



UNIVERSIDAD DE BURGOS  
ESCUELA POLITÉCNICA SUPERIOR  
Grado en Ingeniería Informática



## TFG del Grado en Ingeniería Informática

Análisis de imágenes  
hiperespectrales para la  
detección de cobre en viñedo  
(proyecto Dig4Vitis v2)



Presentado por Diego Urbaneja Portal  
en Universidad de Burgos — 6 de julio de 2025  
Tutores: Carlos Cambra Baseca y Ramón  
Sánchez Alonso







UNIVERSIDAD DE BURGOS  
ESCUELA POLITÉCNICA SUPERIOR  
Grado en Ingeniería Informática



Dr. Carlos Cambra Baseca y Dr. Ramón Sánchez Alonso, del departamento de Digitalización.

Exponen:

Que el alumno D. Diego Urbaneja Portal, con DNI 71305446H, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado Análisis de imágenes hiperespectrales para la detección de cobre en viñedo (proyecto Dig4Vitis v2).

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 6 de julio de 2025

Vº. Bº. del Tutor:

D. Carlos Cambra Baseca

Vº. Bº. del co-tutor:

D. Ramón Sánchez Alonso





## Resumen

Este trabajo, enmarcado en el área de la agricultura de precisión, se centra en el desarrollo y la implementación de una herramienta software, **EcoVid**, para la detección y cuantificación de tratamientos antifúngicos con base de cobre en hojas de viñedo.

El objetivo principal es analizar imágenes hiperespectrales para calcular de forma precisa el porcentaje de recubrimiento del producto sobre la superficie foliar. Para ello, se ha desarrollado una aplicación web en Python con la biblioteca Streamlit , que permite al usuario procesar dos imágenes hiperespectrales de la misma hoja: una con el tratamiento aplicado y otra sin él. La innovación clave del proyecto reside en la implementación de un algoritmo para la reducción de ruido y falsos positivos. Este sistema utiliza el algoritmo ORB (Oriented FAST and Rotated BRIEF) para alinear automáticamente ambas imágenes, permitiendo sustraer el ruido (variaciones naturales de reflectancia de la propia hoja) detectado en la imagen de la hoja sin tratamiento de control de la imagen tratada.

Como resultado, **EcoVid** ofrece una imagen procesada que muestra, de forma resaltada, la parte de la hoja de vid afectada por el tratamiento de producto y el porcentaje de cobertura, de forma numérica. La aplicación ofrece la opción de descargar los resultados en formato .jpg.

El proyecto concluye con una herramienta funcional que ofrece resultados inmediatos e implementables en un entorno real de trabajo en viñedo.

## Descriptores

Imágenes hiperespectrales, agricultura de precisión, detección de cobre, visión por computador, algoritmo ORB, reducción de ruido, viñedo, Python.

## Abstract

This work, framed within the area of precision agriculture, focuses on the development and implementation of a software tool, **EcoVid**, for the detection and quantification of copper-based antifungal treatments on vineyard leaves. The main objective is to analyze hyperspectral images to accurately calculate the percentage of product coverage on the foliar surface.

To this end, a web application has been developed in Python using the Streamlit library, which allows the user to process two hyperspectral images of the same leaf: one with the treatment applied and the other without it. The project's key innovation lies in the implementation of an algorithm for the reduction of noise and false positives. This system uses the ORB (Oriented FAST and Rotated BRIEF) algorithm to automatically align both images, allowing the noise (natural reflectance variations of the leaf itself) detected in the untreated control image to be subtracted from the treated image.

As a result, **EcoVid** provides a processed image that highlights the portion of the grapevine leaf affected by the product treatment and numerically displays the coverage percentage. The application offers the option to download the results in .jpg format.

The project concludes with a functional tool that offers immediate and implementable results in a real-world vineyard setting.

## Keywords

Hyperspectral imaging, precision agriculture, copper detection, computer vision, ORB algorithm, noise reduction, vineyard, Python.

---

# Índice general

---

<b>Índice general</b>	<b>iii</b>
<b>Índice de figuras</b>	<b>v</b>
<b>Índice de tablas</b>	<b>vi</b>
<b>1. Introducción</b>	<b>1</b>
1.1. Estructura de la memoria . . . . .	1
1.2. Estructura de los anexos . . . . .	2
<b>2. Objetivos del proyecto</b>	<b>3</b>
2.1. Objetivos Generales . . . . .	3
2.2. Objetivos Técnicos . . . . .	3
2.3. Objetivos Personales . . . . .	4
<b>3. Conceptos teóricos</b>	<b>7</b>
3.1. Imágenes hiperespectrales . . . . .	7
3.2. Visión por ordenador . . . . .	12
3.3. ORB(Oriented fast and Rotated BRIEF) . . . . .	12
3.4. Binarización . . . . .	14
3.5. Segmentación por Trinarización . . . . .	15
<b>4. Técnicas y herramientas</b>	<b>17</b>
4.1. Lenguaje de Programación y Entorno de Desarrollo . . . . .	17
4.2. Ecosistema de Bibliotecas para el procesamiento de imágenes	18
4.3. Framework de la Aplicación Web y Diseño . . . . .	19
4.4. Herramientas de Gestión y Apoyo al desarrollo . . . . .	20

4.5. Herramientas de IA y ayuda . . . . .	21
<b>5. Aspectos relevantes del desarrollo del proyecto</b>	<b>23</b>
5.1. Carga y Gestión de Datos Hiperespectrales: La Base del Análisis	23
5.2. Procesamiento y Generación de Máscaras de Segmentación (Trinarización) . . . . .	25
5.3. Reducción de Ruido mediante Alineación Automática por ORB	28
<b>6. Trabajos relacionados</b>	<b>37</b>
6.1. Hyperspectral images analysis . . . . .	37
6.2. UAV-Based Remote Sensing Technique to Detect Citrus Canker Disease Utilizing Hyperspectral Imaging and Machine Learning . . . . .	37
6.3. Estudio Comparativo de Técnicas de Clasificación de Imágenes Hiperespectrales . . . . .	39
<b>7. Conclusiones y Líneas de trabajo futuras</b>	<b>41</b>
7.1. Conclusiones: . . . . .	41
7.2. Líneas de trabajo futuras: . . . . .	43
<b>Bibliografía</b>	<b>45</b>

---

# Índice de figuras

---

3.1.	Representación del espectro visible por el ojo humano en comparación con el total de espectros existentes[27] . . . . .	8
3.2.	Representación visual de la información que contiene cada pixel de cada tipo de imagen. . . . .	8
3.3.	representación de una imagen hiperespectral en formato hipercubo [7] . . . . .	10
3.4.	flujo de trabajo para obtener puntos claves de imágenes mediante el algoritmo ORB . . . . .	13
3.5.	ejemplo de binarización de una manzana . . . . .	14
5.1.	El gráfico representa el histograma obtenido en la banda 10 y la imagen la máscara de la hoja obtenida de la segmentación . . .	26
5.2.	extracto del código donde se muestran los umbrales empleados para detectar cada uno de los productos . . . . .	27
5.3.	Imagen resultado de la trinarización, separando fondo (negro) de hoja (verde) de producto (rojo) . . . . .	28
5.4.	imagen resultado de la trinarización de la imagen de control (Izquierda) y la imagen con tratamiento (Derecha) . . . .	30
5.5.	imagen que muestra el resultado de la aplicación del algoritmo ORB y RANSAC par a la alineación y superposición de imágenes	32
5.6.	imagen donde se aprecia el resultado final sin ruido así como el porcentaje de recubrimiento calculado . . . . .	33
5.7.	imagen donde se muestra el resultado una vez aplicada la reducción de ruido . . . . .	34
6.1.	Imagen que muestra y resalta la detección de la enfermedad (c) sobre las hojas . . . . .	38
6.2.	imagen donde se presenta distintos mapas de clasificación de una porción de terreno . . . . .	39

---

## **Índice de tablas**

---

---

# 1. Introducción

---

La aplicación desarrollada durante este Trabajo de fin de Grado analiza dos imágenes hiperespectrales de una misma hoja de viñedo a la que se le ha aplicado un tratamiento fungicida en espray. Las imágenes corresponden a la hoja sin el tratamiento aplicado e inmediatamente después de realizar el espray. Las imágenes hiperespectrales empleadas en este trabajo contienen información para 300 bandas, desde los 380 nm hasta los 1000 nm. El empleo de estas imágenes en el análisis del dosel de hojas de viñedo ha demostrado ser un método efectivo y no destructivo de análisis en tiempo real [32, 12].

Para el desarrollo de la aplicación se han desarrollado e integrado métodos avanzados para la alineación geométrica y superposición de imágenes, utilizando el algoritmo ORB para la detección de características morfológicas coincidentes entre las dos imágenes y RANSAC para una estimación robusta de la transformación entre ambas tomas. Así como el desarrollo de una herramienta para la eliminación del ruido espectral, con el fin de optimizar el resultado.

La aplicación es capaz de detectar las gotas de producto fungicida aplicado de forma similar al tratamiento real en viñedo y devolver una imagen de la doble binarización de la hoja de vid (trinariazada), en la que se puede distinguir la parte recubierta por el producto, además de indicar el porcentaje de recubrimiento con respecto a la superficie total de la hoja.

## 1.1. Estructura de la memoria

1. **Introducción:** Descripción del proyecto y estructura de la documentación.

2. **Objetivos del Proyecto:** Objetivos generales, técnicos y personales que se esperan cumplir en este proyecto.
3. **Conceptos Teóricos:** Explicaciones teóricas sobre los conceptos más relevantes del proyecto.
4. **Técnicas y Herramientas:** Explicación de las técnicas y herramientas que se han usado a lo largo del desarrollo del proyecto.
5. **Aspectos Relevantes del Desarrollo del Proyecto:** Puntos interesantes e importantes que han ido surgiendo a lo largo del desarrollo del proyecto.
6. **Trabajos Relacionados:** Trabajos relacionados con la temática tratada en este proyecto.
7. **Conclusiones y Líneas de Trabajo Futuras:** Las conclusiones que se han extraído de la realización del proyecto y propuestas e ideas de por donde podría seguirse desarrollando este proyecto.

## **1.2. Estructura de los anexos**

- A. **Plan de Proyecto Software:**
- B. **Especificación de Requisitos:**
- C. **Especificación de Diseño:**
- D. **Documentación técnica de programación:**
- E. **Documentación de Usuario:**
- F. **Anexo de Sostenibilidad Curricular:**

---

## **2. Objetivos del proyecto**

---

En este apartado se detallan los objetivos que se persiguen con la realización del proyecto. Se puede distinguir entre los objetivos marcados para los requisitos del software a construir y los objetivos de carácter técnico que se plantean a la hora de llevar a la práctica el proyecto.

### **2.1. Objetivos Generales**

1. Desarrollar una aplicación, con una interfaz intuitiva, en la que el usuario pueda interactuar de forma sencilla y obtener un resultado visual y de forma inmediata de la eficacia de la aplicación de productos fungicidas, con base de cobre. La aplicación estará basada en el análisis de imágenes hiperespectrales de hojas de vid
2. Validar una metodología de análisis de imagen hiperespectral que garantice la obtención de resultados coherentes, fiables y repetibles. El fin último es que la cuantificación del recubrimiento sea aplicable en entornos de investigación y con potencial para su uso en aplicaciones reales de agricultura de precisión.

### **2.2. Objetivos Técnicos**

Para alcanzar los objetivos generales, se establecieron las siguientes metas técnicas específicas, centradas en el diseño y la implementación de la herramienta:

1. La construcción de una aplicación web interactiva en Python, utilizando la biblioteca Streamlit, que integre todo el flujo de trabajo: desde la carga de datos hiperespectrales hasta el procesamiento y la visualización final de los resultados.
2. El diseño e implementación de un pipeline de visión por computador para el procesamiento de imágenes hiperespectrales. Este pipeline debe incluir módulos para la segmentación de la imagen (separación de fondo, hoja y producto) y para la gestión eficiente de los datos del hipercubo.
3. La implementación de un sistema robusto para la eliminación de falsos positivos basado en la sustracción de ruido. Este objetivo es el núcleo técnico del proyecto e implica la alineación geométrica precisa de la imagen tratada con una imagen de control mediante el uso de los algoritmos ORB y RANSAC.
4. La habilitación de funcionalidades para la exportación de los resultados, permitiendo al usuario descargar tanto las imágenes del análisis (las máscaras de segmentación y la imagen final con la reducción de ruido) como los datos cuantitativos del porcentaje de recubrimiento.
5. El desarrollo de la solución siguiendo principios de ingeniería del software que aseguren una estructura de código clara, modular y documentada, facilitando así el mantenimiento, la reusabilidad de los componentes y la futura escalabilidad del proyecto.
6. La creación de una interfaz de usuario (UI) y una experiencia de usuario (UX) que cumplan con criterios de usabilidad, haciendo que el manejo de una herramienta técnicamente compleja sea sencillo e intuitivo.

### **2.3. Objetivos Personales**

1. Aprender el ciclo de vida completo del desarrollo de una aplicación funcional, desde el concepto hasta el despliegue, utilizando el framework **Streamlit**.
2. Mejorar en la comprensión del flujo de trabajo e implementar buenas prácticas a la hora de realizar un desarrollo de un producto informático. Así como mis habilidades de inclusión e implementación de patrones

de diseño en el desarrollo de la aplicación, así como su impacto en el rendimiento y otros aspectos.

3. Aumentar y fortalecer mi comprensión, nivel de programación y comprensión del lenguaje Python así como adquirir experiencia práctica con las bibliotecas especializadas en visión por computador, además de la investigación y descubrimiento de nuevas librerías y algoritmos útiles para el desarrollo de la aplicación.
4. Mejorar a nivel personal en un entorno profesional con sus fechas límites, requisitos a cumplimentar, trabajo científico y buenas prácticas a la hora de desarrollar un producto.



---

## **3. Conceptos teóricos**

---

La principal característica implementada durante este proyecto ha sido el cálculo del recubrimiento de una hoja rociada con producto antifúngico. Para ello se han empleado imágenes hiperespectrales sobre las cuales se han llevado a cabo una serie de análisis. Este proyecto se encuadra en una rama de la agricultura denominada agricultura de precisión. Por ello, para mejorar la comprensión general del proyecto es necesario tener conocimiento de los siguientes conceptos.

### **3.1. Imágenes hiperespectrales**

#### **Definición**

Una imagen hiperespectral [28] es una imagen que en vez de tener un único valor de intensidad para cada pixel de la imagen tendrá tantos como bandas tenga la imagen . Se puede entender este tipo de imágenes como una matriz de 3 dimensiones o un cubo donde para cada pixel existen tantos valores de reflectancia, absorción o fluorescencia como bandas han sido recogidas en la imagen hiperespectral[35].

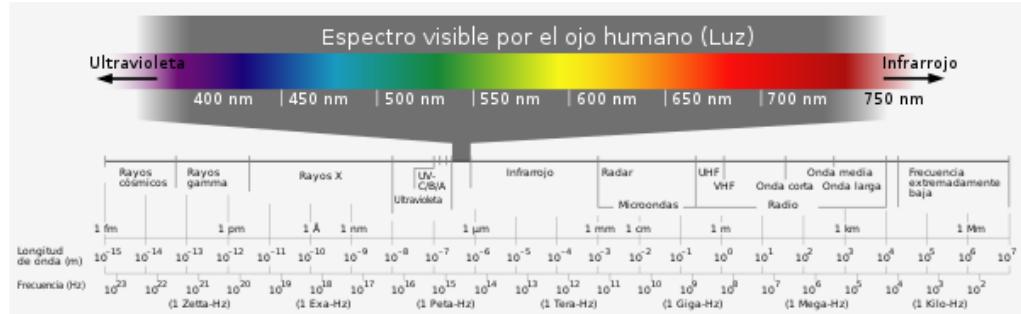


Figura 3.1: Representación del espectro visible por el ojo humano en comparación con el total de espectros existentes[27]

Es importante comprender sobre qué imagen estamos realizando el análisis. Esta podría ser monocroma, en escala de grises, sin en vez de un único color tenemos 3 colores estaríamos ante una imagen RGB. Si además tenemos diferentes longitudes de onda para cada pixel de nuestra imagen (entre 2 y 10) estaríamos ante una imagen multiespectral. Pero si los pixeles de nuestra imagen contienen datos de todo el espectro electromagnético ordenado en bandas estaríamos ante una imagen hiperespectral.

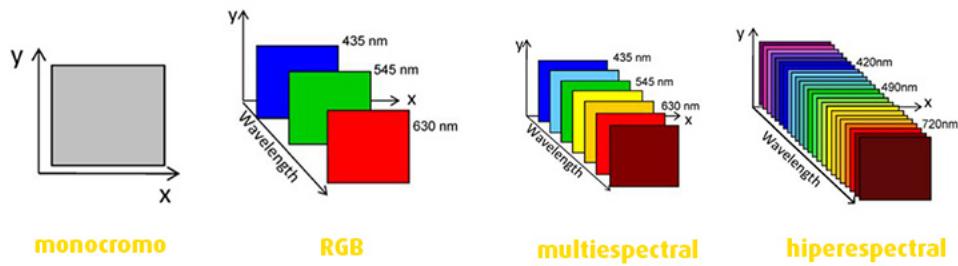


Figura 3.2: Representación visual de la información que contiene cada pixel de cada tipo de imagen.

## Adquisición de imágenes hiperespectrales

Para obtener imágenes hiperespectrales se necesita un sensor especializado capaz de captar todo el espectro de emisión del objeto o área a fotografiar.

Para ello existen diversos métodos para transportar el sensor[24] , puede ser o bien aéreo, mediante aviones o drones; o bien terrestre montando el sensor en vehículos o dispositivos terrestres.

A la hora de realizar la captura de la imagen existen diversas técnicas que se pueden aplicar:

- **Barrido instantáneo (*Snapshot*)**

Captura toda la información espectral y espacial en un único instante, sin necesidad de movimiento relativo entre el sensor y el objeto. Ideal para aplicaciones dinámicas (ej. monitoreo en tiempo real de procesos industriales o inspección de alimentos en cintas transportadoras).

- **Barrido por empuje (*Pushbroom*)**

Captura una línea espacial completa en cada instante, avanzando progresivamente sobre la escena. Es el método más usado en drones y satélites para agricultura de precisión.

- **Barrido por barrido mecánico (*Whiskbroom*)**

El sensor captura un único punto por vez, barriendo mecánicamente la escena en dos dimensiones. Menos común debido a su lentitud, pero útil en microscopía hiperespectral.

## Detalles de una imagen hiperespectral

Las imágenes con las que se trabajó durante el proyecto son imágenes hiperespectrales en formato ENVI de 300 bandas. Estas imágenes constan de 2 archivos diferentes pero complementarios.

El primero de extensión .bil (Band Interleaved by Line) se trata de un fichero binario que contiene los valores de reflectancia de cada pixel para cada una de las bandas. Como su nombre indica cada pixel corresponde con una línea del archivo donde se encuentran todos los diferentes valores de reflectancia en función de la banda.

Por otro lado el fichero con extensión .bil.hdr hace referencia a la cabecera (o header) que describe las características de la imagen y como deben

interpretarse los datos binarios contenidos en el otro archivo. En este encontramos características fundamentales como la numeración de bandas, los valores de éstas, tamaño de los píxeles, además de incluir campos referentes a los tipos de datos de cada valor de reflectancia del pixel entre otros.

Cabe destacar el gran tamaño de los ficheros con extensión .bil. Esto se debe a que al contener los datos de las 300 bandas la cantidad de datos que almacenan es muy grande. Las imágenes empleadas para el proyecto tienen unas dimensiones de 1358 x 900 x 300. En otras palabras los datos almacenados en el fichero .bil es del orden de mas de 360 millones de datos organizados en una matriz de 3 dimensiones de 1358 unidades de alto, 900 de ancho y 300 de profundidad a modo de hipercubo. Por este motivo estos ficheros pueden llegar a pesar del orden de 0,7 GB mientras que los ficheros de cabecera (.bil.hdr) tienen un tamaño mucho menor (unos 3Kb).

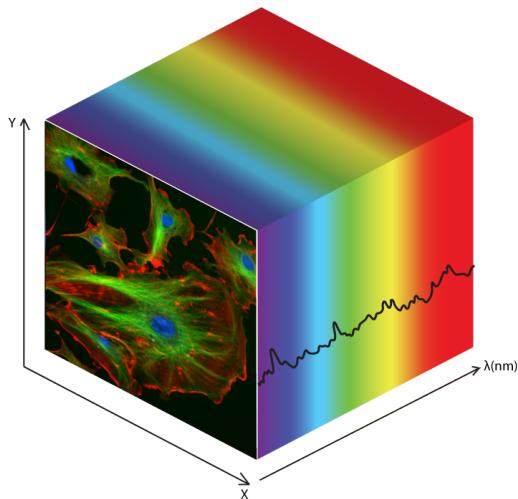


Figura 3.3: representación de una imagen hiperespectral en formato hipercubo [7]

Precisamente por esta riqueza de información, las imágenes hiperespectrales fueron seleccionadas para este proyecto, ya que permiten identificar la firma espectral única del cobre, diferenciándola de la reflectancia de la hoja. La gestión de estos ficheros de gran volumen y su representación como hipercubos mediante `Numpy` ha sido uno de los primeros desafíos técnicos abordados, siendo la base para todo el procesamiento posterior.

### Aplicaciones de las imágenes hiperespectrales:

Las imágenes hiperespectrales tienen aplicaciones muy diversas y de gran impacto en distintos campos, gracias a su capacidad para capturar información detallada tanto espacial como espectral de los objetos y materiales. Algunas de las aplicaciones más relevantes son:

### Aplicaciones principales de las imágenes hiperespectrales

- **Industria alimentaria y control de calidad:** Se utilizan para la determinación de la inocuidad y de la calidad de productos alimentarios, permitiendo detectar contaminantes, adulteraciones o defectos y para evaluar la frescura de productos hidrobiológicos como pescados, camarón y tilapia.
- **Medicina y aplicaciones biomédicas:** Permiten el análisis no invasivo de tejidos y órganos, facilitando la detección temprana de enfermedades, el estudio de la composición de tejidos y el cálculo de abundancias de diferentes componentes biológicos en imágenes médicas
- **Conservación y estudio del patrimonio cultural:** Las imágenes hiperespectrales se emplean en la caracterización espectroscópica de obras de arte y objetos históricos, permitiendo estudiar materiales, pigmentos, restauraciones previas y estados de conservación sin dañar las piezas.
- **Geología y prospección minera:** Se aplican en la prospección de minerales y recursos naturales, como la búsqueda de níquel, mediante el análisis de la composición superficial del terreno y la identificación de minerales específicos a partir de sus firmas espirituales.
- **Agricultura de precisión:** Facilitan el monitoreo de cultivos, la detección de estrés hídrico, de enfermedades, deficiencias nutricionales y la estimación de rendimientos, optimizando el uso de recursos, mejorando la productividad agrícola.
- **Monitoreo ambiental:** Se utilizan para la detección de contaminantes, el seguimiento de la calidad del agua y del aire, y la evaluación de cambios en los ecosistemas.

## Ventajas clave

- Técnicas no invasivas y no destructivas.
- Alta resolución espectral y espacial.
- Capacidad de análisis en tiempo real y sobre grandes áreas o pequeños objetos.

### 3.2. Visión por ordenador

La visión por ordenador [5]es una disciplina de la informática centrada en el desarrollo de métodos y sistemas para la interpretación y análisis de imágenes de manera automática, con el objetivo de obtener información relevante del entorno y tomar las decisiones pertinentes en base a la información recabada. Es común el uso de técnicas de procesamiento de imágenes, inteligencia artificial, aprendizaje automático entre otros para que sean capaces de comprender el contenido visual de manera similar a lo que haría el ojo humano. Algunos ejemplos de sus aplicaciones más comunes:

- Reconocimiento de objetos y patrones.
- Detección y seguimiento de movimientos.
- Diagnóstico médico asistido por imágenes (por ejemplo, detección de caries o apoyo en endoscopias)
- Inspección de calidad en procesos industriales.
- Análisis de escenas en tiempo real para vehículos autónomos.

### 3.3. ORB(Oriented fast and Rotated BRIEF)

Se trata de un algoritmo utilizado en la visión por ordenador para la detección y descripción de puntos clave en imágenes[16] empleado para la alineación automática de las imágenes permitiendo una superposición exacta de dos imágenes de la misma hoja. Obteniendo así un método automático de eliminación de ruido.

### Características principales:

- **Detección de puntos clave:** ORB utiliza el detector FAST para localizar puntos de interés en la imagen, que suelen ser esquinas o regiones con alto contraste.
- **Descripción de los puntos:** Para describir los puntos detectados, ORB emplea un descriptor basado en BRIEF, que es rápido y eficiente, pero lo mejora haciéndolo invariante a la rotación.
- **Invarianza:** ORB es invariante a rotaciones y parcialmente invariante a cambios de escala, lo que significa que puede reconocer objetos aunque estén girados o ligeramente escalados.
- **Eficiencia computacional:** El requisito fundamental para el desarrollo de una aplicación web interactiva es la velocidad de respuesta. Debido al significativamente superior rendimiento computacional, se seleccionó el algoritmo ORB frente a alternativas clásicas como SIFT o SURF. Esto permite realizar la alineación de imágenes en un tiempo aceptable para el usuario sin sacrificar la precisión necesaria.
- **Viabilidad de implementación:** A diferencia de SIFT, ORB no está sujeto a patentes, lo que garantiza que la solución desarrollada es de uso libre y puede ser fácilmente integrada, modificada o extendida en futuros trabajos académicos o comerciales sin restricciones de licencia.

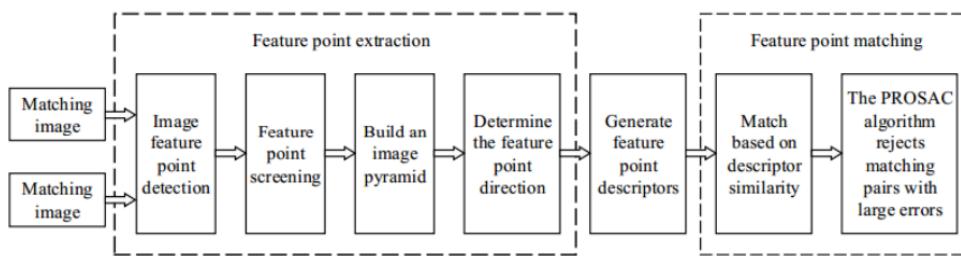


Figura 3.4: flujo de trabajo para obtener puntos claves de imágenes mediante el algoritmo ORB

### Aplicaciones habituales:

- Reconocimiento y emparejamiento de objetos.

- Seguimiento visual en robótica y realidad aumentada.
- Reconstrucción 3D y mosaico de imágenes.
- Detección de movimiento y análisis de escenas.

### 3.4. Binarización

La binarización [29] consiste en convertir una imagen que se encuentra en escala de grises o a color en solo 2 colores de tal manera que los píxeles de dicha imagen solo pueden tener 2 valores, 0 o 1 (blanco o negro). Se realiza definiendo un umbral de tal manera que todo aquello que se encuentra por debajo del umbral pertenece a uno de los dos grupos y todo lo que se encuentra por encima al otro grupo.

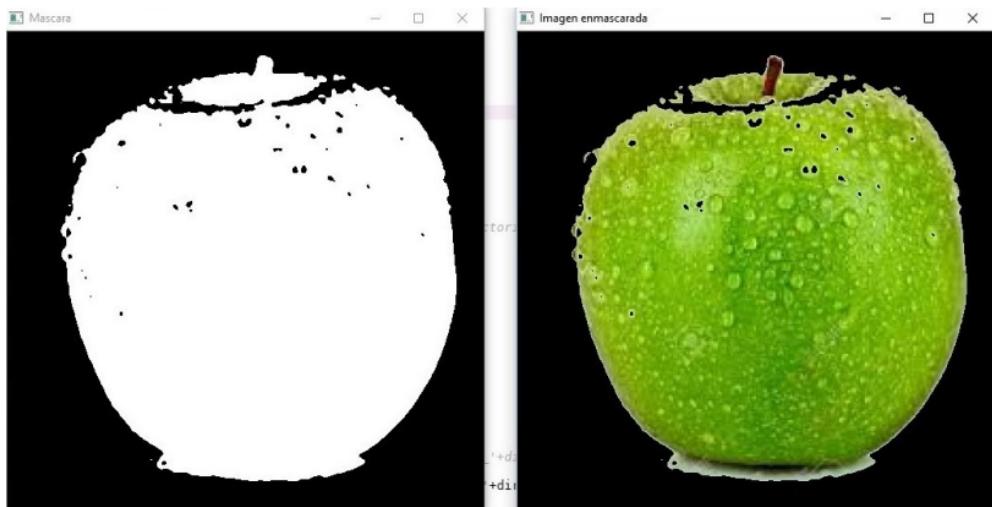


Figura 3.5: ejemplo de binarización de una manzana

En el caso específico de este proyecto la binarización ha sido empleada para realizar la discriminación entre fondo y hoja, y posteriormente se ha realizado una segunda binarización para diferenciar el producto de la hoja. A esta doble binarización se le ha decidido llamar *trinarización* ya que en esencia se están diferenciando 3 grupos (fondo, hoja y producto)

### 3.5. Segmentación por Trinarización

El problema de la identificación del producto requiere la segmentación en tres clases (fondo, hoja y producto) una binarización simple no resulta suficiente. Por ello se ha diseñado e implementado un proceso de segmentación secuencial denominado trinarización. Este método consiste en dos binarizaciones consecutivas de la siguiente forma:

- Una primera binarización con el objetivo de aislar la región de interés (la hoja) del fondo de la imagen, utilizando un umbral sobre la bandapectral 10 (406.68 nm).
- Una segunda binarización únicamente sobre la ROI (Región de interés) para poder discriminar los píxeles correspondientes a producto de cobre basándose en su alta reflectancia en la banda 164 (728,24 nm)[31].

Este enfoque permite una segmentación precisa y eficiente adaptada a las características específicas de la distribución de las imágenes.



---

## **4. Técnicas y herramientas**

---

La consecución de los objetivos de este proyecto ha requerido la selección y articulación de un conjunto de tecnologías específicas. La elección de cada herramienta no ha sido arbitraria, sino que responde a la necesidad de construir una solución robusta, eficiente y de fácil mantenimiento. En este capítulo se detalla el ecosistema tecnológico empleado, justificando el rol de cada componente dentro de la arquitectura del proyecto.

### **4.1. Lenguaje de Programación y Entorno de Desarrollo**

#### **Python**

Lenguaje de programación ampliamente utilizado, denominado de alto nivel. Destaca sobre todo por su simplicidad y versatilidad[26]. El motivo de su elección es debido a su amplia gama de bibliotecas que facilitan mucho el trabajo y la implementación de funcionalidades de manera sencilla y directa. En este caso en concreto la gran cantidad de bibliotecas ampliamente reconocidas para el análisis de imágenes y el trabajo sobre ellas mediante visión por computadora. La disponibilidad de herramientas especializadas como NumPy, OpenCV o Spectral fue un factor decisivo, ya que permitieron acelerar el desarrollo y basar la solución en tecnologías estándar de la industria y la investigación.

#### **Batchfile:**

No es un lenguaje en sí, sino un archivo que al ejecutarse escribe en consola una serie de comandos. El uso en el proyecto se ciñe a agrupar los

comandos a ejecutar para poder abrir la aplicación web del proyecto. Este tipo de archivos puede ser peligroso si no se conoce exactamente lo que hacen ya que podrían ser una fuente de posibles ciberataques.

## **VsCodeInsiders**

IDE (Entorno de Desarrollo Integrado) de programación propiedad de Microsoft elegido por delante de otras alternativas como Pycharm. Se trata de una versión de VsCode[18] que posee las últimas funcionalidades en estado de beta, de manera que permite el acceso a estas antes de que salgan para el público general. Se ha elegido en vez de otros entornos más específicos de Python como puede ser Pycharm, propiedad de JetBrains, debido a que este IDE es más eficiente y supone una menor carga para el dispositivo, sacrificando alguna funcionalidad irrelevante para el desarrollo del proyecto. Se trata de una herramienta gratuita que además incluye herramientas de control de versiones como Git. Esto facilitó un flujo de trabajo ágil y ordenado.

## **4.2. Ecosistema de Bibliotecas para el procesamiento de imágenes**

### **Numpy**

Biblioteca ampliamente utilizada sobre todo en el ecosistema científico en Python, para facilitar los cálculos numéricos eficientes empleando para ello arrays multidimensionales. En este caso en concreto su principal uso ha sido para poder trabajar de manera eficiente con los datos (hipercubo) de las imágenes pudiendo extraer los valores de reflectancia de los píxeles y trabajar con ellos de una manera más eficiente[15].

### **OpenCV**

Biblioteca para la visión por ordenador y tratamiento de imágenes. En el caso de este trabajo se ha usado para la detección de bordes de las hojas así como el algoritmo ORB que realiza la alineación de las 2 imágenes para la eliminación del ruido[6].

## Pandas

Se trata de una de las biblioteca más importantes en el ecosistema de Python. Su principal uso en este proyecto es para la estructuración de los datos obtenidos de las imágenes hiperespectrales para su procesamiento y su exportación[17].

## Pillow

Esta biblioteca se basa en la manipulación y procesamiento de imágenes en Python. Se trata de la evolución de un proyecto anterior PIL y se ha convertido en un estándar de facto para el análisis de imágenes en Python. En el caso de este proyecto, se ha usado para cargar las diferentes imágenes de la interfaz gráfica[9].

## SkImage

Otra biblioteca para el procesamiento de imágenes en Python. En este proyecto se ha usado para facilitar el flujo de trabajo desde la imagen hiperespectral hasta las imágenes normales .png resultado del análisis. Además esta biblioteca posee una gran variedad de algoritmos entre los que se encuentra el algoritmo ORB empleado para alinear las imágenes para la supresión del ruido en el análisis[33].

## Spectral

Biblioteca centrada específicamente en el análisis de imágenes hiperespectrales y multiespectrales. Permitiendo la visualización, manipulación y procesamiento de datos espectrales. En el caso concreto de este trabajo se ha usado para la carga y gestión de los hipercubos (imágenes .bil) en formato ENVI, permitiendo el acceso a otras bibliotecas como numpy a las diferentes bandas del hipercubo[19].

## 4.3. Framework de la Aplicación Web y Diseño

### Streamlit

Esta biblioteca permite la creación de páginas web interactivas de manera sencilla y rápida. Es una gran herramienta ya que permite crear las páginas

web sin conocimientos avanzados de programación web (HTML, JavaScript o CSS). Es el motor principal de la aplicación web de este proyecto sobre la que se ha implementado todo el funcionamiento y visualización de los datos[30].

### **Css (Cascade Style Sheets):**

Lenguaje de programación utilizado principalmente para la representación visual de los elementos en una página web[4]. En este caso la aplicación implementada es una aplicación web basada en los recursos de la librería Streamlit, pero que para todo el aspecto visual, colores, títulos, organización de los contenidos en la página ,etc, ha sido realizado en CSS.

## **4.4. Herramientas de Gestión y Apoyo al desarrollo**

### **Github**

Se trata de una herramienta que permite la gestión de proyectos de código principalmente, basada en el control de versiones Git[13]. La gestión a través de esta aplicación permite un modo más visual e interactivo que puede ser atractiva para usuarios que no tengan un gran dominio de Git y la linea de comandos ya que permite la gestión a través de su interfaz gráfica. Cabe destacar su gran integración con la mayoría de IDEs más conocidos y utilizados facilitando de esta manera la gestión y control de las versiones del proyecto. En este proyecto en concreto se ha empleado un único repositorio donde se aloja todo el código referente a la aplicación desarrollada. Sin embargo para al implementación de nuevas funcionalidades o pruebas alternativas se han usado distintas ramas para facilitar volver a un estado previo del proyecto si algo falla.

### **Overleaf**

Entorno web empleado para la realización de la documentación[22]. Permite la edición de archivos en formato LaTex. Ampliamente utilizado por personal de ámbito científico y técnico para la redacción y edición de papers y artículos científicos. Aunque dispone de una versión de pago que permite funcionalidades más avanzadas como sincronización con GitHub, mayor velocidad de compilación de los archivos, prioridad en alta demanda de servidores o herramientas de IA, para la realización de la memoria

del proyecto estas funcionalidades son irrelevantes por lo que se ha usado únicamente la versión gratuita.

### Spectronon

Se trata de una aplicación desarrollada por el fabricante de la cámara hiperespectral empleada en la toma de imágenes (Resonon Inc.) que permite visualizar y trabajar con las diferentes bandas contenidas en el hipercubo que se sube a la aplicación, así como poder ver los diferentes valores de reflectancia de cada pixel, o las gráficas de representación los distintos valores para las distintas bandas contenidas en el cubo. Sirve como herramienta de apoyo para poder seleccionar la banda correcta para el análisis así como los umbrales para la detección del compuesto[1].

### Docker

Se trata de una aplicación que permite crear, ejecutar y gestionar aplicaciones dentro de contenedores[11]. Estos contenedores son unidades ligeras y portátiles que contienen todo lo necesario para la ejecución de la aplicación. Su utilidad reside en la posibilidad de que una vez terminada la implementación completa de la aplicación, ésta se puede compartir y ejecutar de una manera sencilla en cualquier dispositivo que disponga de Docker instalado, sin necesidad de instalar bibliotecas o aplicaciones adicionales para ello.

## 4.5. Herramientas de IA y ayuda

### ChatGPT

Ampliamente conocida, esta plataforma de la empresa OpenAI permite el acceso a sus modelos para poder uso de ellos, a través de la aplicación de escritorio o de la versión web. Esta herramienta permite resolver dudas técnicas, investigar conceptos, aportar ideas de resolución de problemas de manera sencilla y rápida. Permite al usuario economizar esfuerzos y optimizar flujos de trabajo para mejorar la eficiencia y calidad del producto final obtenido[20]. Su principal uso en este proyecto ha sido como apoyo y ayuda a la explicación de diversos conceptos tanto teóricos como de programación así como la propuesta de posibles soluciones a los problemas encontrados durante la realización del proyecto.

## Gemini

Se trata de otra herramienta de IA. En este caso propiedad de Google, con utilidades y casos de uso similares los de ChatGPT. Destaca por su gran ventana de contexto (1 millón de tokens) permitiendo así poder analizar documentos de mayor tamaño (aproximadamente 1500 páginas) de manera precisa y con todos los detalles[14]. Durante el transcurso del trabajo se ha utilizado como alternativa a ChatGPT en casos donde esa ventana de contexto es relevante como la explicación o resumen de documentos de gran extensión.

## Perplexity

Se trata de un buscador que integra herramientas de inteligencia artificial para mejorar los resultados de las búsquedas[25]. La principal ventaja sobre otros buscadores o herramientas de IA y por la que he decidido usar esta herramienta, es que permite al usuario elegir si la búsqueda se realiza de fuentes generales (internet, wikipedia ...) o si así se desea permite que la búsqueda solo se haga a partir de fuentes meramente académicas como papers científicos o proyectos reglados incluidos en plataformas reconocidas como por ejemplo *Semantic Scholar*[3] o *Arxiv*[10]. Esta herramienta facilita mucho el trabajo de investigación de conceptos teóricos, sabiendo que la información proporcionada está contrastada y es veraz.

---

## **5. Aspectos relevantes del desarrollo del proyecto**

---

La aplicación ha sido desarrollada con el objetivo de crear un sistema de análisis de imágenes hiperespectrales que no solo sea funcional, sino también robusto, preciso y científicamente fundado. El proyecto va más allá de una simple implementación convirtiéndose en una solución de ingeniería que aborda problemas complejos relacionados con la visión por ordenador en el ámbito de la agricultura de precisión.

El sistema se ha implementado como una aplicación web interactiva, con una arquitectura modular estructurada en un flujo de trabajo lógico. Este flujo lo conforman tres fases técnicas fundamentales detalladas a continuación: la adquisición y gestión de los datos hiperespectrales; el procesamiento y segmentación para la extracción de características; y el elemento más innovador del proyecto, un sistema avanzado de reducción de ruido y falsos positivos basado en la alineación y superposición automática de imágenes.

### **5.1. Carga y Gestión de Datos Hiperespectrales: La Base del Análisis**

La base de la aplicación sobre la que se fundamenta el análisis es la correcta obtención y estructuración de los datos. La fiabilidad de los resultados del análisis depende en su totalidad de esta fase inicial. El sistema está diseñado para procesar imágenes en formato ENVI, un estándar en el campo de la teledetección, que encapsula la riqueza de la información hiperespectral en dos ficheros:

- Fichero de datos (**.bil**): Un archivo binario que contiene la matriz de valores de reflectancia. Dada la alta resolución espacial ypectral de las imágenes manejadas (con dimensiones del orden de  $1358 \times 900 \times 300$  píxeles), estos ficheros son de gran tamaño (aproximadamente 0,7 GB), almacenando más de 360 millones de puntos de datos individuales. Se trata del hipercubo de datos.
- Fichero de cabecera (**.hdr**): Un archivo de texto que describe los metadatos de la imagen, como las dimensiones, el número de bandas (300 en este caso), las longitudes de onda asociadas a cada una y el formato de los datos, así como datos de la cámara con la que se ha realizado la captura entre otros metadatos útiles para identificar la imagen y sus características. Es esencial para la correcta interpretación del hipercubo de datos.

Para gestionar estos ficheros se han empleado una selección cuidadosa de bibliotecas del ecosistema científico de Python con el objetivo garantizando eficiencia y precisión:

1. **Interfaz de Usuario con Streamlit:** Para la construcción de la interfaz gráfica se eligió el framework Streamlit. Gracias a esta decisión estratégica el ciclo de desarrollo fue rápido y eficiente. En lugar de invertir un tiempo considerable en el desarrollo web tradicional (HTML, CSS, JavaScript y un backend como Flask o Django), Streamlit permite crear aplicaciones interactivas y centradas en los datos con código Python. Esto liberó recursos para centrarse en el núcleo del problema: los algoritmos de procesamiento de imágenes.
2. **Lectura de Datos con Spectral (SPy):** Para la interpretación de los datos hiperespectrales se seleccionó la biblioteca **Spectral**, el estándar de facto en el ámbito académico para esta tarea. Su elección se justifica por su capacidad única para parsear correctamente los metadatos de los ficheros **.hdr** y mapear los complejos datos binarios del fichero **.bil** en una estructura coherente y accesible. Sin esta biblioteca, sería necesario implementar un lector de bajo nivel, una tarea compleja y propensa a errores.
3. **Representación Numérica con Numpy:** El uso de NumPy es crucial permitiendo realizar una representación de los datos. Una vez leídos los datos, se materializan como un array tridimensional de NumPy. La elección de esta biblioteca se debe a su altísimo rendimiento

## 5.2. PROCESAMIENTO Y GENERACIÓN DE MÁSCARAS DE SEGMENTACIÓN (TRINARIZACIÓN)

25

en operaciones vectorizadas, crucial para manipular los más de 300 millones de puntos de datos de cada hipercubo de forma eficiente y mantener así la interactividad de la aplicación. Esta representación posibilita la posterior realización operaciones directamente con los datos espectrales obtenidos.

Además, el módulo `src/funciones/archivos.py` implementa un sistema de guardado que asigna un identificador único (UUID) a cada fichero subido, evitando colisiones de nombres y gestionando de forma segura los archivos temporales en el servidor.

### 5.2. Procesamiento y Generación de Máscaras de Segmentación (Trinarización)

Una vez el hipercubo de datos está cargado en memoria, la siguiente fase consiste en la segmentación de la imagen. El objetivo es traducir la informaciónpectral obtenida en información semántica, asignando cada píxel a una de las tres categorías: fondo, superficie foliar (hoja) o producto de cobre. Este proceso, el cual se ha denominado *trinarización*, se fundamenta en un método de segmentación por umbral por bandas espectrales, cuyos parámetros fueron definidos empíricamente a partir del análisis de las firmas espectrales del material [31].

El flujo de segmentación, detallado en `src/funciones/procesamiento.py`, es el siguiente:

1. **Segmentación Primaria (Hoja vs. Fondo):** El primer paso a realizar es aislar la hoja del fondo. Se identificó que la **banda 10** del espectro ofrecía un contraste sólido para esta tarea. Después de descartar varias configuraciones, se optó por aplicar un umbral en el que se identifican como hoja todos aquellos píxeles cuyo valor de reflectancia spectral para esa banda es menor de 2000 ( $b10 < 2000$ ), generando una máscara binaria inicial que representa la silueta de la hoja.

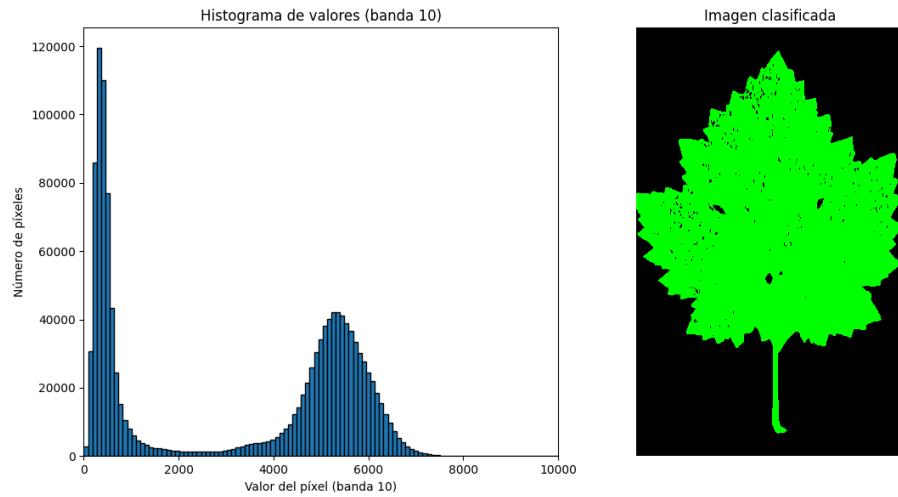


Figura 5.1: El gráfico representa el histograma obtenido en la banda 10 y la imagen la máscara de la hoja obtenida de la segmentación

2. **Refinamiento Morfológico de la Máscara Foliar:** Este método de segmentación por umbral simple a menudo genera máscaras que no son exactas y contienen ruido. Con el objetivo de reducir esta característica no deseada se optó por utilizar operaciones de morfología matemática de la biblioteca Scikit-image (*remove\_small\_holes*) en lugar de filtros de suavizado convencionales (como un filtro Gaussiano). Esta elección se fundamenta en que los filtros de suavizado, si bien reducen el ruido, difuminan los bordes de la imagen. Este factor hace que no sean adecuados debido a que para el posterior proceso de alineación, es fundamental preservar la nitidez de los contornos de la hoja. Por otro lado las operaciones morfológicas, permiten eliminar el ruido (como agujeros o pequeños objetos aislados) respetando la integridad de los bordes de las formas principales.
3. **Segmentación Secundaria (Producto vs. Hoja):** Una vez aislada la región de la hoja (máscara *hoja*), se procede a detectar el producto de cobre. Se identificó que la **banda 164 (728,24 nm)** como la más sensible a la reflectancia de los productos con base de cobre[31]. Sobre esta banda, se aplica un segundo conjunto de umbrales para identificar los píxeles correspondientes a las deposiciones del producto dentro del área foliar. Para asegurar una correcta identificación del producto sobre la hoja esta segunda segmentación solo se realiza sobre aquellos píxeles que conforman la máscara de la hoja. Estos umbrales empleados

## 5.2. PROCESAMIENTO Y GENERACIÓN DE MÁSCARAS DE SEGMENTACIÓN (TRINARIZACIÓN)

27

para la detección del producto son umbrales compuestos permitiendo así una mayor precisión en la detección eliminando parte del ruido generado. Obteniendo de esta manera la máscara *producto*.

```
#cuprocol
drops = ((b164 >= 3900) & (b164 <= 4100)) | ((b164 >= 4900) & (b164 <= 5200))

#cuporantol duo
drops = ((b164 >= 3900) & (b164 <= 4300)) | ((b164 >= 4900) & (b164 <= 5200))
```

Figura 5.2: extracto del código donde se muestran los umbrales empleados para detectar cada uno de los productos

Para cada uno de los dos productos se identificaron distintos rangos de lectura, dentro de los parámetros estadísticos determinados experimentalmente para cada uno de ellos[31]. Para la determinación de Cuprocol se empleó la combinación de rangos: 3900 nm – 4100 nm y 4900 nm – 5200 nm. Para la determinación del compuesto Cuprantol Duo la combinación de rangos usada fue: 3900 nm – 4300 nm y 4900 nm – 5200 nm.

4. **Generación de la Imagen Trinarizada:** Finalmente, las máscaras de *hoja* y *producto* se combinan para crear una única imagen de visualización. En esta imagen, a cada clase se le asigna un color distintivo para facilitar la inspección visual: el fondo se representa en negro, la hoja en verde y el producto detectado en rojo.

Este método de segmentación por umbrales, ha demostrado ser altamente efectivo y eficiente en términos de computación cuando las bandas y los umbrales se han calibrado correctamente, como es el caso de este proyecto.

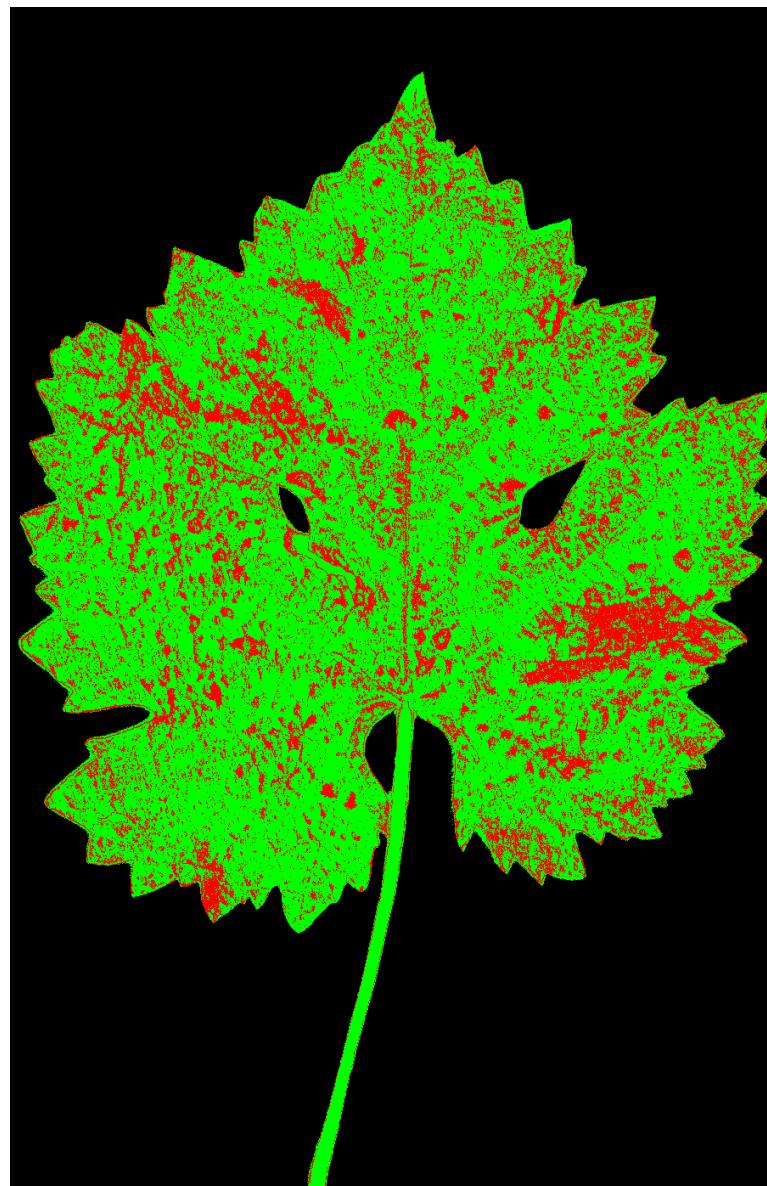


Figura 5.3: Imagen resultado de la trinarización, separando fondo (negro) de hoja (verde) de producto (rojo)

### **5.3. Reducción de Ruido mediante Alineación Automática por ORB**

El reto más complejo del presente trabajo no ha sido la detección del producto en sí, sino la eliminación de falsos positivos. Al realizar la trinarización,

la *imagen resultado* obtenida presenta detecciones erróneas a las cuales se ha referido como ruido. El motivo de la aparición del ruido en la detección del producto se debe a que los valores espectrales de los píxeles que conforman la superficie de una hoja no son uniformes, sino que presentan variaciones naturales de reflectancia debidas a su estructura (nervios, micro-vellosidades, brillos especulares, zonas sombreadas, etc) que pueden ser erróneamente clasificadas como producto por el algoritmo ya que dichos valores podrían estar dentro de los umbrales de detección de producto.

La solución implementada para solucionar la interferencia de ruido espectral es la contribución técnica más significativa de este trabajo: un sistema de sustracción de ruido que utiliza una imagen de control (sin tratamiento) para identificar y eliminar estas variaciones espectrales naturales de la imagen tratada. Para que esta sustracción sea válida, las dos imágenes deben estar perfectamente alineadas a nivel sub-píxel, una tarea no trivial dado que es imposible volver a colocar la hoja exactamente en la misma posición y orientación. El correcto funcionamiento de este sistema de reducción de ruido se fundamenta en el hecho de que las variaciones en la superficie de la hoja que fomentan la aparición de los falsos positivos en la detección son coincidentes en ambas imágenes de la misma hoja antes y después de aplicar el producto permitiendo de esta manera una identificación clara y precisa de aquellos píxeles que conforman las anomalías.

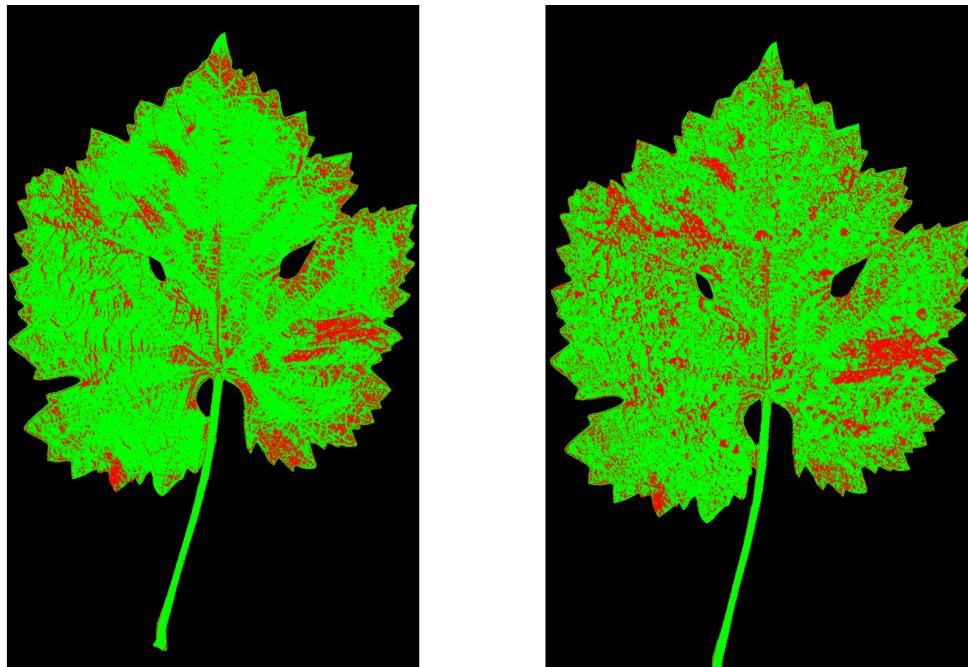


Figura 5.4: imagen resultado de la trinarización de la imagen de control (Izquierda) y la imagen con tratamiento (Derecha)

### El Proceso de Alineación Automática: de ORB a RANSAC

Se diseñó e implementó un pipeline de alineación automática utilizando la biblioteca OpenCV, como se detalla en `src/funciones/alignment.py`. La elección de cada componente de este pipeline fue una decisión de ingeniería deliberada.

1. **Extracción del Limbo Foliar:** Se aísla la parte más estable de la hoja, el limbo, eliminando el pecíolo (`_remove_petiole`) mediante operaciones morfológicas (identificación de objetos similares al pecíolo, una estructura delgada y de longitud considerable) ya que es irrelevante para el análisis y dificultaba la superposición de las imágenes.
2. **Detección de Bordes con Canny:** Se aplica el detector de bordes de Canny[21] a ambas máscaras del limbo. Se eligió Canny por ser un algoritmo multi-etapa (suavizado, cálculo de gradiente, supresión de no máximos y umbralización por histéresis) que produce contornos de un solo píxel de grosor, limpios y bien definidos. Estas características hacen que el resultado de aplicar el algoritmo sea ideal para poder aplicar el detector de características ORB. Ya que dicho detector opera

de manera más eficaz y precisa sobre regiones de alto contraste bien localizadas.

3. **Detección, Descripción y Emparejamiento de Puntos Clave con ORB:** El algoritmo ORB analiza los bordes y extrae cientos de puntos de interés (*keypoints*). Para cada punto, genera un descriptor binario que captura la textura local identificando así las características únicas de ese punto creando de esta manera un identificador único para cada uno de los puntos extraídos. A continuación, un **BFMatcher** (*Brute-Force Matcher*) compara estos descriptores para encontrar correspondencias entre las dos imágenes. Si bien la implementación de un método de fuerza bruta puede parecer prohibitivo en términos computacionales es extremadamente eficiente para los descriptores binarios de ORB [8]. Por lo tanto el BFMatcher garantiza encontrar la correspondencia óptima, debido a que es exhaustivo (compara todos los puntos extraídos de la imagen base con todos los puntos extraídos de la imagen con producto).
4. **Estimación Robusta de la Transformación con RANSAC:** El emparejamiento por fuerza bruta puede producir errores (*outliers*). La inclusión de RANSAC[34] es indispensable para la robustez del sistema. Este método estadístico iterativo aísla el conjunto de correspondencias que son geométricamente consistentes (*inliers*) y calcula la matriz de transformación afín (serie de transformaciones a realizar sobre la imagen de control para obtener la coincidencia de ambas imágenes) basándose únicamente en ellas descartando los posibles errores. Sin RANSAC, unos pocos emparejamientos erróneos podrían desviar por completo la transformación, invalidando la alineación.
5. **Aplicación de la Transformación:** Finalmente, la matriz de transformación calculada se aplica a la imagen de control y su máscara de ruido, alineándola con precisión sobre la imagen tratada, lista para la fase de sustracción.

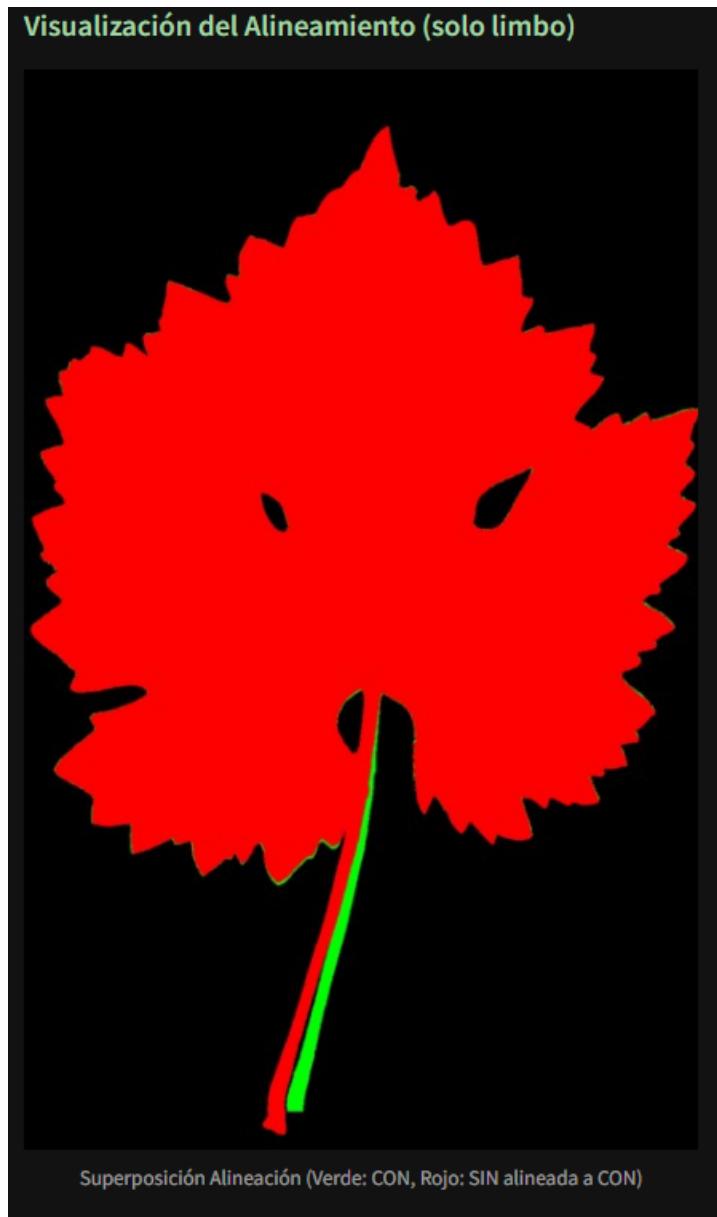


Figura 5.5: imagen que muestra el resultado de la aplicación del algoritmo ORB y RANSAC para la alineación y superposición de imágenes

6. **Sustracción del ruido:** Una vez ambas máscaras han sido alineadas geométricamente se procede a la fase final de sustracción con el objetivo de aislar las detecciones válidas de producto. Para ello se realizan una serie de operaciones lógicas a nivel de píxel:

- a) Definición del área común de análisis: se calcula una nueva máscara *hoja\_común* como el resultado de la intersección lógica de las máscaras de la imagen de control y la imagen con producto una vez alineadas. Esta nueva máscara representa exactamente el área de la hoja visible y válida entre ambas imágenes. De esta manera se asegura que la comparación solo se realice sobre la región de interés compartida.
- b) Sustracción de ruido: posteriormente se comparan las detecciones de ambas máscaras alineadas (imagen de control e imagen con producto). Se analizan los píxeles que conforman la región del área común uno a uno. Se considerarán como producto final, si, y solo si, cumplen dos condiciones: deben de ser identificados como producto en la imagen con gotas (producto) y NO fue detectado como tal en la imagen sin gotas (imagen de control) eliminando así todas aquellas detecciones coincidentes en ambas imágenes. Debido a que el ruido es común en ambas imágenes.
- c) Post-procesado: finalmente, se calcula el porcentaje de recubrimiento dividiendo el número de píxeles de producto final entre el número total de píxeles que conforman la máscara de la hoja inicial (*hoja\_común*).

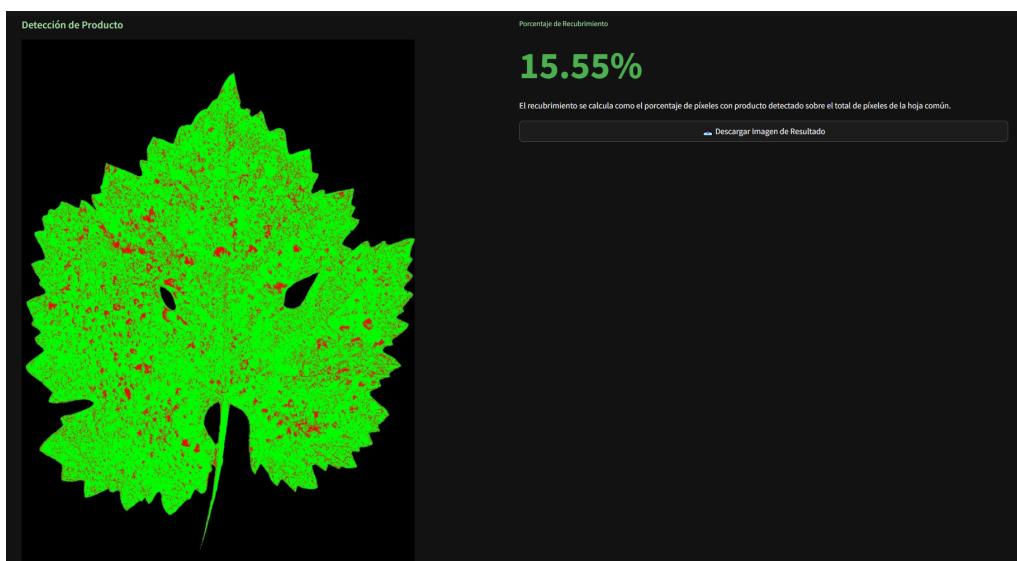


Figura 5.6: imagen donde se aprecia el resultado final sin ruido así como el porcentaje de recubrimiento calculado

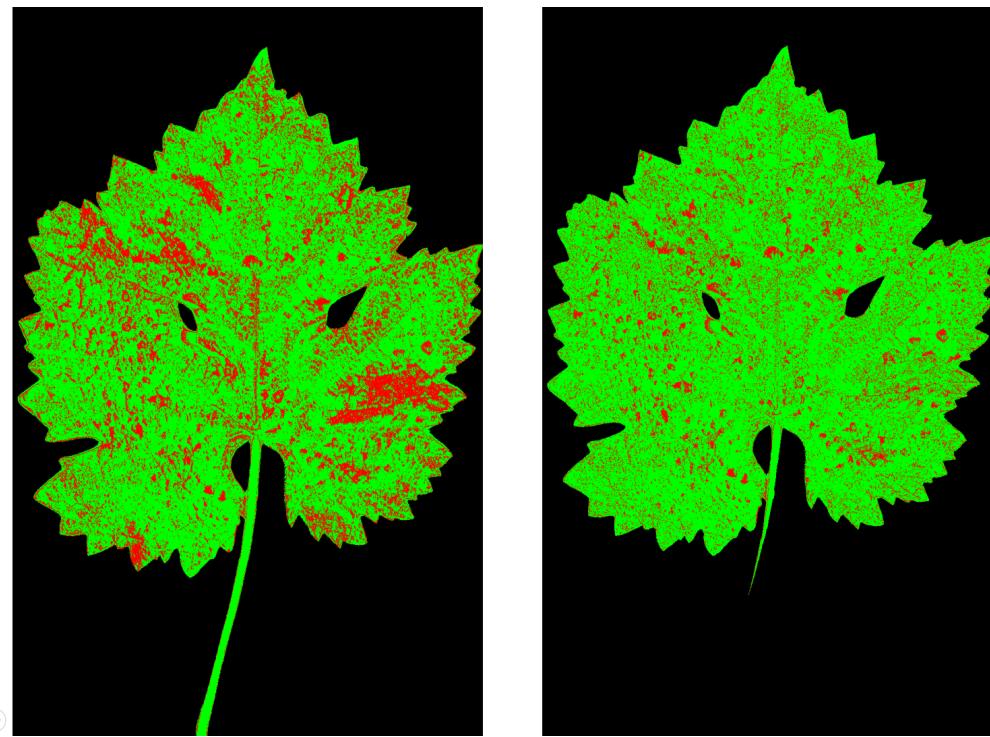


Figura 5.7: imagen donde se muestra el resultado una vez aplicada la reducción de ruido

**Justificación Técnica de la Elección de ORB** La elección de ORB frente a otros algoritmos de detección de características bien conocidos como SIFT, SURF o AKAZE no fue casual, sino una decisión de ingeniería basada en un análisis de coste-beneficio adaptado a los requisitos del proyecto:

- **Eficiencia Computacional:** Este es el aspecto central que fundamenta su selección frente a las alternativas. ORB es órdenes de magnitud más rápido que SIFT y SURF. Mientras que SIFT/SURF requieren complejas operaciones en coma flotante, ORB se basa en descriptores binarios cuya comparación mediante distancias de Hamming es extremadamente eficiente. Para una aplicación interactiva que maneja imágenes de gran tamaño (más de un millón de píxeles), esta velocidad es crucial para garantizar tiempos de respuesta aceptables para el usuario sin necesidad de hardware especializado (GPU).
- **Licencia y Disponibilidad:** Históricamente, SIFT y SURF estaban sujetos a patentes que limitaban su uso en muchos entornos. ORB, en cambio, fue diseñado desde su origen como una alternativa libre

y de código abierto. Su integración nativa en el módulo principal de OpenCV facilita enormemente su implementación, evitando las complicaciones de compilación o licenciamiento que otros algoritmos presentaban.

- **Robustez Adecuada para el Problema:** Si bien SIFT es a menudo considerado el estándar de facto en precisión bajo condiciones extremas, para este caso de uso específico (alinear dos imágenes casi idénticas de la misma hoja tomadas con mínimos cambios de perspectiva, iluminación o posición) la robustez de ORB es más que suficiente. ORB es explícitamente invariante a la rotación y tolera bien cambios moderados de escala y brillo. En este escenario, la ganancia marginal de precisión que podría ofrecer un algoritmo más pesado no compensa la drástica pérdida de rendimiento. ORB ofrece el equilibrio óptimo entre robustez y velocidad.
- **Idoneidad para Flujos Automatizados:** ORB es fácil de parametrizar y funciona de manera estable en una amplia variedad de imágenes sin necesidad de ajustes finos, lo que lo hace ideal para el pipeline totalmente automático que se pretendía construir.

En definitiva, la sustracción de ruido mediante la alineación automática con ORB y RANSAC no solo resuelve el principal desafío técnico del proyecto, sino que lo hace de una manera elegante, robusta y científicamente rigurosa, cumpliendo con éxito los objetivos más exigentes del proyecto.



---

## **6. Trabajos relacionados**

---

A continuación se van a presentar otras tesis o trabajos que utilizan herramientas similares a las empleadas en el desarrollo de este proyecto así como algunos que comparten objetivos comunes o similares.

### **6.1. Hyperspectral images analysis**

Este artículo describe [27] de manera general que son las imágenes hiperespectrales, exponiendo conceptos teóricos como su definición o posibles aplicaciones además de tratar los diferentes métodos de obtención y procesamiento.

Sirve como punto de partida de muchos de los temas descritos a lo largo de esta memoria como las diferentes definiciones de los componentes de una imagen hiperespectral. También realiza una gran descripción de los diferentes métodos que existen para la obtención de estas imágenes así como sus posibles aplicaciones.

### **6.2. UAV-Based Remote Sensing Technique to Detect Citrus Canker Disease Utilizing Hyperspectral Imaging and Machine Learning**

En el trabajo de Abdulridha et al. [2], se desarrolló y evaluó una técnica de teledetección para la detección del chancre de los cítricos, una enfermedad causada por la bacteria *Xanthomonas citri* subsp. *citri*. El objetivo principal

fue detectar la enfermedad en hojas y frutos inmaduros de la variedad 'Sugar Belle' en diferentes estadios (asintomático, temprano y tardío), tanto en condiciones de laboratorio como en campo mediante un vehículo aéreo no tripulado (UAV).

Para ello, se utilizó un sistema de imagen hiperespectral en el rango de 400-1000 nm. Los datos espectrales se analizaron comparando dos métodos de clasificación de aprendizaje automático: la Red Neuronal de Función de Base Radial (RBF) y el K-Nearest Neighbor (KNN). Además, se evaluó la eficacia de 31 índices de vegetación para identificar la enfermedad.



Figura 6.1: Imagen que muestra y resalta la detección de la enfermedad (c) sobre las hojas

Los resultados demostraron una alta capacidad de detección en hojas. En laboratorio, se alcanzó una precisión de clasificación de hasta el 96 % con el método RBF y 94 % con KNN para diferenciar hojas sanas de hojas en estado asintomático. Sin embargo, la detección temprana en frutos inmaduros no resultó fiable, obteniendo una precisión baja (47 % con RBF) en la fase asintomática. A pesar de ello, la técnica sí logró identificar los frutos en la fase tardía de la enfermedad con un 92 % de precisión.

### 6.3. Estudio Comparativo de Técnicas de Clasificación de Imágenes Hiperespectrales

Una de las áreas más exploradas en el tratamiento de imágenes hiperespectrales es la asignación a cada pixel con una etiqueta correspondiente a una clase predefinida (por ejemplo, minerales, tipos de cultivo o zonas urbanas). A esto se le denomina clasificación supervisada.

En el *Estudio Comparativo de Técnicas de Clasificación de Imágenes Hiperespectrales* de Paoletti et al. [23], se realiza una revisión exhaustiva de los algoritmos más comunes para esta tarea. En el artículo se compara el rendimiento de distintos métodos como las máquinas de Vectores de Soporte (SVM), Random Forest (RF) y diversas arquitecturas de redes neuronales (MLP, CNN). Se destaca que, si bien todos los métodos expuestos son viables y muy potentes para la caracterización de materiales con firmas espectrales distintas su resultado depende en gran medida de los datos disponibles para su entrenamiento entre otros factores.

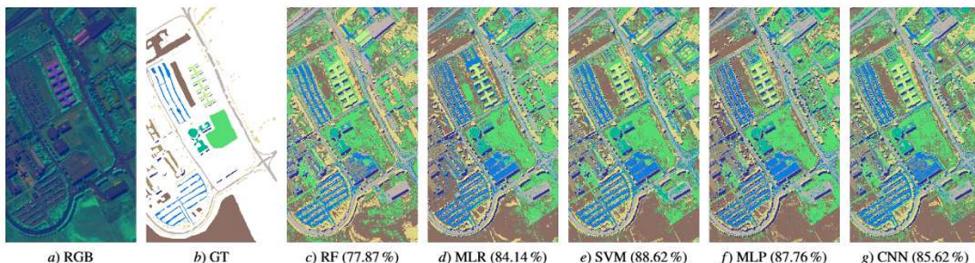


Figura 6.2: imagen donde se presenta distintos mapas de clasificación de una porción de terreno

Este enfoque si bien es fundamental en la teledetección propone una alternativa diferente a la aplicada en el desarrollo del presente proyecto. La clasificación Supervisada busca diferenciar entre clases heterogéneas (ej. soja vs. maíz) donde ambas clases son sustancialmente diferentes entre sí, nuestro proyecto se enfrenta a un reto de detección de cambios sutiles en vez de la identificación de clases planteadas en el artículo. Por este motivo se optó por una metodología diferencial y geométrica basada en la alineación de imágenes que no requiere entrenamiento previo en vez de un clasificador estadístico.



---

## **7. Conclusiones y Líneas de trabajo futuras**

---

Una vez finalizado el desarrollo del proyecto, se puede afirmar a modo de conclusión que se han cumplido todos los objetivos impuesto tanto al inicio del mismo como durante las diferentes fases del desarrollo. Este capítulo presenta una síntesis de los logros alcanzados así como las contribuciones técnicas más significativas y una reflexión sobre los desafíos superados. A modo de conclusión se proponen las líneas de trabajo futuras que se abren a partir de los resultados obtenidos.

### **7.1. Conclusiones:**

A lo largo del desarrollo del proyecto se han ido cumpliendo con éxito los objetivos planteados, culminando en el desarrollo de una solución de software funcional, robusta y válida, a modo de aplicación web interactiva para el análisis de la eficiencia de tratamientos fitosanitarios en viñedos. Dicha aplicación posee la capacidad de cuantificar de manera precisa y repetible el porcentaje de recubrimiento de productos con base de cobre sobre hojas de vid, abordando un problema relevante en el campo de la agricultura de precisión.

Como principal contribución del proyecto destaca la validación de una metodología completa desde la obtención y gestión de datos hiperespectrales hasta la reducción de ruido mediante un enfoque de visión por computador. Se ha demostrado que la estrategia de emplear una imagen de control para obtener las variaciones de reflectancia inherentes a la hoja implementada

para la reducción de falsos positivos es un método eficaz que mejora de manera considerable los resultados obtenidos.

La aplicación desarrollada no solo proporciona como resultado un valor numérico sino que ofrece una interfaz intuitiva y la posibilidad de visualizar de manera clara los resultados a modo de imagen trinarizada, consolidándose de esta manera como una base sólida con aplicabilidad en entornos de investigación y con un claro potencial de evolución hacia soluciones de campo.

El desarrollo de la aplicación ha implicado la superación de desafíos técnicos significativos cuyas soluciones constituyen el núcleo de la ingeniería de este proyecto:

1. El principal obstáculo fue la alta variabilidad espectral de la superficie de la hoja. Generando un número considerable de falsos positivos, confundiendo el producto aplicado con la textura natural de la hoja. Para solventarlo se decidió implementar un enfoque diferencial, utilizando para ello una hoja sin tratamiento como imagen de control sobre la que identificar los patrones y las formaciones de la hoja que creaban los falsos positivos. Este enfoque permite medir y sustraer el ruido específico de cada muestra ofreciendo una precisión difícilmente alcanzable con filtros genéricos.
2. Si bien el enfoque de la estrategia diferencial es acertado, solo es viable en el caso de que la superposición de la imagen de control y la imagen con tratamiento es perfecta. Para superar este obstáculo fue diseñado un pipeline de alineación totalmente automático empleando la detección de bordes mediante el algoritmo Canny, el emparejamiento de características con ORB y la estimación de la matriz de transformación afín mediante RANSAC. Esta implementación demostró ser clave para garantizar una alineación a nivel sub-pixel, precisa y repetible, convirtiendo de esta manera un proceso manual inviable en una operación automática, fiable y repetible.

En términos personales gracias al desarrollo de este proyecto he adquirido conocimientos sobre los procesos y las técnicas para desarrollar una solución de ingeniería a un problema real, desde la identificación del problema, hasta la construcción de un producto que cumpla los requisitos necesarios para ser aplicado en el mundo real. Durante este proceso no solo he evolucionado como desarrollador sino que me ha aportado una visión crítica del trabajo a realizar siendo capaz de identificar posibles problemas antes de que lleguen

a ocurrir convirtiéndome en un desarrollador más completo y crítico. Otro aspecto esencial adquirido durante el proyecto ha sido la comunicación precisa y correcta de las dificultades encontradas durante el transcurso del proyecto con los tutores, dando indicaciones concretas y fundamentadas de los problemas originados durante el proyecto, facilitando así su posterior subsanación.

## 7.2. Líneas de trabajo futuras:

El sistema desarrollado si bien es una solución completa para el problema planteado, establece las bases para futuras investigaciones y mejoras que podrían resultar en una amplificación significativa del impacto y alcance de la aplicación.

Actualmente la aplicación esta optimizada para compuestos con base de cobre. Una evolución natural del proyecto seria extender esta funcionalidad a la detección de más compuestos convirtiendo así el sistema en una plataforma más flexible. Esto implicaría el desarrollo de una interfaz de usuario avanzada que permita modificar al usuario parámetros clave del análisis como la banda espectral sobre la que realizar el análisis o modificar los umbrales permitiendo una mayor adaptabilidad a las necesidades de detección y análisis de otro tipos de tratamientos, cultivos o condiciones de captura, llegando a poder detectar múltiples compuestos de manera simultánea y pudiendo analizar la totalidad química de un tratamiento sin necesidad de centrarse en un compuesto en específico.

El objetivo final de un proyecto de estas características es la implementación de la tecnología desarrollada en la maquinaria de cultivo. Por ello es esencial que el análisis se pueda realizar en tiempo real, en vez de sobre imágenes estáticas de laboratorio. Para ello sería necesario una serie de optimizaciones y mejoras sobre los algoritmos implementados con el objetivo de permitir el análisis sobre secuencias de video. Como consecuencia indirecta se obtendría un mayor alcance de la aplicación abriendo la puerta a su integración en sistemas montados en vehículos agrícolas o drones, obteniendo un mayor alcance del proyecto.

La aproximación implementada para la detección de ruido es efectiva y replicable, sin embargo, posee el inconveniente de que es necesario para ello una imagen de control. Esta imagen no es viable o no se puede obtener como en los casos de detección en tiempo real. Por ello la implementación de modelos de Aprendizaje Profundo que ayuden identificar o detectar la morfología de la superficie correspondiente al producto sería una solución

robusta y viable, que subsanaría el problema de la ausencia de una imagen de control. Para ello se debería crear y etiquetar un *dataset* con un gran número de muestras representativas para la implementación y entrenamiento de un modelo que ayude a descartar el ruido y falsos positivos creados por las variaciones de reflectancia intrínsecas de la superficie de la hoja además de mejorar la aplicabilidad del sistema a distintas condiciones más adversas o variables.

---

## Bibliografía

---

- [1] Resonon - hyperspectral imaging systems, 2025.
- [2] Jaafar Abdulridha, Ozgur Batuman, and Yiannis Ampatzidis. Uav-based remote sensing technique to detect citrus canker disease utilizing hyperspectral imaging and machine learning. *Remote Sensing*, 11(11), 2019.
- [3] Allen Institute for AI. Semantic scholar, 2024.
- [4] Miguel Angel Álvarez, Sara Alvarez, Jhon Nadie, David Rousset, Jorge Vargas Vega, Jaime Peña Tresancos, and Juan R Castro Lurita. Manual de css 3, 2017.
- [5] Serpa Andrade and Luís Javier. Propuesta de un método basado en visión por computadora como herramienta de apoyo en el diagnóstico endoscópico. 2014.
- [6] Gary Bradski. The OpenCV library. *Dr. Dobb's Journal of Software Tools*, 25:120–125, 2000.
- [7] Rodolfo José Piedra Camacho. Aproximación inicial a la comparación de cámaras hiperespectrales para su aplicación en agricultura. *Tecnología en marcha*, 33(Mov. 7):82–91, 2020.
- [8] Qinhan Chen, Lijian Yao, Lijun Xu, Yankun Yang, Taotao Xu, Yuncong Yang, and Yu Liu. Horticultural image feature matching algorithm based on improved orb and lk optical flow. *Remote Sensing*, 14(18), 2022.
- [9] Alex Clark and Contributors. Pillow (the friendly pil fork), 2010. Versión 10.4.0.

- [10] Cornell University. arxiv.org e-print archive, 2024.
- [11] Docker, Inc. *Docker Documentation*, 2025.
- [12] S. F. Di Gennaro, P. Toscano, M. Gatti, S. Poni, A. Berton, and A. Matese. Spectral comparison of uav-based hyper and multispectral cameras for precision viticulture. *Remote Sensing*, 14(3), feb 2022.
- [13] GitHub, Inc. Github: Where the world builds software, 2025.
- [14] Google. Gemini, 2025. Se recomienda especificar la versión del modelo y la fecha de la conversación si se cita una respuesta específica.
- [15] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, et al. Array programming with NumPy. *Nature*, 585:357–362, 2020.
- [16] Chuan Luo, Wei Yang, Panling Huang, and Jun Zhou. Overview of image matching based on orb algorithm. 1237(3):032020, jun 2019.
- [17] Wes McKinney. Data structures for statistical computing in python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56–61, 2010.
- [18] Microsoft. Visual studio code, 2025.
- [19] Thomas V. O’Connor. Spectral python (spy), 2024. Versión 0.23.1.
- [20] OpenAI. Chatgpt, 2025. Se recomienda especificar la versión del modelo (p. ej., GPT-4) y la fecha de la conversación si se cita una respuesta específica.
- [21] OpenCV team. Canny edge detection. In *OpenCV-Python Tutorials*. 2025. Documentación para la versión 4.x.
- [22] Overleaf. Overleaf, editor de latex online, 2025.
- [23] Mercedes Eugenia Paoletti, Juan Mario Haut, Javier Plaza, and Antonio Plaza. Estudio comparativo de técnicas de clasificación de imágenes hiperespectrales. *Revista Iberoamericana de Automática e Informática industrial*, 16(2):129–137, mar. 2019.
- [24] José Manuel Pérez. Computación de altas prestaciones aplicadas a la detección de anomalías en imágenes hiperespectrales. high performance computing applied to anomaly detection on hyperspectral images. 2018.
- [25] Perplexity AI, Inc. Perplexity, 2025.

- [26] Python Software Foundation. *The Python Language Reference, Version 3.11*, 2023.
- [27] Avid Roman-Gonzalez and Natalia Indira Vargas-Cuentas. Análisis de imágenes hiperespectrales. *Revista Ingeniería & Desarrollo*, Año 9(Nº 35):14–17, September 2013.
- [28] Esther Sánchez Bernabé. Procesado de imágenes hiperespectrales. 2016.
- [29] J. Sauvola and M. Pietikäinen. Adaptive document image binarization. *Pattern Recognition*, 33(2):225–236, 2000.
- [30] Streamlit Team. Streamlit: The fastest way to build and share data apps, 2024.
- [31] Ramón Sánchez, Carlos Rad, Carlos Cambra, Rocío Barros, and Álvaro Herrero. Detecting copper-based fungicides in vineyards by means of hyperspectral imagery. *Smart Agricultural Technology*, 12:101049, 2025.
- [32] R. Tosin et al. Assessing predawn leaf water potential based on hyperspectral data and pigment's concentration of vitis vinifera l. in the douro wine region. *Scientia Horticulturae*, 278, feb 2021.
- [33] Stéfan J. van der Walt, Johannes L. Schönberger, et al. scikit-image: image processing in Python. *PeerJ*, 2:e453, 2014.
- [34] Vinay A., Avani S. Rao, Vinay S. Shekhar, Akshay Kumar C., K N Balasubramanya Murthy, and S. Natarajan. Feature extraction using orb-ransac for face recognition. *Procedia Computer Science*, 70:174–184, 2015. Proceedings of the 4th International Conference on Eco-friendly Computing and Communication Systems.
- [35] Maria Laura Vranić, Claudio Delrieux, and Juan Vorobioff. Identificación de bandas espectrales claves en imágenes hiperespectrales para la detección de aflatoxinas en maní. *AJEA*, 2024.