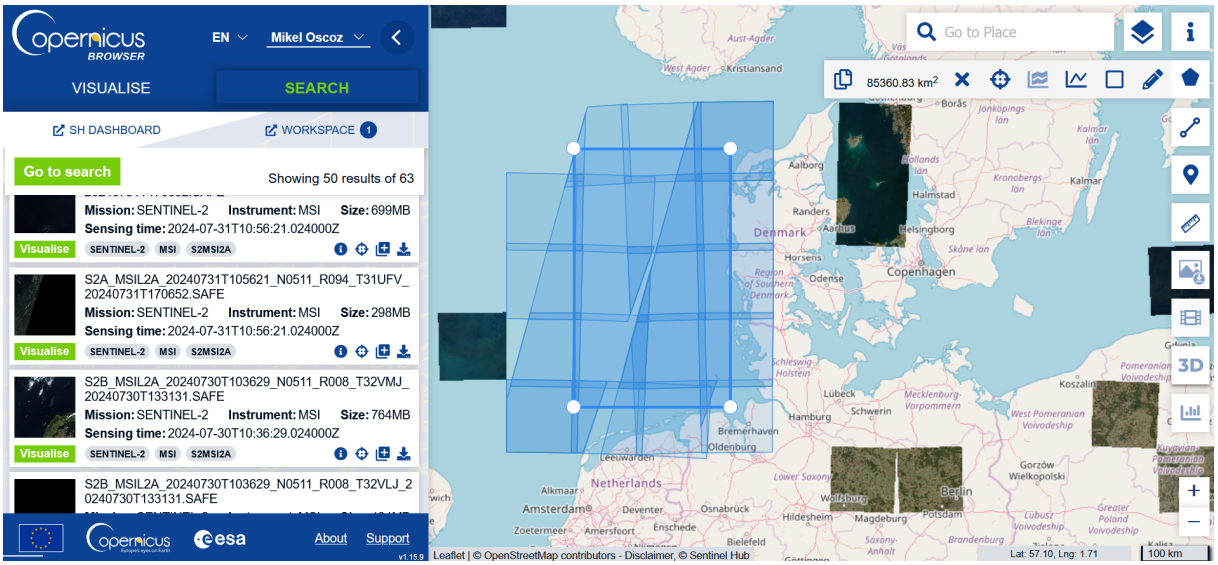

Figura 5.2: Ejemplo del visualizador de Copernicus

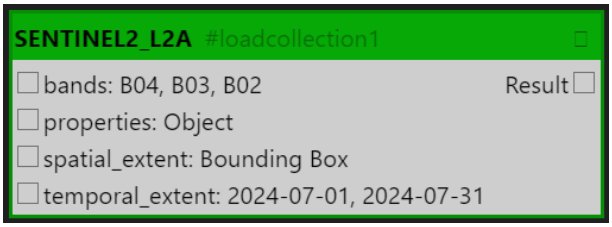

Figura 5.3: Ejemplo de un datacube

La solución para este problema fue dividir toda el área en subsecciones más pequeñas y luego hacer una función que descargue todas las imágenes de cada subsección individualmente. El área de cada subsección es de alrededor de 12.000 km², aproximadamente el mismo tamaño que las imágenes tomadas por el Sentinel-2. Tras esto, el tamaño del área no era un problema. Con el fin de evitar errores, se establecerá este tamaño como límite para el sistema. En el siguiente código, se puede ver un ejemplo del proceso de descarga de imágenes en OpenEO.

```python
def cube_download(connection, dateIni, dateFin, cloud_cover,area):
        #Creacion del cubo de datos
        cube = connection.load_collection(
            "SENTINEL2_L2A", #Satelite que queremos usar
            temporal_extent=(dateIni, dateFin), #Rango de fechas
            spatial_extent={
                "west": area["west"],
                "south": area["south"],
                "east": area["east"],
                "north": area["north"],
                "crs": "EPSG:4326",
            }, #Coordenadas del area de interes
            bands=["B04", "B03", "B02"], #Bandas elegidas
            max_cloud_cover=cloud_cover, #Nubosidad
        )
        job = cube.create_job(output_format="GTiff") #Formato en el que
            queremos la imagen (GeoTiff)
        job.start_and_wait()
        results = job.get_results()
        directory = f"../data/area-{area['id']}"
        if not os.path.exists(directory):
            os.makedirs(directory)
        results.download_files(directory) #Directorio donde se va a guardar
            la imagen
```

Después de probar este código, surgió un nuevo problema. Los satélites de Copernicus dividen la superficie de la tierra en lo que se conoce como "Tiles". El nombre se debe a que la distribución es parecida a la de unos azulejos; un ejemplo de esto se puede apreciar en la figura 5.2. Cada cuadrado es un Tile diferente, y normalmente, cuando definimos unas coordenadas, esperamos obtener el Tile que más se ajuste a esas coordenadas. Por lo que se puede deducir, cuando OpenEO encuentra varios Tiles que cumplen los parámetros, parece elegir uno de manera aleatoria. En la figura 5.4, podemos ver un claro ejemplo de un conflicto entre Tiles. En la imagen se puede ver un Tile destacado en verde, el que nos interesa en este caso, y un cuadrado con puntos en las esquinas, que sería el área definida en el proceso de descarga. Si vemos la imagen detenidamente, podemos apreciar que los Tiles que rodean a nuestro Tile de interés también están dentro del área designada. Por lo que, la función de descarga elegirá uno de ellos aleatoriamente.

Afortunadamente, existe una solución para este problema. La función **Load Collection** mencionada previamente tiene un parámetro opcional llamado *properties* [8]. Este último sirve para especificar aún más el cubo de datos. Encontrar este parámetro resultó bastante complicado, ya que aunque se menciona en la documentación de OpenEO de manera superficial, en ningún lado se dice que pueda valer para especificar el Tile.

Después de especificar el Tile en la función de descarga, sigue existiendo la remota posibilidad de que la imagen descargada no corresponda al Tile deseado. Por ejemplo, si introducimos un margen de fechas muy pequeño, como un día, e intentamos descargar una imagen. Si no se

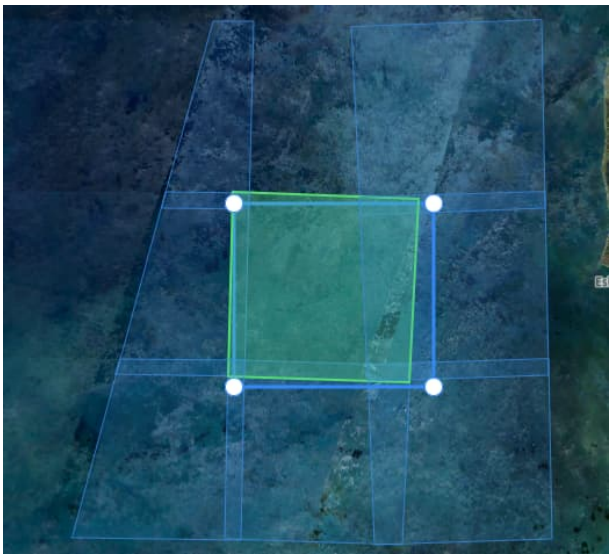

Figura 5.4: Conflicto entre tiles

sacó ninguna imagen de ese Tile en ese día, la función nos descargará una imagen de otro Tile. Pero como se ha mencionado, es un caso muy raro. De todos modos, para asegurarse de que en próximos apartados las imágenes de distintos Tiles no se mezclen, se añadió una comprobación que extrae el identificador del Tile de las imágenes descargadas y se guarda junto a cada imagen en un archivo JSON para más tarde utilizarlo a modo de comprobación.

5.2. Módulo de preprocesado

En el apartado de preprocesado, se diferencian tres tareas principales. Fusión temporal o *"Temporal Merging"*, Eliminado de tierra y Localización.

5.2.1. Temporal Merging

Un problema común al utilizar imágenes de satélite para la detección de estructuras como granjas eólicas, es la posible ambigüedad de la detección con otros tipos de estructuras presentes en el océano (por ejemplo, olas, barcos, nubes, objetos a la deriva, etc.). Estos objetos en movimiento pueden considerarse como ruido en una secuencia de imágenes (si lo que queremos detectar es una estructura estática). Para eliminarlos, se calculó una mediana temporal de cada píxel fusionando múltiples imágenes de la misma área (Kurekin et al., 2022) [1]. De esta manera, se eliminó la mayoría del ruido causado por objetos transitorios y solo se mantuvieron los objetos estáticos. Este proceso es conocido como fusión temporal y nos garantiza que los objetos que aparecen en las imágenes son totalmente estáticos y permanecen en el mismo lugar con el paso del tiempo, que es precisamente el caso de las granjas eólicas. En el caso de que nos interesen los objetos dinámicos como los barcos, simplemente hay que utilizar las imágenes sin fusionar para la detección, aunque estas tendrán mucho más ruido.

El primer problema surgió al hacer la primera prueba de fusión temporal, y es que a veces se estaba intentando hacer la mediana de dos imágenes pertenecientes a *Tiles* distintos. Por este motivo, como se menciona en el apartado de descarga, se implementó una función que recibiendo la retroalimentación de la descarga como parámetro, guarda en un archivo **JSON** el *Tile* de la zona que se acaba de descargar. Esto se hizo mediante un patrón **regex** que extrae el *Tile* de una *URL* que se genera cada vez que se descarga una imagen. Ahora, al tener cada imagen junto con su información en formato **JSON**, se puede hacer una comprobación a la hora de hacer la fusión temporal y descartar todas las imágenes que no pertenezcan al *Tile* del que queremos hacer la fusión.

El siguiente desafío encontrado durante la implementación del proceso de temporal merging fue la limitación de la memoria RAM del sistema. Este procedimiento, aunque no realiza un análisis directo píxel por píxel de cada imagen, implica una serie de operaciones que, en conjunto, resultan muy demandantes computacionalmente. El proceso comienza con recorrer un directorio que contiene múltiples subdirectorios, cada uno con imágenes en formato TIFF. Estas imágenes, que suelen ser de alta resolución y contener múltiples bandas espectrales, se cargan en la memoria como *Arrays* de *NumPy*. Esta carga masiva de datos representa un primer cuello de botella, sobre todo cuando se trabaja con grandes cantidades de imágenes, que es el caso. Una vez en memoria, el procedimiento continúa con el apilamiento de cada banda para todas las imágenes en estructuras tridimensionales (Tensores), donde cada capa representa una imagen distinta en el tiempo. Este apilamiento incrementa significativamente el uso de memoria, ya que se deben mantener múltiples copias de los datos simultáneamente.

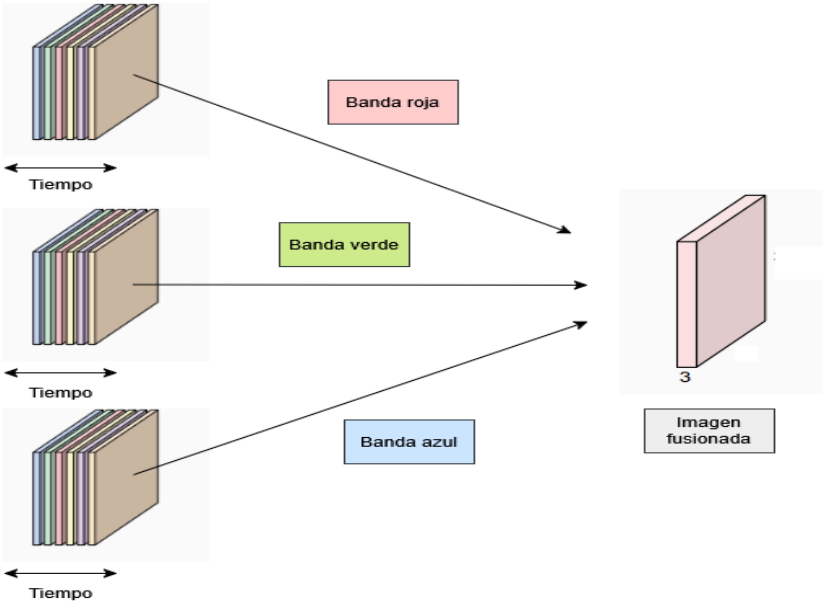

Figura 5.5: Proceso de fusión temporal

Después, se realiza el cálculo de la mediana para cada píxel a lo largo del eje temporal, es decir, considerando todas las imágenes disponibles. Esta operación es muy costosa por la necesidad de ordenar los valores para cada píxel antes de calcular la mediana. A diferencia

de otras operaciones, la mediana no puede resolverse mediante simples acumulaciones, lo que implica un mayor costo en términos de tiempo de procesamiento. Además, este cálculo debe repetirse para cada banda de la imagen, lo que multiplica la carga computacional. Finalmente, una vez obtenida la imagen fusionada, esta se escribe en el disco. En el siguiente código, se muestra la parte de la función de fusión temporal que se encarga de hacer estos cálculos.

```
try:
    #images es un array que guarda todas las imagenes validas
    if images:
        merged_bands = []
        #Iterar sobre cada banda por separado
        for band_idx in range(meta["count"]):
            #Crea un stack, donde se apilan todas las imagenes de la banda
                actual
            band_stack = np.stack([img[band_idx] for img in images], axis=0)
            band_stack = np.where(band_stack == 0.0, np.nan, band_stack)
            #Calcula la mediana de cada pixel ignorando los valores NaN
            merged_band = np.nanmedian(band_stack, axis=0)
            #Anade la banda a la lista
            merged_bands.append(merged_band)

        #Apila las bandas fusionadas en una imagen
        merged_image = np.stack(merged_bands, axis=0)

        #Se actualizan los metadatos
        meta.update(dtype="float32", count=merged_image.shape[0])

        #Se guarda la imagen fusionada
        output_path = os.path.join(output_dir, f"{area}_merged.tif")
        if os.path.exists(output_path):
            os.remove(output_path)

        with rasterio.open(output_path, "w", **meta) as dst:
            dst.write(merged_image.astype("float32"))
        print(f"Imagen fusionada guardada en: {output_path}")
except Exception as e:
    print(f"Error al procesar el area {area}: {e}")
```

Para evitar el problema de la memoria RAM, se han utilizado los servidores de *AZTI* para hacer la fusión temporal y luego los datos se han vuelto a transferir al ordenador original para seguir con el siguiente paso.

5.2.2. Eliminado de tierra

En este proyecto se trabaja con imágenes satelitales de zonas marítimas con el objetivo de detectar estructuras humanas como granjas eólicas marinas. Dado que estas estructuras son estáticas, su presencia es constante a lo largo del tiempo en las distintas capturas satelitales, al igual que ocurre con la línea de costa. Sin embargo, la presencia de tierra firme en las imágenes representa un elemento no deseado para el análisis, ya que introduce ruido en los resultados y

puede interferir en la detección precisa de objetos en el mar. Por este motivo, se implementa un proceso de eliminación de tierra cuyo propósito es enmascarar o excluir aquellas regiones de las imágenes que corresponden a zonas terrestres. Este procedimiento se basa generalmente en el uso de máscaras geoespaciales o en la aplicación de algoritmos de segmentación que permiten diferenciar entre superficie terrestre y marítima. Una vez identificadas las áreas correspondientes a tierra, estas se eliminan o se sustituyen por valores nulos (NaN) en las matrices de datos, de modo que no influyan en los análisis posteriores, como la detección de objetos. Este paso es fundamental para garantizar que el análisis se centre exclusivamente en el entorno marino, mejorando así la precisión y la eficiencia del sistema de detección.

El primer problema surgió a la hora de encontrar una máscara válida para eliminar la tierra de la zona de Dinamarca y Países Bajos. Tras hacer unas búsquedas, se encontró un archivo en formato **shapefile (.shp)** que contenía una máscara para toda Europa. Este archivo está disponible en la página de la Agencia Medioambiental Europea. En la figura 5.6 se puede apreciar como es esta máscara creada por la Agencia, y a pesar de que es muy precisa, tiene un gran inconveniente, y es que, como se puede ver, es una zona de marismas lo que dificulta más el proceso de eliminado de tierra ya que las imágenes resultantes van a tener la línea de costa eliminada de manera muy irregular.

Para solucionar este problema, se decidió crear una máscara de manera manual en *QGis* partiendo de la máscara de Europa. Este proceso es un poco lento si no se conoce la herramienta de QGis, pero tras definir la máscara se obtiene lo que se aprecia en la figura 5.7. A primera vista, puede parecer un arreglo inestable y de baja calidad pero a continuación, se puede ver una muestra de su eficacia en la figura 5.8 donde se probó a eliminar la tierra de la misma imagen utilizando primero la máscara de la Agencia Europea, y luego la máscara hecha a mano. Como se puede ver, el resultado de la máscara hecha a mano es más estable, ya que elimina toda la tierra. De este modo, se evitarán posibles confusiones en el proceso de detección.

5.2.3. Localización

Como última tarea del procesado, se encuentra la localización o identificación de granjas eólicas. Este proceso tiene como objetivo destacar la actividad humana en cada imagen para que sea más llamativa y de este modo se puede analizar de manera visual. Para lograr este cometido, se ha utilizado un umbral y se ha pasado por cada píxel de la imagen fusionada, haciendo un filtrado. Como los objetos más interesantes tienen un brillo mayor al de los objetos dinámicos, se puede establecer un umbral, para convertir en negro todos los píxeles menores al umbral y destacar los mayores. Teniendo en cuenta que las imágenes RGB están formadas por píxeles en el rango de $[0 - 255]$, un umbral adecuado podría estar entre $[200 - 245]$.

En el caso de las imágenes SAR, al presentar menos ruido y solo tener 2 bandas, este proceso no será necesario. Solamente convirtiéndola en una imagen RGB pasando la banda VV a la banda roja, la banda VH a la banda verde y dejando la banda azul con valor de 0, se destacarán los píxeles más interesantes de rojo.

Si colocamos las imágenes resultantes en un mapa, podría obtenerse un primer resultado. Ya que se verán destacadas las granjas eólicas colocadas en su respectiva posición en el mapa.

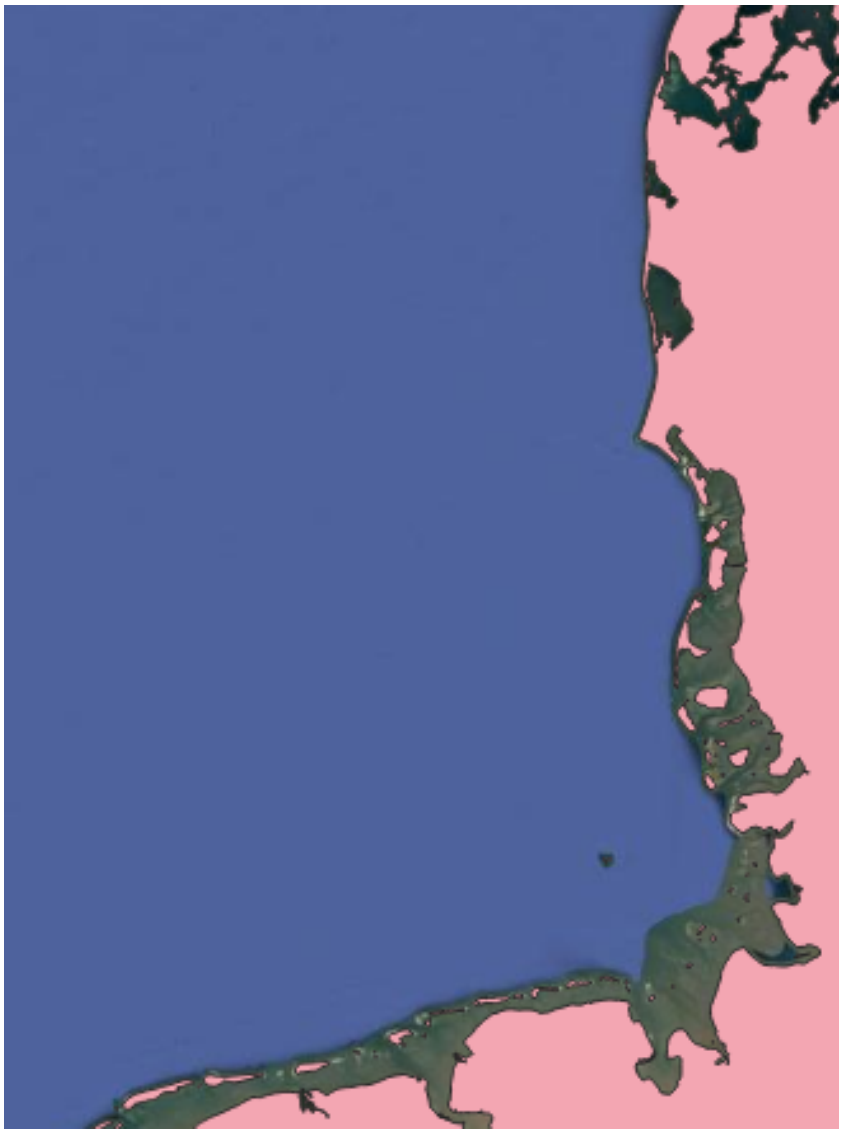

Figura 5.6: Máscara de la Agencia Medioambiental Europea

En la Figura 5.9 se muestra un ejemplo de la identificación de granjas eólicas con imágenes ópticas, donde la granja se puede apreciar en la parte inferior y a la izquierda de la imagen.

5.3. Módulo de detección

El módulo de detección es el encargado de detectar con exactitud la actividad humana a partir de las imágenes obtenidas. Una primera aproximación fue mediante la segmentación de imágenes, pero debido a la cantidad insuficiente de datos de entrenamiento, esta idea fue descartada.

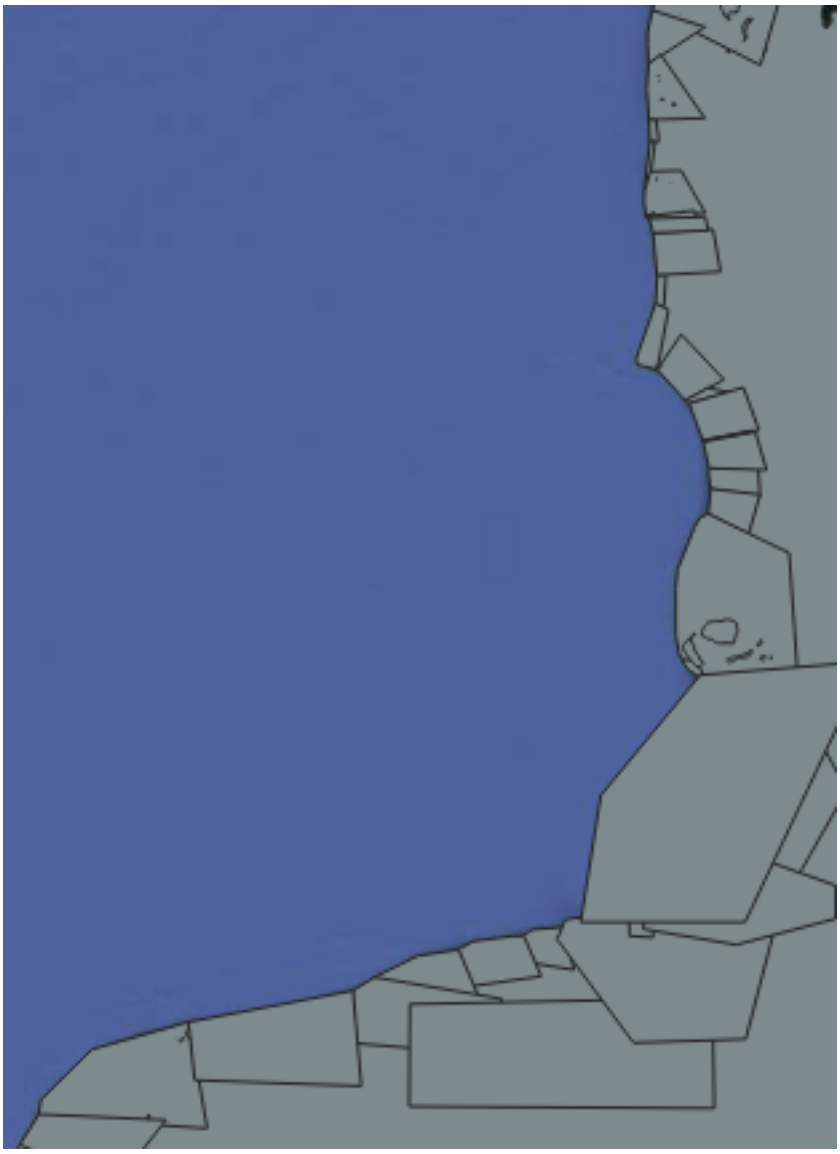

Figura 5.7: Máscara hecha a mano

La segunda idea para llevar a cabo esta tarea fue entrenar una pequeña red neuronal para detectar las granjas eólicas. Para crear el conjunto de entrenamiento se empleó la técnica de etiquetado "*bounding boxes*". Para esto se utilizaría el Software de etiquetado **CVAT**.

En el primer intento, se clasificó el contenido de las imágenes en Ruido y Granja Eólica. Esto se hizo con todas las imágenes del Mar del Norte tras pasar por el proceso de detección. Tras etiquetar manualmente cada imagen, se exportó el conjunto de datos listo para entrenar una red de detección de objetos. En este caso se utilizó la red YoloV8, que es una red especializada en detección. Para el primer entrenamiento, se utilizó la versión *"n"* que es la más pequeña del catálogo. Después del entrenamiento, se comprobó que los resultados eran muy pobres y que

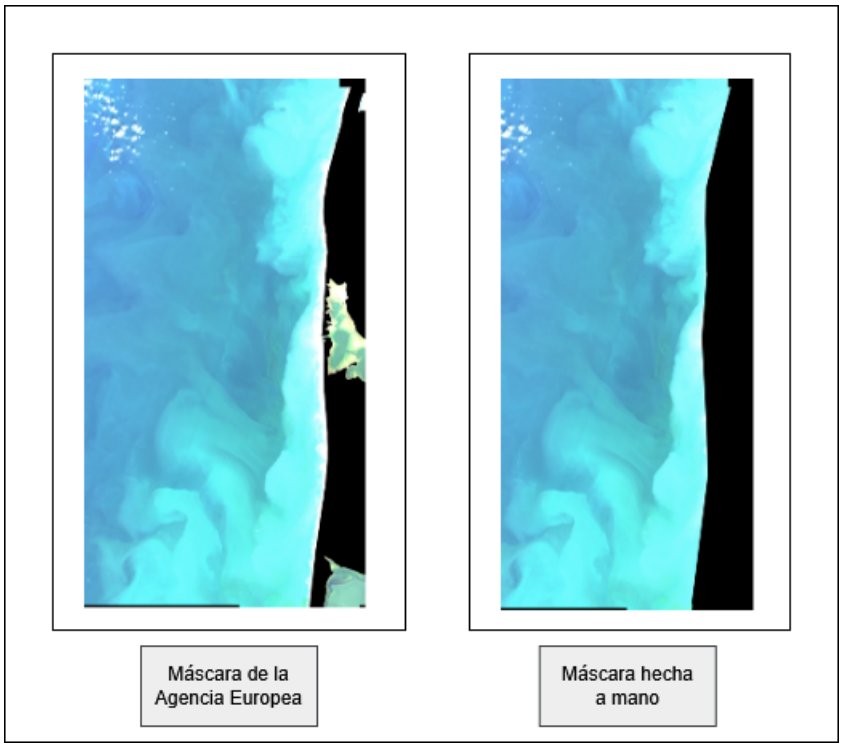

Figura 5.8: Comparación de las dos máscaras en la misma imagen

apenas detectaba objetos.

Como se vio que la red tenía problemas con la etiqueta de ruido, se volvió a etiquetar el conjunto de entrenamiento entero, pero esta vez, solo con la etiqueta de ”Granja Eólica”, para simplificar el modelo. En esta segunda prueba se utilizó la versión *”x”* que se trata de la más grande, y después de un entrenamiento algo más largo, esta vez los resultados eran muy diferentes. Ahora, en vez de no detectar casi ningún objeto, detectaba granjas donde no las había, quedando una imagen resultante difícil de analizar. Como todas las pruebas de entrenar un modelo habían fracasado, se tomó la decisión de no implementar el módulo de clasificación, ya que se trataba de un objetivo opcional. En su lugar, se consiguió que el sistema funcionara tanto con imágenes ópticas como SAR, dando la opción de elegir entre uno de estos tipos de imagen. También se decidió referirse a la función de localización como detección en la interfaz de usuario, ya que era lo más parecido a esta que había.

A pesar de no haber conseguido que el proceso de clasificación sea funcional, el sistema tiene la base adecuada para trabajar desde el punto actual, y si en un futuro alguien quisiera retomar este proyecto, tendría el flujo de trabajo completo implementado y en caso de disponer de más datos, podría utilizar la parte de clasificación.

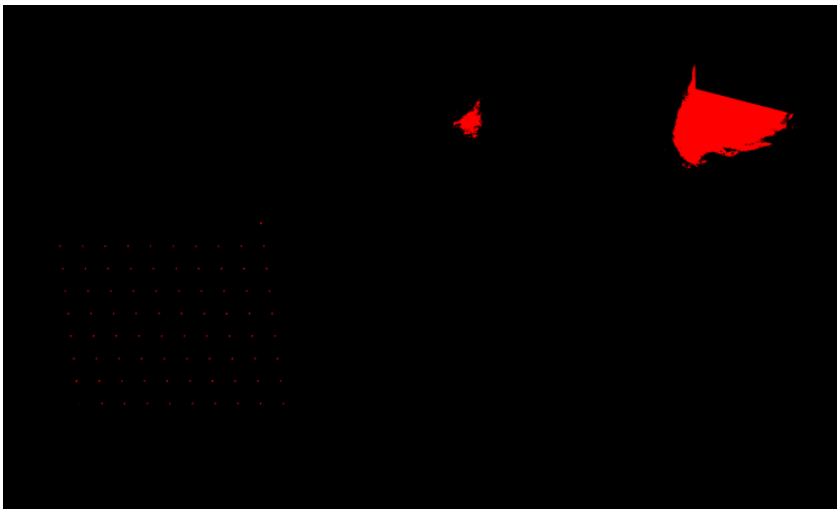

Figura 5.9: Resultado de la localización de granjas eólicas

CAPÍTULO 6

Pruebas

En este capítulo se mencionan todas las pruebas o "*tests*"que se han hecho para comprobar la funcionalidad del sistema ante cualquier situación.

6.1. Pruebas realizadas

Las primeras pruebas se remontan al inicio del proyecto, cuando todavía se estaba haciendo un estudio de las tecnologías. Para hacer todas las pruebas necesarias, se creó un cuaderno de **Jupyter Notebook (.ipynb)** por la comodidad que ofrece al poder ejecutar el código en celdas individuales. En ese cuaderno, se exploró en profundidad la función para descargar datos haciendo decenas de pruebas en distintas zonas. También se creó el código para extraer los metadatos de las imágenes descargadas, analizando el link que se obtiene al descargarlas. Más tarde, se creó otro cuaderno para hacer pruebas con las imágenes ya descargadas, como por ejemplo el eliminado de tierra y la visualización de múltiples imágenes en un mapa interactivo. Una vez hechas todas las pruebas necesarias para entender las bases de las tecnologías, se empezó con el sistema.

Al terminar el producto mínimo viable, el cual permitía hacer descargas solo del área de estudio y con imágenes ópticas, empezaron las pruebas para verificar su funcionamiento. Como solo permitía descargarse el área de estudio, el único parámetro variable eran las fechas, para las cuales se utilizó un rango desde mayo de 2024 hasta finales de agosto del mismo año. Se inició la descarga de datos y se dejó en segundo plano durante toda una noche ya que se trataba de una gran cantidad de datos. Al terminar la descarga, todas las imágenes de cada área se habían guardado de forma exitosa, así que era momento de pasar al preprocesado. El preprocesado fue bastante rápido porque se ejecutó en el servidor de **AZTI**, tras ver que se había completado sin ningún problema, solo quedaba la detección y visualización que se completaron en un momento. En la figura 6.1 se muestra el mapeado final que se obtuvo tras la prueba. Donde por el tamaño no se aprecia ninguna granja eólica, pero se puede ver el ruido y que cada foto se coloca en su posición real en el mapa.

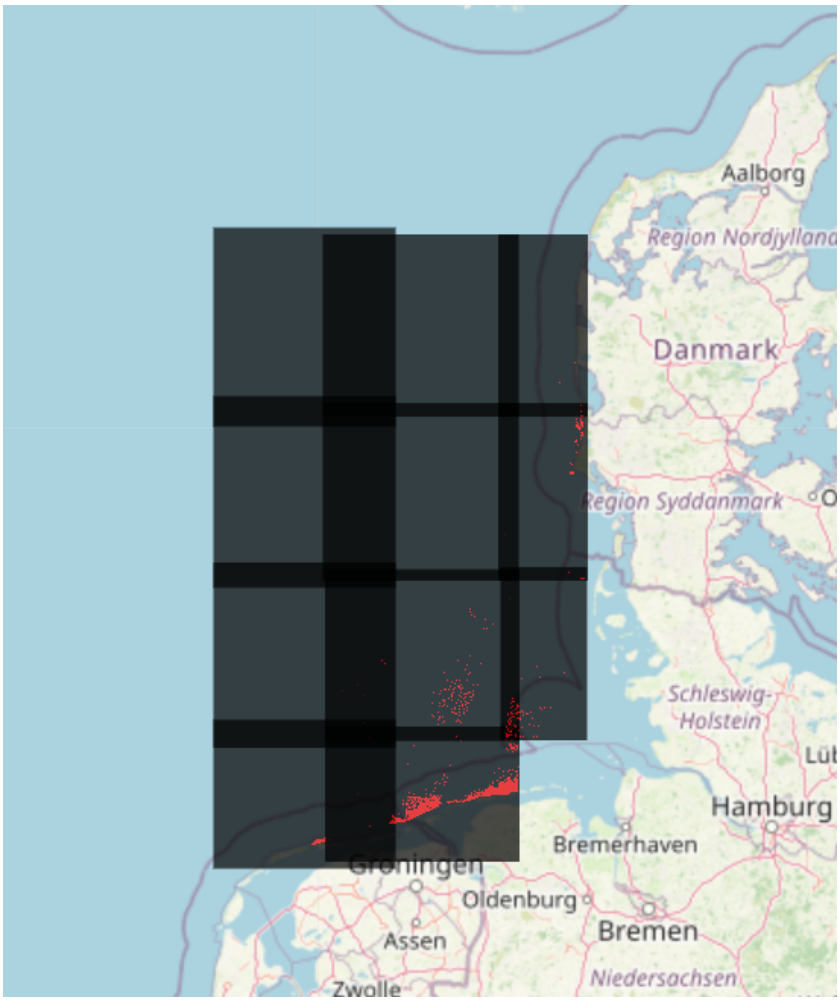

Figura 6.1: Prueba de la zona de estudio con imágenes ópticas

Tras la primera prueba se hicieron avances en el sistema, y ahora, permitía definir una zona personalizada y elegir si descargar imágenes ópticas o SAR. Para esta prueba se seleccionó una pequeña área de la costa de Dinamarca, exactamente al lado del poblado de **Oksby**, se eligió esta zona porque hay una granja eólica con una forma cuadrada muy fácil de reconocer. En la prueba de Optical se volvió a elegir el mismo periodo de fechas, y la descarga fue mucho más rápida que la anterior debido a la pequeña área de la zona. Tras completar todos los pasos la prueba fue exitosa. Sin embargo, al intentar descargar la misma zona con imágenes SAR ocurrió un pequeño imprevisto, y una de las imágenes descargadas estaba corrupta. Esto no se trata de ninguna falla en el sistema, sino que es un problema de **Copernicus**. Por motivos desconocidos, algunas imágenes carecen de sistema de referencia espacial (SRS) y estas se descargan como archivos corruptos que no se pueden visualizar. La solución es sencilla: descartar del procesado cualquier imagen que no esté georreferenciada. Es un factor que no se tuvo en cuenta a la hora de hacer el programa y gracias a la prueba se pudo detectar a tiempo.

6.2. Automatización

En primer lugar se planteó hacer un *"testeo"* automático del sistema, pero se descartó por varios motivos:

- **Necesidad de supervisión humana:** La validación de los resultados implica verificar manualmente si lo detectado realmente está presente en las imágenes. Esto requiere interpretación visual, algo que los tests automatizados no pueden hacer de forma fiable.
- **Falta de datos de referencia:** No se dispone de una base de datos anotada de manera fiable que indique qué imágenes contienen granjas eólicas y cuáles no. Sin estos datos, no es posible comparar automáticamente los resultados obtenidos con un resultado esperado".
- **Coste de preparación de casos de prueba:** Para cubrir los diferentes escenarios, habría que preparar manualmente una gran cantidad de áreas de prueba, lo cual implicaría mucho tiempo y recursos sin aportar mejoras directas al desarrollo.
- **Elevado peso de los datos y lentitud del proceso:** Cada ejecución puede tardar minutos u horas dependiendo del área, las fechas y el tipo de imagen. Esto hace que ejecutar tests sea costoso en tiempo.

6.3. Validar resultados

Como se menciona en el apartado anterior, no existen maneras fiables para comprobar que en los resultados obtenidos, efectivamente, haya granjas eólicas. Si al ver una imagen, se puede diferenciar un conjunto de puntos llamativos colocados de manera aparentemente ordenada, se puede intuir que se trata de una granja eólica, pero no hay manera precisa de confirmarlo, ya que la información exacta es propiedad del gobierno al que pertenezcan las granjas y ellos deciden si compartirla o no.

Sin embargo, existen maneras de comprobar algunos resultados. La web de 4C Offshore muestra un mapa donde se ven algunas zonas marcadas con colores, que indican la presencia de windfarms o si va a haberlas en un futuro. Otra alternativa es consultar la página de Wikipedia que aparentemente muestra un listado completo de todas las granjas que existen en el mar del norte, sus coordenadas y a qué país pertenecen. Aunque tras realizar el proyecto, se han descubierto granjas que no aparecen en este listado, así que esta lista no es del todo fiable por el momento.

CAPÍTULO 7

Seguimiento

En este capítulo se documentan todas las tareas relacionadas con el seguimiento del proyecto, desde la revisión del alcance, hasta la gestión del tiempo y riesgos.

7.1. Revisión del Alcance

El alcance definido al inicio del proyecto se ha mantenido estable a lo largo del desarrollo. El objetivo principal de detectar y mapear actividad humana en el mar se ha respetado y ha permitido explorar aplicaciones reales en el ámbito de la teledetección.

En cuanto a los objetivos complementarios, todos han sido abordados de forma satisfactoria:

- **Aprender a trabajar con datos satelitales:** Se ha trabajado con imágenes del satélite SENTINEL-2, utilizando herramientas de la plataforma Copernicus. Esto ha permitido adquirir experiencia en la descarga, interpretación y preprocesamiento de datos ópticos.
- **Desarrollar un sistema de software para el mapeo:** Se ha implementado un sistema que integra los distintos componentes necesarios (descarga, preprocesado y detección), cumpliendo con el objetivo de crear una solución funcional y modular.
- **Aplicar conocimientos adquiridos en la carrera:** El proyecto ha permitido aplicar conceptos de visión por computador, aprendizaje automático y desarrollo de software, consolidando así los conocimientos adquiridos durante el grado.

En resumen, el alcance inicial se ha cumplido en su totalidad, sin necesidad de modificaciones significativas. Esto ha facilitado una ejecución coherente y alineada con los objetivos planteados desde el inicio.

7.2. Gestión del tiempo

En esta sección, se verán las desviaciones de fechas y dedicaciones respecto a la planificación original.

7.2.1. Desviaciones de fechas

En primer lugar, se detallarán las desviaciones respecto a los hitos establecidos para el proyecto. En la tabla 7.1 se contrastan las fechas de finalización de cada hito con las establecidas inicialmente.

Hito	Fecha estimada	Fecha real
Inicio del proyecto	17-02-2025	17-02-2025
Descarga (I.D)	02-04-2025	07-04-2025
Preprocesado (I.P)	25-04-2025	21-04-2025
Detección (I.DET)	16-05-2025	19-05-2025
Fin Implementación	06-06-2025	08-06-2025
Fin memoria	08-06-2025	10-06-2025

Tabla 7.1: Desviaciones en los hitos del proyecto

Como se puede apreciar, en general no ha habido grandes desviaciones en cuanto a las fechas de finalización de cada hito. La descarga se alargó unos días más de lo esperado, ya que se encontraron bastantes problemas a la hora de la implementación de esta parte. El preprocesado resultó ser más corto de lo esperado, ya que se utilizaron técnicas ya creadas por otras personas y tan solo se tuvo que adaptar a este contexto. La detección no se desvió mucho respecto a la finalización original así que podría decirse que su duración se estimó adecuadamente. El fin de la memoria y la implementación tenían una fecha bastante parecida y al final solo se desvió un par de días cada uno.

7.2.2. Desviaciones en las dedicaciones

Ahora, se mostrarán las desviaciones en la dedicación en horas de cada tarea. Estas desviaciones se muestran en la tabla 7.2

Al igual que en la sección anterior, tampoco ha habido grandes desviaciones en cuanto a las dedicaciones. Las tareas de estudio de las tecnologías, descarga, detección y escritura de la memoria han llevado un poco más de tiempo de lo esperado. Por otro lado, las tareas de preprocesado y visualización un poco menos.

7.3. Gestión de riesgos

Durante el desarrollo del proyecto se identificaron tres riesgos principales. A continuación, se analiza cómo han evolucionado y si se han materializado:

TAREA	DEDICACIÓN ESTIMADA	DEDICACIÓN REAL
Estudio (E)		
Revisión de Literatura (E.R)	15h	15h
Estudio de las tecnologias (E.E)	10h	15h
Subtotal	25h	30h
Implementación (I)		
Descarga (I.D)	35h	40h
Preprocesado (I..P)	40h	30h
Detección (I.DET)	50h	60h
Visualización (I.V)	10h	5h
Clasificación (I.C)	40h	40h
Documentación del código (I.DC)	10h	10h
Subtotal	185h	185h
Documentación (D)		
Escritura de la Memoria (D.M)	80h	90h
Subtotal	80h	90h
Seguimiento (S)		
Reuniones (S.R)	20h	20h
Subtotal	20h	20h
Total	310h	325h

Tabla 7.2: Dedicación real de cada tarea

- **Riesgo 1 - Estudios:** Este riesgo hacía referencia a la posible dificultad de compaginar el proyecto con las asignaturas del segundo cuatrimestre. Aunque en algunos momentos puntuales como las semanas de horario agrupado o en los exámenes finales se ha producido un ligero retraso en ciertas tareas, en general este riesgo ha estado bien gestionado gracias a una planificación realista que tuvo en cuenta la carga académica. No ha supuesto un impacto significativo en el desarrollo global del proyecto.

- **Riesgo 2 - Tecnologías desconocidas:** Este ha sido el riesgo que más impacto ha tenido. El uso de tecnologías nuevas, especialmente relacionadas con la teledetección y el procesamiento de imágenes satelitales, ha supuesto una curva de aprendizaje considerable. En algunos casos, esto ha ralentizado el avance de ciertas tareas, aunque se ha podido compensar con una mayor dedicación, búsqueda de recursos de aprendizaje y apoyo de expertos en AZTI. La previsión de dedicar más tiempo al estudio inicial fue acertada, aunque podría haberse ampliado aún más.

- **Riesgo 3 - Planificación inadecuada:** A pesar de la incertidumbre inicial, este riesgo no se ha materializado. La planificación resultó ser adecuada y realista, en parte gracias a la consulta con el responsable del proyecto en la empresa y con reuniones semanales de seguimiento de la planificación. Las tareas y tiempos asignados se ajustaron bien a la realidad del desarrollo.

7.4. Desviaciones de tareas

Durante el desarrollo del proyecto, se decidió no implementar el apartado opcional de clasificación por priorizar el procesado también de imágenes SAR y por la falta de un conjunto de entrenamiento adecuado en la literatura y disponible de forma abierta. Se consideró que el tiempo para etiquetar imágenes para generar un conjunto de entrenamiento propio excedía el tiempo disponible y se considera trabajo futuro.

CAPÍTULO 8

Conclusiones

En esta sección se detallan todas las conclusiones del proyecto, desde limitaciones o problemas con la **API** de **OpenEO**, hasta las nuevas competencias que se han aprendido o han sido necesarias para la implementación del proyecto.

8.1. Conclusiones generales

El objetivo inicial de la aplicación era crear un sistema capaz de mapear la actividad descargando y procesando imágenes ópticas. Durante el desarrollo del proyecto, se decidió implementar estas funcionalidades para imágenes SAR también. Así que, en estas conclusiones, se comentarán las ventajas y desventajas de cada alternativa y se compararán ambas entre sí.

Por un lado, tenemos las imágenes ópticas. Su principal ventaja es que proporcionan información a color (RGB), lo que permite una interpretación visual intuitiva. Esta característica puede ser útil para identificar estructuras artificiales o cambios en el entorno que sean evidentes a simple vista. Pero, en el mapeo de granjas eólicas, esta ventaja resulta limitada. Las condiciones meteorológicas, como las nubes o las corrientes de agua, afectan significativamente la calidad de las imágenes ópticas. Además, a la hora de procesar las imágenes, el hecho de que sean a color no presenta ninguna ventaja significativa.

Por otro lado, las imágenes SAR presentan una serie de ventajas clave para este tipo de aplicaciones. Al no depender de la luz solar ni verse afectadas por las condiciones meteorológicas, son más fiables. Esto es especialmente relevante en zonas costeras o marinas, donde la nubosidad es habitual. Además, las imágenes SAR son menos sensibles al ruido generado por las mareas o las marismas, lo que mejora la estabilidad del análisis temporal. Aunque no ofrecen información en color, su capacidad para detectar estructuras en la superficie del agua las convierte en una herramienta muy potente para el seguimiento de actividad humana como las granjas eólicas.

En resumen, aunque las imágenes ópticas pueden resultar útiles en condiciones ideales, las

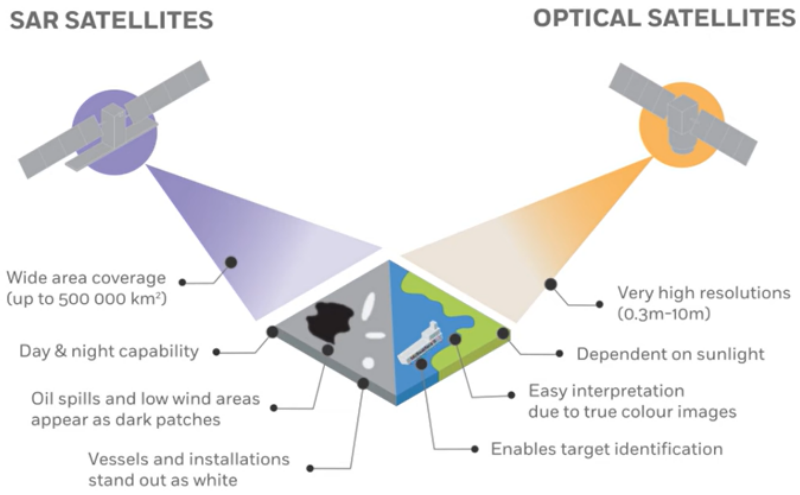

Figura 8.1: Diferencias entre SAR y Optical (Fuente: Cervest)

imágenes SAR ofrecen una robustez y continuidad que las hacen más adecuadas para el mapeo sistemático de actividades humanas en entornos acuáticos. La combinación de ambas fuentes podría ser una vía interesante para futuras mejoras del sistema, aprovechando lo mejor de cada tecnología.

8.2. Limitaciones de OpenEO

Uno de los objetivos del proyecto era utilizar la **API** de **OpenEO** para la descarga de datos. Aunque ha cumplido su función, en esta sección se mencionan los aspectos que podrían mejorarse:

- **Conflictos entre zonas:** Como se menciona en el capítulo 5, uno de los mayores inconvenientes al definir un área manualmente es que varias *tiles* o *slices* pueden cumplir con los parámetros establecidos. Al intentar descargar la imagen, **OpenEO** puede devolver cualquiera de las zonas que coincidan con dichos parámetros. La solución consiste en definir el *tile* específico del cual se desean descargar los datos, información que puede consultarse fácilmente en el buscador de **Copernicus**. Aunque esto pueda parecer una limitación, lo cierto es que las fechas de descarga también deben consultarse en ese mismo buscador, por lo que ambos datos pueden obtenerse simultáneamente.

- **Parámetros de descarga insuficientes:** En el punto anterior se menciona que el conflicto entre zonas puede resolverse especificando el *tile* deseado. Sin embargo, esta solución solo es válida para imágenes ópticas, ya que las imágenes SAR se dividen en *slices*. Estos *slices* también cuentan con un identificador único, pero **OpenEO** no permite filtrar los datos en base a ellos, como sí ocurre con los *tiles*. Por este motivo, al descargar imágenes SAR, si dos *slices* coinciden en el área definida, no se puede garantizar que se obtenga

la imagen deseada. Aunque los *slices* son significativamente más grandes que los *tiles*, lo que reduce la frecuencia del problema, este puede seguir ocurriendo.

- **Feedback de descarga:** Otro inconveniente relacionado con los anteriores es la falta de **feedback** por parte de la función de descarga. Esta devuelve un objeto llamado **results**, en el que se esperaría encontrar los metadatos de las imágenes recién descargadas. Sin embargo, solo se obtiene un objeto **JSON** lleno de enlaces extensos. Aunque es posible encontrar algunos metadatos útiles, su accesibilidad es limitada y, para extraerlos, es necesario aplicar patrones **regex**.
- **Tamaño de las zonas de descarga:** Este problema se refiere a la clara limitación de **OpenEO** para manejar datos de gran tamaño. Aunque la documentación ofrece una forma sencilla y directa de descargar datos, también indica que para zonas más grandes es necesario recurrir a trabajos por lotes. En la práctica, incluso al intentar descargar áreas relativamente pequeñas, ya es necesario utilizar este tipo de trabajos, que resultan considerablemente más lentos.

8.3. Nuevas competencias

Estas son las competencias que se han adquirido a lo largo del desarrollo del proyecto:

- **Manejo de QGIS:** Durante el proyecto, QGIS ha sido una herramienta fundamental para la visualización y análisis de los resultados. Se ha utilizado para inspeccionar las imágenes obtenidas en cada etapa del procesamiento, lo que ha permitido validar visualmente la calidad y precisión de los datos. Además, se ha trabajado con capas personalizadas, incluyendo la creación de nuevas capas a partir de otras existentes, lo que ha requerido un conocimiento práctico de las herramientas de edición y gestión de capas dentro del entorno de QGIS.
- **Manejo de datos de gran tamaño:** El trabajo con imágenes satelitales implica el manejo de archivos de gran tamaño y en ocasiones en grandes cantidades. Esto ha requerido una organización cuidadosa de los datos, estructurando los directorios de forma eficiente y desarrollando scripts capaces de recorrer automáticamente estas estructuras para procesar los archivos. Esta experiencia ha sido clave para mejorar la eficiencia del flujo de trabajo y evitar errores derivados del manejo manual de grandes volúmenes de información.
- **Visión por computador:** Se han desarrollado diversas funciones relacionadas con el procesamiento de imágenes, como la modificación de imágenes o la fusión de estas. Estas tareas han permitido adquirir conocimientos prácticos en técnicas básicas de visión por computador, así como en el uso de librerías especializadas.
- **Gestión de proyectos:** El proyecto ha supuesto un reto en cuanto a la planificación y gestión del tiempo, debido a su duración y complejidad. Ha sido necesario establecer una planificación clara, definir hitos. Esta experiencia ha contribuido al desarrollo de

habilidades de organización, priorización de tareas y toma de decisiones en contextos de incertidumbre, competencias esenciales para abordar proyectos de mayor envergadura en el futuro.

8.4. Aportaciones novedosas respecto a trabajos previos

Durante el desarrollo de este proyecto, se han realizado múltiples aportaciones que han contribuido de forma significativa al avance del sistema de detección de granjas eólicas mediante imágenes satelitales. A continuación, se detallan algunas de las principales contribuciones técnicas y prácticas al proyecto:

- **Automatización del proceso de descarga de imágenes satelitales**: Se investigó y desarrolló un sistema capaz de descargar datos del satélite Sentinel-2 y Sentinel-1 utilizando la API de OpenEO. Se implementó una función robusta que permite definir áreas de interés, rangos temporales y cobertura de nubes.
- **Fusión temporal de imágenes**: Se desarrolló un módulo de *Temporal Merging* que permite fusionar múltiples imágenes de la misma zona para eliminar objetos transitorios como barcos o nubes. Optimizando el uso de recursos, estos procesos se ejecutaron en servidores de la empresa para superar las limitaciones de memoria de los equipos locales.
- **Capa de tierra manual**: Se modificó la capa de la Agencia Europea de Medio Ambiente para ajustarla a las necesidades del proyecto, ya que esta era demasiado específica teniendo en cuenta zonas de marismas pequeñas que dificultaban la detección de granjas eólicas.

Estas tareas han implicado tanto investigación como desarrollo, y han permitido aplicar conocimientos de programación, tratamiento de imágenes, uso de APIs geoespaciales, y aprendizaje automático. Considero que todas estas contribuciones no solo han ayudado a avanzar el proyecto, sino que también han generado herramientas que la empresa podrá reutilizar o ampliar en el futuro.

8.5. Integración en equipo de AZTI

Durante mi tiempo en la empresa, he logrado desenvolverme bien en la parte técnica del proyecto, a pesar de su complejidad. Me he organizado con autonomía para avanzar en el trabajo, y he sabido adaptarme a los cambios que han ido surgiendo, colaborando estrechamente con varios expertos para desarrollar el proyecto de forma efectiva.

Bibliografía

[1] Arlene L. Avillanosa Joel D. C. Sumeldan Andrey A. Kurekin, Peter I. Miller. Monitoring of coastal aquaculture sites in the philippines through automated time series analysis of sentinel-1 sar images. *MDPI*, 2022. Ver página 29.

[2] Arantza Murillas Kemal Pınarbaşı Jose A. Fernandes Caroline Coccoli, Ibon Galparsoro. Conflict analysis and reallocation opportunities in the framework of marine spatial planning: A novel, spatially explicit bayesian belief network approach for artisanal fishing and aquaculture. 2018. Ver página 1.

[3] Emilien Pecoul Laurent Demagistri Michel Petit Christina Corbane, Laurent Najman. A complete processing chain for ship detection using optical satellite imagery. 2010. Ver página 3.

[4] Copernicus. *openEO Basics: How to load a data cube from a data collection.* URL: https://documentation.dataspace.copernicus.eu/notebook-samples/openeo/Load_Collection.html. Ver página 25.

[5] Shufan Ji Feng Gao Qizhi Xu Guang Yang, Bo Li. Ship detection from optical satellite images based on sea surface analysis. 2014. Ver página 3.

[6] Joxe Mikel Garmendia Ángel Borja Ana D. Maldonado Gregorio Iglesias Juan Bald Ibon Galparsoro, Iratxe Menchaca. Reviewing the ecological impacts of offshore wind farms. *Nature*, 2022. Ver página 1.

[7] OpenEO Organization. *openEO Cookbook.* URL: https://openeo.org/documentation/1.0/cookbook/#chapter-1. Ver página 25.

[8] OpenEO Organization. *openEO: Finding and loading data.* URL: https://open-eo.github.io/openeo-python-client/data_access.html#filter-on-collection-properties. Ver página 28.

[9] OpenEO Organization. *openEO Python Client.* URL: https://openeo.org/documentation/1.0/python/#installation. Ver página 25.

[10] Claudia Kuenzer Thorsten Hoeser, Stefanie Feuerstein. Deepowt: a global offshore wind turbine data set derived with deep learning from sentinel-1 data. *ESSD*, 2022. Ver página 3.

[11] Fashuai Li Yuwei Chen Maoxin Zhang Qiming Zheng Yukan Jin He Ren Tingting He, Yihua Hu. Mapping land- and offshore-based wind turbines in china in 2023 with sentinel-2 satellite data. *ScienceDirect*, 2025. Ver página 3.

[12] Konstantinos N. Topouzelis. Oil spill detection by sar images: Dark formation detection, feature extraction and classification algorithms. 2008. Ver página 2.

[13] Wei Wu Yanzhu Dong Wanyun Lu Yongchao Liu Bingxue Zhao Huiting Li Renfei Yang Wenxuan Xu, Yongxue Liu. Proliferation of offshore wind farms in the north sea and surrounding waters revealed by satellite image time series. *ScienceDirect*, 2020. Ver página 3.

[14] Ignacio Arganda-Carreras Jose A. Fernandes-Salvador Xabier Lekunberri, J. David Ballester-Berman c. Automatic mapping of aquaculture activity in the atlantic ocean. *ScienceDirect*, 2024. Ver página 2.

[15] Yuexia Wang Xiaoyue Wang Shuaikang Xue Weixue Liu Zhihuo Xu, Hongchen Zhang. Dynamic detection of offshore wind turbines by spatial machine learning from spaceborne synthetic aperture radar imagery. *ScienceDirect*, 2022. Ver página 3.