ESCUELA DE INGENIERÍA DE FUENLABRADA

GRADO EN INGENIERÍA DE ROBÓTICA SOFTWARE

TRABAJO FIN DE GRADO

DESARROLLO DE UN SISTEMA DE VEHÍCULO AÉREO NO TRIPULADO PARA APLICACIONES DE AGRICULTURA DE PRECISIÓN

Autor: Diego Encabo del Castillo

Tutora: Xin Chen

Cotutor: José Miguel Guerrero Hernández

Curso académico 2023 / 2024

RESUMEN

La comprobación de la salud de un cultivo durante el transcurso de sus fases fenológicas, puede resultar en un proceso tedioso en el que se desaprovechan muchos de los recursos empleados. El avance en los últimos años de los sistemas de control y automatización, ha producido un avance en las técnicas utilizadas en agricultura dando lugar a la agricultura de precisión: una metodología de gestión de los cultivos cuyo objetivo es mejorar su producción y abaratar los costes.

Esta memoria tiene como intención aportar una solución a todo este proceso de análisis, teniendo como principal requisito: el uso de software libre y de elementos presentes en las instalaciones de la universidad para abaratar los costes. Para el caso de nuestra aplicación, se utilizará un dron con una serie de complementos añadidos, el cual se encargará de realizar una misión planificada con el objetivo de tomar imágenes y recopilar datos de un terreno con vegetación a cierta altura. Se presenta un diseño de arquitectura, tanto software como hardware, en el que se detalla la funcionalidad de cada uno de los elementos presentes en nuestro proyecto, además de la metodología seguida para que todos ellos trabajen en cohesión con un mismo objetivo. Además, se exponen una serie de pruebas en vuelo realizadas con la finalidad de verificar que el diseño estructurado inicialmente puede ser trasladado al mundo real, aparte de ajustar cualquier tipo de desviación por parte de la idea preconcebida del proyecto.

Finalmente, se utilizan todos los datos recogidos en los diferentes vuelos planificados realizados y se genera un diagnóstico parcial de cada una de las zonas capturadas por las diferentes imágenes con ayuda de la visión artificial y el procesamiento de imagen, en el que se pueden localizar los puntos del cultivo que requieren de intervención por parte del agricultor.

A mis padres:

Gracias por estar ahí siempre, os quiero.

A bubu:

Gracias por darme ese último empujón, te amo.

ÍNDICE

ÍNDICE DE FIGURAS

LISTA DE ABREVIATURAS

CCW	*Counter Clock Wise: Sentido anti horario.*
CW	*Clock Wise: Sentido horario.*
DDS	*Data Distribution Service: Middleware tipo publicador/subscriptor para sistemas distribuidos.*
EVI	*Enhanced Vegetation Index: Índice de Vegetación Mejorado.*
ExG	*Excess Green Index: Índice de detección del verdor en una imagen*
GB	*Gigabyte.*
GPS	*Global Positioning System: Sistema de Posicionamiento Global.*
ID	*Identificador.*
JST	*Japanese Solderless Terminal: Empresa japonesa que es una de las principales manufacturadoras de conectores para la industria eléctrica y electrónica.*
MAVLink	*Micro Air Vehicle Link: Protocolo de comunicación para pequeños vehículos no tripulados*
NDVI	*Normalized Difference Vegetation Index: Índice de diferencia de vegetación normalizado.*
NED	*North East Down: Sistema de referencia global*
NIR	*Near Infrarred: Infrarrojo cercano.*
pH	*potencial de Hidrógeno.*
PLA	*Polylactic Acid: Material utilizado en impresión 3D.*

PWM	*Pulse Width Module: Modulación por ancho de pulsos de una señal o fuente de energía.*
PX4	*PixHawk 4.*
QML	*Qt Meta Language: Lenguaje de programación utilizado para programar con Qt.*
QoS	*Quality of Service: política que se utiliza durante la comunicación entre nodos en ROS2.*
RAM	*Random Access Memory: Memoria que utiliza un programa en ejecución.*
RGB	*Red,Green,Blue: Sistema de composición de colores basado en la adicción de colores primarios.*
ROS	*Robot Operating System: Framework para el desarrollo de aplicaciones robóticas.*
Rpi4	Raspberry Pi 4
RTPS	*Real Time Publish Subscribe: Tipo de implementación de DDS.*
RX	*Pin de recepción de datos.*
SAVI	*Soil Adjusted Vegetation Index: Índice de Vegetación Ajustado al Suelo.*
TX	*Pin de transmisión de datos.*
UART	*Universal Asynchronous Receiver/Transmitter: Protocolo de intercambio de datos en serie.*
UAV	*Unmaned Aerial Vehicle: Vehículo aéreo no tripulado.*
URDF	*Unified Robot Description Format: Formato de descripción de robot unificado*
USB	*Universal Serial Bus: Bus de comunicación estandarizado que sirve para proveer de alimentación eléctrica a dispositivos electrónicos.*
VTOL	*Vertical Take Off and Landing: Despegue y aterrizaje vertical.*

1 Introducción

Desde hace aproximadamente 12.000 años, durante el Neolítico y cuando los seres humanos nos encontrábamos en transición de pasar de una vida nómada a una vida sedentaria en sociedad, el surgimiento de la agricultura supuso una revolución en la forma de pensar (Editors, 2018). Este cambio de mentalidad ocasionó dejar a un lado la caza y la recolección de bienes viajando de un lugar a otro, para centrarnos en desarrollar un sistema de cosechas que cumpliera con las necesidades alimenticias del asentamiento durante las diferentes estaciones del año.

Durante el avance de los años, se ha realizado constantemente un proceso de mejora en la forma en la que cultivamos, obteniendo avances que aumentan la productividad agrícola y la calidad de los alimentos para poder adaptarse a un incremento masivo de la población mundial. Pero la evolución en la agricultura no ha llegado a un tope: desde que se han producido grandes avances tecnológicos en la sociedad, tanto en hardware como en software, se ha abierto una nueva corriente de investigación de la agricultura enfocada a las nuevas tecnologías más innovadoras: Agricultura 4.0 (McCormick, 2021). Este nuevo tipo de metodologías tiene como principal objetivo aportar aplicaciones que optimicen la producción al máximo. En el caso de nuestro trabajo y dentro de este nuevo movimiento tecnológico, nuestro objetivo es evitar los desperdicios en una cosecha detectando con antelación la aparición de determinadas enfermedades o parásitos que agraven la salud de los cultivos.

Para ello, con la ayuda de distintos dispositivos tecnológicos y robóticos, buscaremos dar una solución a cómo podemos comprobar de primera mano la salud de los cultivos periódicamente, mejorar la calidad y producción de las cosechas, y reducir considerablemente los tiempos de mantenimiento.

1.1 Estado del arte

Cuando se quiere empezar a realizar un nuevo trabajo de investigación, lo primero que debemos hacer es observar distintos puntos de vista tomados en el mismo sector (Rivas, 2023). Con todas estas investigaciones previas, podemos obtener un contexto de la situación actual que puede ser realmente útil a la hora de tomar decisiones o enfocar distintas partes de nuestro trabajo.

En este apartado, vamos a intentar recopilar información de trabajos relacionados con nuestro campo, para poder sacar una serie de ideas de nuestro ámbito de investigación y cómo deberíamos de enfocar nuestro proyecto en la etapa de diseño. En este caso, lo dividiremos en: robótica aérea y agricultura de precisión.

1.1.1 Robótica aérea

Desde el primer vuelo controlado de manera automática en 1916 (Pedro Castillo García, 2016), las aplicaciones de UAVs en el ámbito civil y militar han aumentado considerablemente. Los UAVs de ala fija han sido normalmente el tipo de vehículo más utilizado durante estos años, donde las misiones de las que se encargaban eran de vigilancia. Sin embargo, en otro tipo de aplicaciones en las que el vehículo necesite flotar en el aire, es decir, permanecer en una posición con un vuelo quasi-estacionario (Lozano, 2001), será necesario utilizar UAVs de ala rotatoria o multirrotores que sean capaces de cumplir estos requisitos, además de permitir un despegue y aterrizaje vertical (VTOL)

En cuanto al comportamiento de estos vehículos no tripulados, se estabiliza mediante un controlador a bordo: encargado de recopilar los datos de distintos sensores y ajustar los parámetros de otros componentes a bordo para obtener el comportamiento deseado durante la misión. Por ejemplo, el controlador puede recibir la posición del vehículo en el Planeta Tierra del módulo GPS conectado a este, o puede también ajustar la velocidad de los motores que se encargan de hacer girar las hélices del vehículo. Mientras tanto, un operario supervisa de manera remota el estado de la misión y, en algunos casos, también controla la posición del vehículo. Esta conexión remota normalmente se realiza con un sistema de telemetría entre el operario y el vehículo, donde será necesario tener un dispositivo de transmisión y recepción conectado a ambos para que puedan comunicarse.

Además, para todas estas aplicaciones es muy importante la auto-localización del vehículo en el entorno de la misión, con una alta precisión de la estimación de la posición para poder interactuar con el entorno: saber dónde se encuentra el vehículo y qué se encuentra a su alrededor es imprescindible a la hora de navegar por un entorno.

En la actualidad, estos UAVs multirrotores denominados comúnmente como *drones*, se utilizan de manera muy diferente en trabajos profesionales de distintos sectores (Ruiz, 2019). Algunos ejemplos son:

a) Filmaciones y publicidad aérea: Se consigue abaratar el coste de obtener planos a vista de pájaro utilizando drones, a diferencia de aeronaves más caras como avionetas o helicópteros que suponen un mayor presupuesto.

b) Búsqueda y salvamento, vigilancia: Sobrevolando parques naturales en busca de posibles focos de origen de un incendio, o navegando por zonas de catástrofes naturales para tener una mejor perspectiva y poder localizar personas en busca de auxilio.

c) Fumigaciones y agricultura de precisión: Junto con otros dispositivos acoplados como cámaras, permiten obtener un análisis de la zona de cultivo a vista de pájaro que acelera los procesos de distribución de la densidad de plantas por cultivo y comprobación de la salud de la cosecha, para posibles fumigaciones o trasplantes de aquellas zonas en mal estado.

En este caso, la aplicación en nuestro trabajo consiste en conseguir que un dron mediante un controlador a bordo y una serie de componentes externos, consiga comunicarse con una estación de tierra (dispositivo que maneja el operario para supervisar la misión), con el objetivo de conseguir obtener imágenes de una zona de cultivo para comprobar la salud actual de la cosecha. Para ello, será necesario tener en cuenta todos los requisitos explicados previamente para un correcto comportamiento del vehículo durante la misión.

En posteriores secciones, explicaremos con detalle: los modelos que se han utilizado en los distintos componentes de la misión, la función que desempeña cada uno y qué les hace imprescindibles para un correcto funcionamiento durante el vuelo. Además, explicaremos también en profundidad la disposición y las conexiones del conjunto de todos ellos para formar uno durante la misión.

1.1.2 Agricultura de precisión

Hoy en día, hemos conseguido numerosos avances en tecnología que se han conseguido aplicar en distintos ámbitos laborales. Por ejemplo, los médicos pueden conseguir una mejor evaluación de sus pacientes gracias a métodos como la radiología o los análisis de sangre, para poder realizar un tratamiento específico para cada paciente (BBVA, 2023).

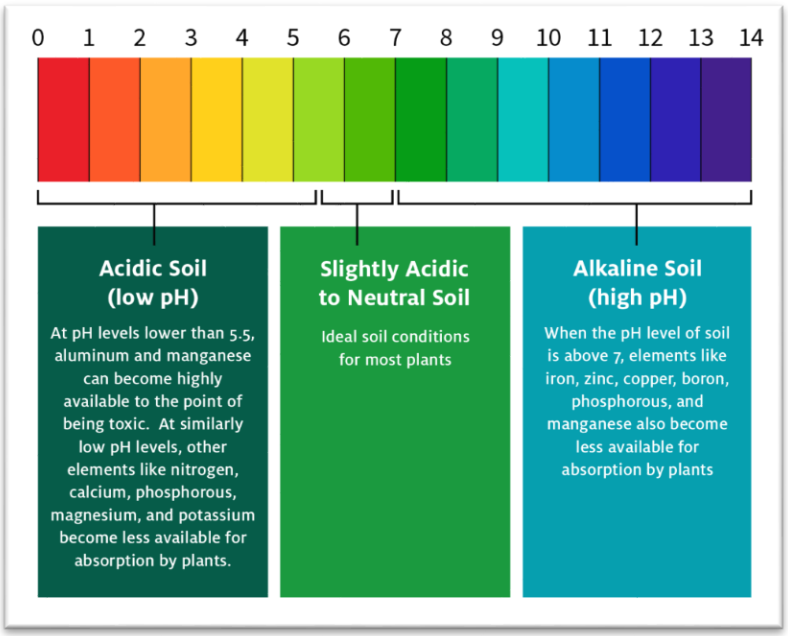


Figura 1-1: Salud del cultivo según el pH del suelo (FastGrowingTrees, 2023)

Basándonos en esta analogía con la medicina y al igual que en los humanos, cada planta dentro de un cultivo es un paciente diferente: habrá que realizar un diagnóstico concreto y dividirlo en pequeñas zonas localizadas, diferenciadas todas ellas por el problema con el que los agricultores tienen que lidiar. En vez de utilizar soluciones genéricas para ver si se soluciona el problema, podemos conocer de primera mano los problemas concretos que necesitan una solución: en algunas zonas secas será necesario suministrar más agua, en algunas otras zonas será necesario rociar pesticida o replantar debido a algún parásito presente, y en otras zonas será necesario aplicar fertilizantes nitrogenados o sulfatos para reducir el pH del suelo alcalino que impide un correcto crecimiento de las plantas (Sembralia, 2020). En la Figura 1-1 (FastGrowingTrees, 2023), podemos observar la salud de un cultivo en función del pH del suelo. Por tanto, para poder tener una idea clara del estado de un cultivo, con la ayuda de las nuevas tecnologías sería

más fácil obtener un diagnóstico concreto del estado de las distintas plantas que lo forman, reduciendo los costes de mantenimiento y el tiempo necesario para este.

Sin embargo, este nuevo método de agricultura no sólo puede ser aplicado para gestionar el estado de salud de las plantas. Se puede sobrevolar con un dron las parcelas del cultivo para poder conseguir las dimensiones completas de terreno cultivable, y así poder dividir el terreno de la forma más óptima para una productiva cosecha: se puede obtener de forma precisa cuántas filas de cultivo caben en dicho terreno y cuántas plantas caben por fila para un crecimiento apropiado.

Por tanto, la agricultura de precisión nos permite obtener una inmensa cantidad de datos de nuestro terreno para poder tomar la mejor decisión posible en términos de distribución, control de parásitos y salud del terreno y las plantas, todo ello abaratando los costes y el tiempo dedicado a ello. Un ejemplo de ello es cuando se procede a realizar un examen del cultivo en busca de plantas contaminadas por parásitos: si sabemos las plantas o zonas de cultivo concretas donde hay que fumigar, estamos reduciendo el uso de pesticida y reduciendo la cantidad de químicos presentes en el resto de la cosecha, produciendo un ahorro de pesticida y una mejora de la c de las plantas que evitan ser expuestas a sustancias nocivas, produciendo un ahorro tanto en coste como en tiempo de mantenimiento.

En relación a nuestro trabajo, necesitamos utilizar la agricultura de precisión para poder obtener la salud del cultivo a analizar. Como explicaremos detalladamente en los próximos apartados, necesitaremos distintos dispositivos tecnológicos (dron, cámara, GPS…) trabajando en conjunto con distintos softwares, que nos permitirán obtener un análisis correcto de las distintas zonas del cultivo que pueden ser candidatas a necesitar un tratamiento de mejora de su estado de salud.

1.2 Objetivos

Para el caso de nuestro proyecto, en el que se busca desarrollar un sistema de vehículo aéreo no tripulado aplicado a la agricultura de precisión, los objetivos durante el desarrollo de este trabajo son:

a) Captura de imágenes aéreas de los terrenos de cultivo a estudiar, junto con datos de importancia para su posterior análisis. Incluye la cohesión de los diferentes dispositivos utilizados durante el proyecto para la realización de dichos vuelos.

b) Estudio y aplicación de técnicas de detección de vegetación dentro de una imagen, cuya finalidad es generar un diagnóstico de la vegetación en una zona para que finalmente esa información sea interpretada.

c) Localización de los puntos críticos del terreno de cultivo en un sistema de referencia que nos permita geolocalizar con precisión estas áreas en el planeta Tierra.

d) Análisis y validación de los resultados obtenidos con la aplicación propuesta.

1.3 Planteamiento

Dados los objetivos presentados en el apartado anterior, se planifica una estrategia a seguir para la consecución de todos ellos. Ésta se define a continuación:

Objetivo a):

1. Búsqueda de las necesidades para la implementación y elección de los diferentes dispositivos necesarios durante el proyecto, además del software que utilizarán para su aplicación.

2. Diseño de una arquitectura hardware que permita la cohesión de todos los dispositivos para trabajar de manera unificada y sincronizada.

3. Diseño de una arquitectura software que permita la comunicación entre todos los dispositivos, con la ayuda de las conexiones físicas desarrolladas en el paso 2. Incluye la instalación de cualquier librería o aplicación necesaria para cumplir su función.

Objetivo b):

1. Estudio de las diferentes técnicas de detección de vegetación, exponiendo los resultados obtenidos con diferentes índices.

2. Investigación de una librería que permita aplicar dichas técnicas y generar una representación gráfica del diagnóstico mediante la manipulación de los datos obtenidos durante la misión.

Objetivo c):

1. Investigación del proceso para realizar una misión planificada con la aeronave y sobrevolar un cultivo. Búsqueda de un software que permita automatizar toda la misión.

2. Interpretación de los datos obtenidos con las técnicas de detección de vegetación aplicadas, para convertir los resultados obtenidos en una representación gráfica que muestre al usuario las áreas que requieren de intervención geolocalizadas.

Objetivo d):

1. Verificación del funcionamiento de todos los dispositivos de manera sincronizada y cohesionada durante la misión. Realización de diferentes pruebas para comprobar que todo funciona adecuadamente.

2. Análisis de los resultados obtenidos. Comprobación del funcionamiento de todo el proceso de análisis del terreno de cultivo con ayuda de los datos obtenidos durante la misión. Representación de los resultados de forma legible y con unas conclusiones coherentes para el usuario.

2 Metodología y componentes

En los proyectos de ingeniería como en muchas otras ramas de investigación, la base de toda aplicación en el mundo real es la metodología de trabajo que se va a seguir durante todo el proyecto. En ella, se busca realizar un planteamiento que, dados unos recursos concretos, se encargue de dar soluciones viables a todas las incógnitas dentro del trabajo, para posteriormente seguir con la aplicación de esta idea en el mundo real.

En este apartado, nos encargaremos inicialmente de dar solución a la forma en la que vamos a detectar la vegetación dentro del terreno de cultivo, para después continuar con la explicación de los diferentes dispositivos y programas que serán necesarios para hacer realidad nuestra solución dentro del proyecto. Para la elección de los distintos componentes, explicaremos: la función que cumple cada uno dentro del proyecto, el modelo concreto utilizado según los recursos disponibles por parte tanto de la universidad como de los integrantes del trabajo, y la instalación de algunos de ellos si fuera necesario.

2.1 Técnicas de detección de vegetación

En una aplicación de agricultura de precisión, una de las tareas más importantes a la hora de comprobar la salud de un cultivo es detectar la vegetación de antemano, para luego poder realizar un diagnóstico. Pero, antes de poder explicar cómo podemos detectar esta vegetación con el uso de la tecnología, es necesario hablar previamente sobre la luz, las plantas y algunas de las características que, con el tiempo, se ha visto que se cumplen como norma general.

¿Por qué vemos las plantas de color verde? Principalmente vemos las plantas de color verde por 2 razones: la primera es debido a la reflexión de la luz sobre cualquier objeto o superficie, mientras que la segunda es debido a su composición biológica. La luz solar incide sobre cualquier superficie, absorbiendo parte de esta luz y reflejando la restante. Esa luz reflejada incide en nuestros ojos, dando lugar al color con el que vemos los objetos. El color viene denotado por la longitud de onda de la luz reflejada, pudiendo ver una escala de los colores según su longitud de onda en la Figura 2-1 (Linazasoro, Linazasoro Optika, 2023).

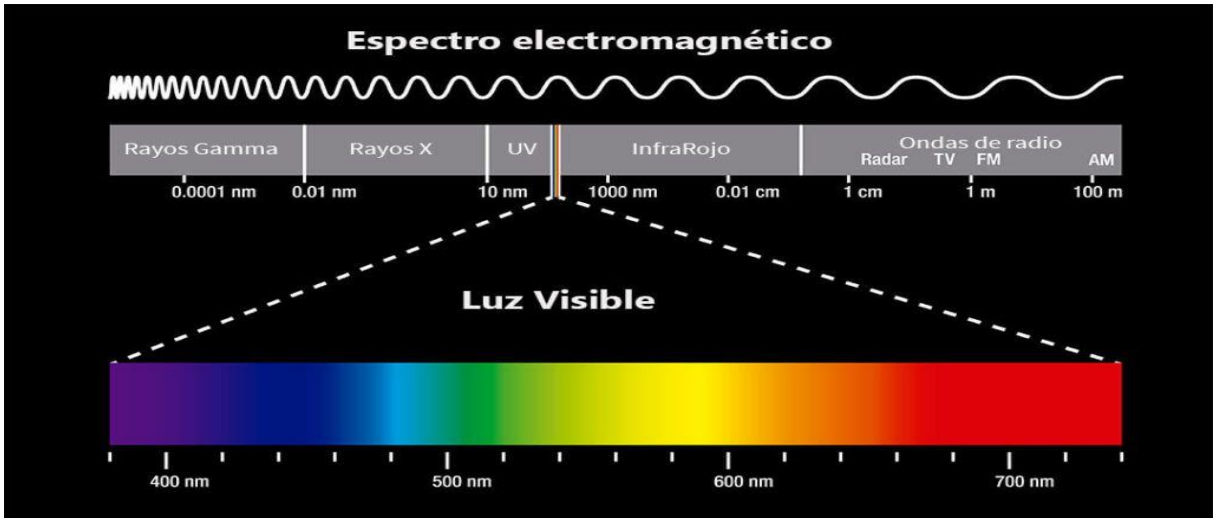


Figura 2-1: Espectro electromagnético de la luz (Linazasoro, Linazasoro optika, 2023)

Las plantas son unos organismos vivos que viven sujetos al suelo, es decir, durante su vida crecen sin moverse del sitio en el que empezaron su desarrollo. Para poder obtener energía y llevar a cabo este crecimiento, las plantas realizan un proceso llamado fotosíntesis: transforman la energía lumínica del sol utilizando dióxido de carbono, en distintas moléculas orgánicas que pueden utilizar los seres vivos como fuente de energía (carbohidratos), liberando oxígeno al finalizar el proceso (BioInteractive, 2020). Dentro de las hojas de las plantas, existen unas células mesófilas con una gran concentración de unos orgánulos celulares, encargados de realizar la fotosíntesis: los cloroplastos. Además, dentro de estos cloroplastos se encuentra un pigmento de color verde que es sensible a la luz del sol para poder absorberla: la clorofila. Por tanto, podemos llegar con toda esta información a una conclusión: si las plantas necesitan hacer la fotosíntesis, necesitan contener en sus hojas una gran concentración de células mesófilas para poder, tanto absorber la luz del sol como para poder realizar la fotosíntesis. Si existe una gran concentración de células mesófilas, existe una gran concentración de clorofila presente en dichas hojas. Es por ello que, cuánto más intenso sea el verdor de la hoja de una planta, mayor

será la actividad fotosintética de esta y mejor será su salud. Por tanto, una de las principales formas de diagnosticar la salud de una planta será detectar la concentración de células mesófilas y clorofila presente en sus hojas.

A partir de este principio, se realizó mediante el uso de las tecnologías una combinación de diferentes bandas espectrales detectadas por dispositivos como las cámaras, cuya función es ensalzar la vegetación y atenuar el resto de elementos de la imagen: suelo, iluminación, rocas, etc… A estas combinaciones espectrales, junto con un procesamiento de imagen y fórmulas matemáticas, se les conoce como índices de vegetación.

Uno de los índices más utilizados desde su introducción en los años 70 es el NDVI, el cual sigue siendo un estándar en las técnicas de detección de vegetación. Este índice se encarga de detectar la vegetación dados 2 espectros diferentes de la luz solar capturados en imagen: el espectro de la luz roja, el cual se encuentra dentro del espectro visible por el ser humano, y el espectro infrarrojo (más concretamente el infrarrojo cercano: NIR), presente en el espectro de la luz no visible por el ser humano. Este índice se basa en todo el contexto previo que hemos explicado sobre las plantas, pero aplicado al ámbito tecnológico: las células mesófilas reflejan notablemente el espectro infrarrojo cercano, mientras que la clorofila debido su color verde, absorbe notablemente el color rojo presente en la luz solar. En la Figura 2-2 (DST, 2023), podemos observar un esquema que resume brevemente la salud de una planta en función de ambos espectros de la luz.

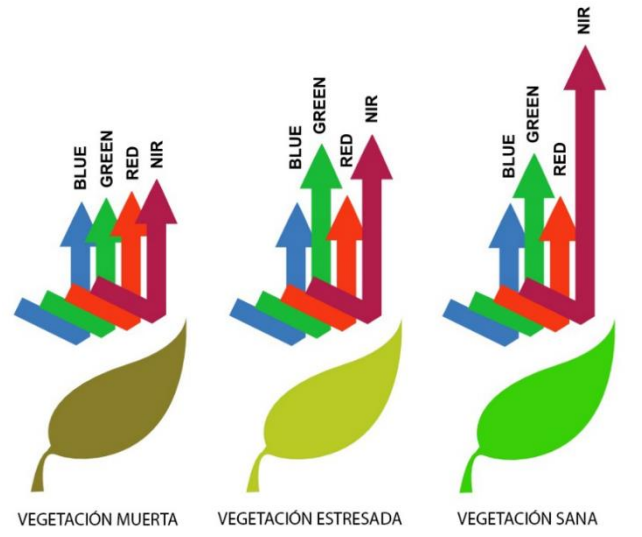


Figura 2-2: Representación de los diferentes estados de salud de una planta (DST, 2023)

Basándonos en la información presente en ambos espectros, para poder obtener el índice NDVI será necesario tener una imagen a color y una imagen en infrarrojo del cultivo (Toribio, 2019). Una vez obtenidas ambas imágenes, aplicaremos para cada uno de los píxeles de la imagen la fórmula:

$$NDVI = \frac{NIR - RED}{NIR + RED}$$

NIR = Valor de reflectancia del infrarrojo cercano en un píxel.

RED = Valor de reflectancia del canal rojo en un píxel de una imagen en color (RGB).

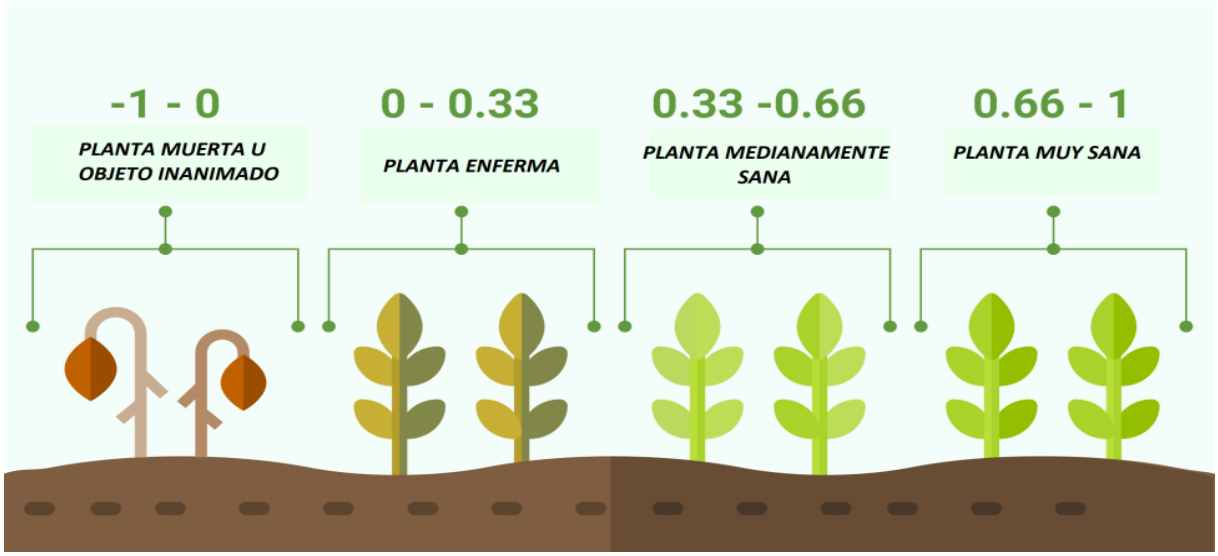


Figura 2-3: Aplicación de índice NDVI en las plantas (Gladys Toribio, 2020)

El valor resultante de la fórmula estará normalizado entre -1 y 1. Los valores negativos se considerarán como píxeles no pertenecientes a vegetación, mientras que los valores entre 0 y 1 describirán la salud de la vegetación en cada píxel. Los valores positivos más cercanos al 0 serán píxeles que describen que la planta se encuentra en mal estado, mientras que los valores más cercanos a RED = 0, describirán píxeles donde la salud de la planta es adecuada. En la Figura 2-3 (Gladys Toribio, 2020), podemos apreciar un ejemplo de aplicación de NDVI en la que vemos cómo se ha clasificado la salud de una planta respecto a los resultados del índice.

Sin embargo, este índice tiene una serie de inconvenientes que pueden hacer que no se comporte de forma adecuada: las características del suelo y la posible saturación de los valores según la fase fenológica del cultivo (germinación, floración y cosecha). Además, cabe destacar que este tipo de índices de vegetación han sido diseñados principalmente para ser utilizados con

imágenes satelitales, lo cual también puede afectar a los resultados obtenidos con un índice u otro. Debido a estas limitaciones existentes, se han desarrollado varios índices cuyo objetivo es mejorar los resultados obtenidos en terrenos que destacan los puntos débiles del NDVI. En el caso de nuestro trabajo, nos hemos centrado en 2 variantes: el índice SAVI y el índice EVI. Para ajustar estos índices a alturas menores de las imágenes satelitales, en algunas de las fórmulas será necesario probar con diferentes valores de constantes hasta obtener el resultado esperado. En nuestro caso, en este documento se aportarán los valores finales utilizados para realizar el procesamiento.

El índice SAVI fue diseñado con la idea de minimizar las influencias de la reflectancia de la luz solar en el suelo. Para ello, se agregó una constante de ajuste del suelo *L* a la fórmula con el objetivo de mitigar este efecto. El valor de un píxel con el índice aplicado se obtendría con la siguiente fórmula:

$$SAVI = \left(\frac{(NIR - RED)}{(NIR + RED + L)}\right) \cdot (1 + L)$$

NIR = Valor de reflectancia del infrarrojo cercano en un píxel.
RED = Valor de reflectancia del canal rojo en un píxel de una imagen en color (RGB).
L = Constante de ajuste de la reflectancia del suelo.

En cuanto al índice EVI, su objetivo es minimizar tanto la reflectancia del suelo como los efectos atmosféricos. Para ello, se añaden las constantes *C1* y *C2* como valores de resistencia atmosférica, además de añadir la constante de ajuste del suelo *L* como en el índice SAVI. La fórmula para obtener el valor del índice en un píxel es:

$$EVI = G \cdot \frac{(NIR - RED)}{(NIR + C_1 \cdot RED - C_2 \cdot BLUE + L)}$$

G = Constante de ganancia estática.
C_1 = Constante de resistencia atmosférica del canal rojo.
C_2 = Constante de resistencia atmosférica del canal azul.

Una vez realicemos la misión, aplicaremos el índice NDVI a las imágenes capturadas durante el vuelo. Si los resultados no son los esperados debido a los factores que hemos mencionado previamente, aplicaremos estas alternativas de índices.

2.2 Hardware

Una vez hemos hablado de cómo vamos a detectar la vegetación en los terrenos de cultivo, el siguiente paso a realizar es elegir los diferentes componentes necesarios para poder llevar a cabo el proceso previamente explicado.

En este apartado, hablaremos de cada uno de los principales dispositivos que vamos a utilizar durante la misión. Además, mencionaremos los diferentes accesorios o complementos que necesitan estos dispositivos en algunos casos, para realizar su tarea y mejorar los resultados obtenidos. Por último, destacar que en este apartado no vamos a profundizar en exceso sobre cómo se compenetran los dispositivos entre ellos para funcionar en conjunto, sino que en la sección: Desarrollo del proyecto, desglosaremos las diferentes formas de conexión entre los dispositivos.

2.2.1 Dron

Es una aeronave no tripulada manejada de forma remota mediante un dispositivo de radio control supervisado por un humano, o mediante un software ejecutado desde otro dispositivo electrónico el cual tiene comunicación directa con el vehículo. En nuestro caso y como explicaremos detalladamente en el apartado de software, utilizaremos un programa específico para planificar misiones con el objetivo de recorrer de forma completa el terreno de cultivo de forma automática. Respecto al vehículo elegido para nuestro proyecto y como ya mencionamos previamente en el comienzo de este apartado, es importante tener en cuenta los recursos disponibles por parte de la universidad a la hora de aportar una solución lo más accesible económicamente. En este caso, el dron es probablemente junto con la cámara, una de los elementos más caros dentro de los componentes utilizados, por lo que se priorizará en utilizar material presente en las instalaciones de nuestro campus. Aprovechando el material utilizado en el taller de drones de la asignatura Robótica Aérea de nuestro grado, el kit que vamos a utilizar como aeronave es el Holybro X500 V1. Actualmente este kit ha sido reemplazado por el kit Holybro X500 V2, por lo que no es posible desde la web oficial acceder al kit utilizado en este trabajo ni solicitar la compra de este (Holybro, 2023).

Sin embargo, desde la página oficial del controlador que viene incluido en este kit, tenemos una guía de usuario para poder montar este dron con el controlador incorporado. En la Figura 2-4 (PX4 Autopilot, 2023), tenemos representado el kit Holybro X500 V1 con el montaje ya realizado.

Figura 2-4: Kit Holybro X500 V1 montado (PX4 Autopilot, 2023)

Los principales componentes incluidos en este kit son:

- 1x Controlador de vuelo Holybro PixHawk 4.
- 1x Módulo GPS Holybro M8N.
- 1x Placa de gestión de voltaje PM07 (Power Management Board).
- 4x Motores Holybro 2216 KV880.
- 4x hélices 1045.
- 4x ESC para controlar la velocidad de los motores.
- 2x radio de telemetría Holybro 433MHz / 915MHz.

Además de todos estos componentes, hay que tener en cuenta que hay un montón de complementos y accesorios necesarios como pueden ser: embellecedores, cables de distintos tipos para todas las conexiones, plataformas para apoyar los distintos dispositivos, etc… Para mejor entendimiento de todo lo que incluye este kit, en la Figura 2-5 (PX4 Autopilot, 2023) podemos apreciar los diferentes componentes previamente mencionados y algunos de los complementos, designados cada uno de ellos con etiquetas.

Figura 2-5: Componentes del kit Holybro X500 identificados (PX4 Autopilot, 2023)

La función principal del dron consistirá en nuestro caso en ejecutar una misión planificada previamente en la que, mientras la aeronave sigue su curso, seremos capaces de: tomar imágenes ventrales del terreno de cultivo durante el vuelo, obtener a tiempo real la información de los diferentes componentes gracias al controlador (posición GPS y odometría, orientación, velocidad) y abortar la misión en cualquier momento de manera remota en caso de que ocurra cualquier evento inesperado. Obviamente, el dron es el encargado simplemente de recorrer el espacio del terreno de cultivo desde el aire, por lo que todas estas demás funciones las realizarán otros dispositivos implementados en la aeronave, que inicialmente no pertenecían a ella. En los próximos subapartados, explicaremos las especificaciones y función de cada uno de ellos.

2.2.2 Controlador del vuelo PixHawk4

Es el encargado de gestionar los diferentes elementos electrónicos pertenecientes a la aeronave e interconectar toda su información en un solo dispositivo. Es posible comunicarse con el controlador directamente para modificar el comportamiento del vehículo, siempre que haya un medio de comunicación instalado tanto en el dispositivo controlado por el humano como en la aeronave (normalmente mediante telemetría). Además, es el encargado de suministrar alimentación a algunos de los componentes como: el GPS o la antena de telemetría. Respecto al modelo elegido en este caso, debido a que el kit del dron previamente presentado incluía un controlador manufacturado por la empresa, utilizaremos dicho modelo: Holybro PixHawk 4. Este controlador ha sido diseñado siguiendo un estándar por el equipo PX4, mientras que Holybro se ha encargado de adaptarlo a su modelo de dron y ensamblar todos los procesadores en una única pieza.

Figura 2-6: Controlador Holybro PixHawk 4 (AliExpress, 2024)

En la Figura 2-6 (AliExpress, 2024), podemos apreciar el modelo de controlador utilizado con los diferentes puertos de conexión disponibles para los diferentes componentes de la aeronave. Cabe destacar que no tiene por qué ser necesario utilizar todos los puertos durante una misión, por ejemplo, podemos apreciar que existen 2 puertos de telemetría (TELEM1 y TELEM2), que sólo serán utilizados si necesitamos que más de un dispositivo reciba los datos del controlador. Sin embargo, los 2 puertos de alimentación que existen (POWER1 y POWER2) es importante utilizar ambos durante el vuelo por motivos de seguridad: el controlador necesita recibir alimentación desde 1 de los puertos, pero tener 2 en funcionamiento sirve como una medida de

redundancia por si cualquiera de ellos fallara durante la misión, siempre hubiera una segunda opción y el dron no estuviera en riesgo de perder la alimentación del controlador y todos los dispositivos conectados a este. Todas estas observaciones respecto a las conexiones necesarias entre los puertos del controlador y los diferentes componentes de la aeronave, serán expuestas en el apartado 3.1: Arquitectura de hardware.

2.2.3 Cámara

Es el primero de los dispositivos que vamos a presentar que ha sido añadido al dron fuera del kit inicial. Para el desarrollo de nuestro proyecto, lo más importante es utilizar una cámara que, aparte de que tome fotos de una buena calidad, sea capaz de tomar tanto imágenes RGB como de infrarrojos. Esto se debe a que, para poder aplicar uno de los índices de vegetación expues en el apartado 2.1, necesitamos ambos espectros para poder procesar los datos y conseguir la imagen resultado. Además, será necesario que tenga algún punto de montaje (por ejemplo, un hueco para un tornillo) para poder acoplar la cámara a la aeronave. La manera en que ha sido implementado su acople al dron, será desarrollada en la parte de Arquitectura de hardware como el resto de accesorios complementarios. Pero, ¿por qué necesitamos una cámara que tenga tanto un sensor de imágenes a color como de infrarrojos?, ¿no se puede con un mismo sensor capturar todo el espectro de la luz? La mayoría de cámaras comerciales que encontramos, por ejemplo en teléfonos móviles, no suelen capturar todo el espectro de la luz solar (Linazasoro, 2023). Estas cámaras tienden a capturar el espectro de luz cuya longitud de onda se encuentra en el espectro visible por el ojo humano, aunque dentro de los rayos solares se encuentre, en una parte de la luz absorbida, un espectro de luz no visible por el ser humano formado por: el espectro ultravioleta, el espectro infrarrojo, los rayos Gamma, etc.... Por tanto, las cámaras se encargan de capturar los rayos de luz solar, permitiendo pasar a una parte del espectro mientras que impiden al espectro de luz no visible pasar mediante: filtros físicos delante de la lente, o eliminándolo con un post-procesamiento tras haber capturado la imagen mediante filtros software. Si queremos capturar otro de los espectros de luz no visibles, necesitaremos otro sensor diferente que se encargue de capturar esta información. Por tanto, necesitaremos una cámara que tenga 2 sensores diferentes: uno que se encarga de capturar el espectro de luz visible y otro que se encarga de capturar el espectro no visible, más concretamente el espectro infrarrojo.

Teniendo en cuenta esta información, la empresa Intel ha desarrollado una serie de cámaras que aportan una solución de visión computarizada orientada al entorno de la robótica: la tecnología Intel RealSense (Intel RealSense, 2023). Este tipo de cámaras utilizan una visión estereoscópica mediante un par de cámaras infrarrojos a ambos lados del dispositivo, para calcular la profundidad de los elementos que se encuentran alrededor de su entorno. Además, toda esta serie de cámaras contienen una cámara RGB, capaz de tomar imágenes a color desde prácticamente el mismo ángulo que las cámaras de infrarrojos (en el apartado de tratamiento de imagen explicaremos en detalle cómo alinear ambas imágenes). Teniendo en cuenta el material disponible en los laboratorios de la universidad, inicialmente se empezó a trabajar con la cámara Intel RealSense R200, representada en la Figura 2-7 (idorobotics, 2018). Esta cámara, lanzada al público en junio de 2016, pertenece a las primeras cámaras que se desarrollaron dentro de esta nueva tecnología. Contiene un par de cámaras infrarrojos, un proyector de rayos de este mismo espectro y una cámara RGB.

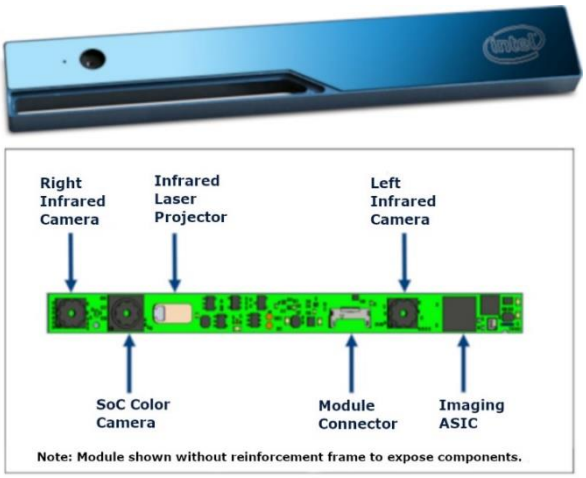


Figura 2-7: Exterior e interior de la cámara Intel RealSense R200 (idorobotics, 2018)

Sin embargo, tras un tiempo trabajando con ella para poder implementarla en el proyecto, surgieron una serie de contratiempos que forzaron a buscar otras opciones. El principal problema es que, para poder almacenar imágenes procedentes de estas cámaras, es necesario acoplar al dron una unidad de procesamiento que contenga un puerto USB 3.0. En nuestro caso y como explicaremos con detalle en el próximo apartado, la opción económica con puerto USB 3.0 disponible era la Raspberry Pi 4. Este pequeño ordenador, aparte de ser el único con este tipo de puerto, si se le instala Ubuntu sólo se permite a partir de la versión 19.10 (ROS Answers, 2020). La cámara Intel RealSense utiliza una librería llamada librealsense en su primera

versión, la cual tiene soporte hasta la versión 16.04 de Ubuntu (Intel RealSense Support, 2020). Si se conectara esta cámara en una edición de Raspberry anterior (Rpi2, RPi3), debería de conectarse en un puerto USB 2.0. Conectar un dispositivo con USB 3.0 a un puerto USB 2.0 supone una diferencia en ancho de banda permitido para transmitir, lo que limitaría notablemente la calidad de las imágenes de la cámara.

Tras descartar este modelo de cámara, la siguiente elección y definitiva fue la Intel RealSense D435. Esta cámara pertenece a la última serie de cámaras de la tecnología RealSense: Serie D400. El número de cámaras disponibles, al igual que en la cámara Intel RealSense R200, son: un par de cámaras de infrarrojos, un proyector de infrarrojos y una cámara RGB (Intel RealSense, 2023). A diferencia de la R200, la cámara Intel RealSense D435 utiliza la librería librealsense2, la cual tiene soporte en Ubuntu 20.04, versión disponible para instalar en la Raspberry Pi 4. Para saber más sobre la instalación de esta librería, vaya a **este enlace** (GitHub, 2024). En la Figura 2-8 (Intel RealSense, 2024), podemos observar el interior y exterior de la cámara, junto con la ubicación de los elementos que la componen.

Figura 2-8: Exterior e interior de la cámara Intel RealSense D435 (Intel RealSense, 2024)

2.2.4 Raspberry Pi 4

Hemos mencionado que el controlador va a ser el encargado de proporcionarnos la información y el estado de los distintos componentes del dron, mientras que la cámara nos proporcionará imágenes a tiempo real durante la misión. Sin embargo, por el momento la cámara y el controlador no están comunicados de ninguna manera, es decir, ¿cómo sabe la cámara en qué

momento hay que capturar las imágenes? Además, el controlador PX4 utiliza un software específico que la cámara no sabe traducir, al igual que la cámara no sabe interpretar la información del dron. Por tanto, ¿Quién unificara la información de ambos dispositivos en un lenguaje común? ¿Quién almacenará los datos para poder procesar después de la misión?

Este tipo de preguntas nos llevan a pensar que todavía es necesario añadir un dispositivo más a la aeronave durante el vuelo: una unidad de procesamiento capaz de poder recoger la información, interpretarla y actuar en consecuencia una vez se haya generado una solución. Es por ello que en nuestro trabajo necesitaremos añadir un ordenador a bordo, que sea lo más simple posible en cuanto a diseño, pero con la potencia suficiente para recopilar los datos necesarios durante la misión de manera exitosa. Con estas características en la mesa y atendiendo al material disponible en los laboratorios de la universidad, la solución que más se adapta a nuestro proyecto sería una Raspberry Pi, concretamente la Raspberry Pi 4 Modelo B. Es importante matizar que necesitaremos este modelo concreto de Raspberry Pi, dado que actualmente durante el desarrollo de este trabajo es el único modelo que contiene puertos USB 3.0: si recordamos el subapartado donde hemos hablado de la cámara, las cámaras de la gama Intel RealSense requieren de mínimo un puerto USB 3.0 al que conectarse para poder transmitir la información con el ancho de banda correspondiente.

En cuanto a más detalles sobre las características del producto, la Raspberry Pi 4 que teníamos disponible tiene una memoria RAM de 4GB. Existen distintos tamaños de memoria RAM para este modelo (1GB, 2GB, 4GB y 8GB), así que aconsejamos que, para no tener problemas durante la implementación, se utilice el modelo con mayor memoria RAM disponible. Otras de las características de este producto que son importantes de mencionar para futuras secciones son: 1 puerto USB tipo-C para alimentación (5V 3A), 2 puertos USB 3.0 y una tarjeta de red Wi-Fi 2.4/5 GHz. Respecto a las preguntas mencionadas al inicio, no hemos especificado mucho en esta parte sobre cómo se solucionarán algunos de los problemas que han surgido: unificar los mensajes en un formato común y almacenar la información para post-procesamiento. Este tipo de temas se abordarán tanto en el apartado 2.3: Software como en los apartados 3.1: Arquitectura de hardware y 3.2: Arquitectura de software.

2.2.5 Estación de tierra (portátil)

Todo vehículo aéreo no tripulado necesita ser supervisado durante su misión, ya sea mediante un operario que controla la aeronave de manera remota, o mediante un dispositivo tecnológico controlado por una o varias personas que comprueba el status a tiempo real. De momento, ya tenemos todos los dispositivos necesarios a bordo, pero ¿cómo se realiza la comunicación del dron tierra-aire durante la misión? ¿cómo obtenemos la información a tiempo real durante el vuelo? ¿cómo podemos comprobar si la Raspberry está obteniendo los datos y está almacenando las imágenes?

Se conoce como estación de tierra al centro de control terrestre que permite el manejo humano de UAVs de manera remota. Normalmente, los UAVs militares que se consideran aeronaves de un tamaño considerable (puede llegar a ser mayor que el de una avioneta) son controlados por estaciones de tierra en las que varios operarios supervisan el estado (Wikipedia, 2023). Además, como dichas estaciones suelen estar a una gran distancia de la zona donde se va a realizar la misión (muchas veces operan en territorio hostil), se utilizan sistemas de comunicación de largo alcance como la comunicación satelital. Pero además de estas estaciones fijas, existen para las aeronaves pequeñas una serie estaciones de tierra portables, las cuales contienen: una computadora interna encargada de procesar los datos obtenidos durante el vuelo, un transmisor de radio control encargado de poder manejar la posición y orientación de la aeronave de manera remota, y una pantalla que muestre al operario la situación a tiempo real con la ayuda de un software específico.

En nuestro caso, utilizaremos como estación de tierra un portátil de cualquier marca comercial, con el único requisito que, para el software que vamos a utilizar en este proyecto, es necesario tener instalado Linux, concretamente Ubuntu 20.04. La comunicación entre el portátil y la aeronave se hará mediante telemetría, utilizando ambas antenas que contiene el kit Holybro X500 mencionado previamente, conectando una a bordo y otra en el portátil mediante conexión por puerto USB. El manejo de la posición y la orientación del dron es automático en nuestro caso, gracias a una misión planificada creada previamente mediante un programa que explicaremos en futuros apartados. Nuestra estación de tierra nos permitirá ver a tiempo real la situación de la misión, tanto del dron mediante la comunicación con el controlador por telemetría, como del software ejecutado en la Raspberry Pi: podremos comprobar si se están tomando y almacenando las imágenes, si está detectando los waypoints de la misión, si alguno de los programas ejecutándose ha dejado de funcionar debido a algún fallo, etc…

2.2.6 Radio control

Encargado de manejar remotamente la aeronave desde tierra, controlando la velocidad y maniobrabilidad del vehículo. Aunque nuestra misión planificada será automatizada por el software de la estación de tierra, el control remoto también es de gran utilidad de otras formas. Previamente a realizar una de las misiones planificadas, podemos realizar una serie de pruebas de vuelo cortas en la que manejamos manualmente el dron para comprobar que todo está montado adecuadamente y que el comportamiento en el aire es el adecuado. Además, este tipo de controles remotos tienen una serie de canales que se pueden asignar para realizar diferentes funciones, principalmente cambiando el modo de vuelo en cualquier momento. El modo de vuelo consiste en el comportamiento que tomará la aeronave, es decir, la forma en la que controlará su trayectoria y la trayectoria que realizará. En nuestro caso, los modos de vuelo que utilizaremos en las misiones son:

- Mission: el dron realizará la misión planificada que tiene cargada en memoria.
- Return: el dron abortará cualquier acción previa y volverá al punto inicial previo a despegar (home).
- Position: el dron se maneja manualmente mediante el operario. Automáticamente cuando esté en vuelo estacionario, el dron neutralizará cualquier fuerza externa que pueda variar la posición de este, como el viento.

Figura 2-9: Control remoto FlySky-6iX (FlyingTech, 2024)

En nuestro caso, vamos a utilizar el transmisor FlySky-i6X que tenemos disponible en los laboratorios de la universidad. En la Figura 2-9 (FlyingTech, 2024), podemos observar una imagen del modelo utilizado.

2.3 Software

Una vez presentados los principales componentes del proyecto, vamos a hablar sobre los diferentes programas que van a permitir a estos dispositivos cumplir con sus funciones durante la misión. En este apartado: explicaremos cada uno de los softwares que se utilizan, qué dispositivo utiliza cada uno de ellos y cuál es su función dentro del proyecto. Por último, matizar que todo el software utilizado en este proyecto es Open Source, por lo que no se requiere de ninguna licencia para poder instalar cualquiera de estos programas.

2.3.1 PX4 AutoPilot

Se trata de un programa que proporciona un estándar para poder controlar una gran variedad de vehículos de diferentes estilos: UAVs de ala fija y multirrotores, vehículos terrestres (rovers) e incluso vehículos marinos y submarinos. Una de las principales ventajas es su flexibilidad: permite una amplia serie de distintos modos de vuelo e implementaciones de seguridad, además de estar integrado con mucha profundidad, pudiendo ser implementado junto con computadoras y frameworks robóticos (por ejemplo, ROS 2). PX4 Autopilot pertenece junto con otros softwares (MAVSDK, MAVLink y QGroundControl) al proyecto DroneCode: una comunidad estadounidense cuyo objetivo es aportar una ayuda gratuita a todos los proyectos de drones de código abierto (DroneCode, 2023).

En cuanto a su funcionamiento, PX4 utiliza los diferentes dispositivos que se encuentran conectados al controlador donde se encuentra instalado para determinar el estado del vehículo. Además, en nuestra situación en la que la aeronave sigue una misión ya planificada, este estado del vehículo permitirá que durante la misión se aporten soluciones una vez procesados los datos, para poder corregir la estabilización y el control autónomo para llegar a los sitios de forma precisa. En el caso de querer realizar un vuelo autónomo, PX4 necesitará además del hardware esencial (giroscopio, acelerómetro, barómetro y brújula), un GPS u otro sistema de

posicionamiento para reducir el error de posición y orientación durante el vuelo (PX4 Autopilot, 2023). Aparte de aportar al usuario información a tiempo real, también permite que el propio operario sea capaz de modificar ciertas trayectorias (por ejemplo, forzar al dron a volver la posición inicial). Para este tipo de operaciones, PX4 trabaja en conjunto con otro programa específico en el que es posible visualizar el estado de todos los datos recopilados por el controlador desde la estación de tierra: QGroundControl.

2.3.2 QGroundControl

Aplicación que funciona como supervisor de las misiones para aeronaves que utilizan el protocolo MAVLink. Como hemos mencionado anteriormente, QGroundControl pertenece junto con PX4 y MAVLink al proyecto DroneCode, el cual ha aportado una solución accesible a todo el mundo para poder trabajar con todos estos programas en cohesión. Sobre la aplicación, concretamente se trata de una interfaz gráfica de usuario la cual permite al operario de vuelo personalizar de manera libre la manera en la que se va a mostrar la información durante el vuelo. La interfaz gráfica está implementada utilizando Qt QML, permitiendo poder ser instalada en dispositivos portables más simples que un portátil, como móviles o tablets. De todas maneras, para nuestra implementación en la que necesitamos un ordenador para poder supervisar a otros componentes durante el vuelo, la aplicación será instalada en este mismo dispositivo.

Otro de los principales usos de la aplicación es la calibración de los distintos componentes del dron: previamente a poder realizar una misión, la aplicación se encargará de diagnosticar el funcionamiento de los distintos dispositivos conectados al controlador, para comprobar si todo se encuentra en orden y ajustar algunos parámetros en caso de que existan algunos errores de cálculo. Para poder realizar todo este proceso, QGroundControl tiene un apartado completo dentro de su aplicación donde explica componente a componente como poder realizar las distintas calibraciones, además de poder ajustar manualmente otros componentes como el voltaje mínimo/máximo de la batería. Todo este proceso contiene partes explicadas de manera gráfica mediante imágenes ilustrativas de los pasos para facilitar todo el procedimiento (por ejemplo, la calibración del giroscopio). Por último, la aplicación contiene una herramienta de creación de misiones planificadas para poder construir una trayectoria concreta de puntos en el plano, la cual el dron debe seguir de manera automatizada. Esta herramienta permite, tanto

elegir localizaciones concretas con ayuda de un mapa de la Tierra a vista satelital, como auto generar patrones que ocupen un espacio concreto por el que la aeronave tiene que ir visitando puntos realizando una trayectoria de cobertura: el dron recorre una zona delimitada realizando un patrón en forma de S.En nuestro caso y como explicaremos con más detalle en la sección de pruebas en vuelo, utilizaremos todas estas herramientas de la aplicación para calibrar el dron y realizar una misión planificada que recorra de manera automática el terreno de cultivo, para posteriormente utilizar QGroundControl como herramienta de supervisión durante el transcurso de esta.

2.3.3 ROS 2

Consiste en un conjunto de librerías software y herramientas que habilitan al usuario a poder crear aplicaciones robóticas de una manera más sencilla. Una vez en funcionamiento, se trata de un middleware basado en un sistema de publicador/subscriptor para permitir la comunicación de distintas aplicaciones entre ellas. Toda información que se comparte está empaquetada en topics: método para transportar información entre los diferentes programas en ejecución de ROS 2 (nodos) con un formato concreto de mensaje. Cada topic se diferencia de otro principalmente por un nombre específico, lo cual permite a los nodos poder acceder a dicha información. Para que un nodo comparta información con otros, utilizará un publicador con el nombre del topic que contiene ese formato de mensaje, para almacenar en la nube la información con los demás nodos. Al contrario, si un nodo quiere consultar información de un topic concreto, será necesario que dicho nodo cree un subscriptor a ese topic además de que otro nodo tenga un publicador activo que comparta con los demás nodos sus datos (ROS 2, 2023). Los nodos cuando empiezan a ejecutarse utilizan una operación bloqueante en el código para que se dedique a comprobar constantemente los eventos, es decir, normalmente cuando un nodo crea un suscriptor para obtener datos, al recibir esos datos se llama directamente a una función que se encarga de hacer algo concreto con ellos, llamada callback. Los publicadores envían la información de forma periódica, mientras que los subscriptores reciben la información y la procesan en los callbacks en el momento que se reciben los datos, es decir, si por ejemplo un callback procesa los datos en un tiempo mayor que la próxima publicación, esos datos se almacenarán en una cola de datos que se irá procesando según se acabe con los datos previos. De todas formas, este comportamiento se puede modificar ajustando la calidad de servicio del topic: QoS (ROS 2, 2024).

En cuanto a la forma en la que los nodos pueden ver los topics de los demás nodos, no es necesario que todos los topics se ejecuten en el mismo dispositivo. Si conectamos varios dispositivos a una red y todos ellos tienen ROS 2 instalado, se podrán comunicar entre ellos mediante los topics, acciones o servicios, siempre y cuando no se añadan configuraciones adicionales como seguridad en ROS 2 o un ID de dominio en las máquinas: si se asigna un dominio a una máquina concreta, aunque pertenezca a la misma red que otros dispositivos que están publicando información en topics, sólo serán visibles aquellos topics cuyas máquinas pertenezcan al mismo dominio dentro de esta red. Por defecto, el dominio es el mismo para todas las máquinas instaladas con ROS 2, salvo en casos en los que se configure un ID de dominio concreto para encapsular los datos. En nuestro caso, ROS 2 será la manera en la que conseguiremos tanto la información de la cámara como del controlador, traducida al mismo lenguaje para unificar la información recopilada por todos los componentes en un nodo, el cual será el encargado de decidir cuándo se capturan las imágenes y se almacena información adicional para el post-procesamiento y la posterior aplicación del índice de detección de vegetación.

2.3.4 Qt

Desarrollada inicialmente por los fundadores de la empresa Trolltech (Haavard Nord y Eirik Chambe-Eng), se trata de una compañía noruega que desarrolla diferentes herramientas y bibliotecas de desarrollo de software. Su biblioteca más importante es Qt: una biblioteca multiplataforma de código abierto, principalmente utilizada para desarrollar aplicaciones que utilicen una interfaz gráfica de usuario. Un ejemplo de aplicación desarrollada con Qt sería QGroundControl, cuya interfaz permite al operario abstraer diferentes operaciones a nivel de forma gráfica y más accesible al usuario.

Relacionado con el proyecto, es posible combinar Qt con ROS 2 para poder realizar aplicaciones que permitan interactuar gráficamente al operario con otros nodos de ROS 2, por ejemplo, enviando/recibiendo la información de los diferentes topics para supervisar el estado de los datos que se encuentran en la nube. Inicialmente en la idea de nuestro proyecto, el objetivo era desarrollar una aplicación con Qt que mostrara la información de las imágenes RGB e infrarrojos, obtenida de los topics de ROS 2 que publican dicha información. Cuando el

dron se encontrase en la posición deseada, se comprobaría en la aplicación que las imágenes obtenidas son de buena calidad, para presionar un botón de la interfaz que permitiera almacenar una captura de cada una de las imágenes, además de otros datos obtenidos mediante otros topics como posición y orientación. El principal inconveniente que tenía esta implementación es la calidad de la conexión: si recordamos la explicación previa sobre ROS 2, si varias máquinas tienen ROS 2 instalado y pertenecen a una misma red de internet, todas las máquinas podrán acceder a la información de los topics utilizados por todos los nodos ejecutados en estas máquinas. La aplicación se ejecutaría en la estación de tierra, mientras que los topics serían publicados por los nodos ejecutados en la Raspberry Pi 4 a bordo del dron. Durante la misión, existen tramos en los que la distancia entre ambos dispositivos es mayor que el rango de alcance máximo de la conexión de red, por lo que durante intervalos de tiempo esta aplicación no tendría efecto alguno. Por ello y como explicaremos en la Arquitectura de software, decidimos dejar la aplicación de Qt como una herramienta complementaria de supervisión.

En la Figura 2-10, podemos ver un ejemplo de ejecución de la aplicación creada con Qt, en la que se pueden apreciar las imágenes en tiempo real de las cámaras de color e infrarrojos, unas etiquetas que muestran la posición y la orientación del dron, y un botón que al pulsar permite capturar la imagen de ambas cámaras en el momento. Para saber más sobre la implementación, véase **este enlace** (GitHub, 2024).

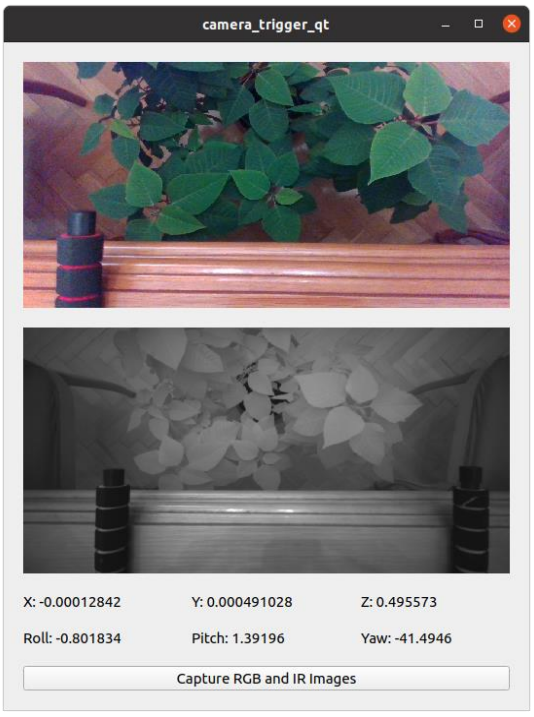


Figura 2-10: Aplicación Qt creada para la misión

2.3.5 OpenCV

Hemos hablado ya sobre varios programas o aplicaciones que conforman el software utilizado durante este proyecto. Sin embargo, todos ellos se encargan de realizar su función ya sea durante la misión o previamente a ella. Por tanto, necesitaremos algún software que nos permita tras la misión realizar el post-procesamiento de los datos recopilados para: alinear las imágenes RGB e infrarrojos de una misma posición, aplicar el índice de vegetación en todas ellas y finalmente calcular los puntos críticos de salud del cultivo y mostrarlos en la imagen con sus coordenadas GPS. OpenCV es una biblioteca de software libre de visión artificial desarrollada en sus inicios por Intel, cuyo objetivo es dar una amplia solución al tratamiento de imagen de manera accesible a todo el mundo (Wikipedia, 2023). Esta biblioteca contiene una extensa cantidad de librerías que permiten hacer distintas operaciones de tratamiento de imagen y vídeo (OpenCV Docs, 2023):

- Procesamiento de imagen: filtrado lineal y no lineal, transformaciones geométricas (escalado, resize, conversión en otros espacios de color, histogramas, etc…).
- Análisis de video: Estimación del movimiento, borrado del fondo y algoritmos de seguimiento de objetos.
- Calibración de cámaras mono y estéreo.
- Detección de objetos: de algunos objetos ya predefinidos (caras, ojos, tazas, personas, coches, etc…).

Además de contener diferentes librerías que permiten poder procesar los datos de imagen y vídeo, contiene una librería (highgui) que permite mostrar gráficamente las imágenes en cualquier parte del proceso, lo cual resulta muy útil a la hora de depurar el código del procesamiento de imagen. En nuestro caso, OpenCV se utilizará principalmente con la ayuda de los datos recopilados durante la misión, para obtener la imagen resultado del terreno de cultivo con el índice de vegetación aplicado, usando las diferentes librerías que aporta esta biblioteca. Para obtener esta imagen, se realizarán una serie de pasos intermedios que previamente se explicarán, tanto en el apartado 3.3: Procesamiento de imagen, como en la sección 5: Resultados

3 Desarrollo del proyecto

Una vez presentados los principales componentes hardware, su función y los programas que se van a utilizar durante el proyecto, el siguiente paso es hablar sobre cómo se van a utilizar estos elementos en conjunto, es decir, cómo se comunican entre ellos tanto de manera física como digital. En esta sección, primero desarrollaremos las arquitecturas hardware y software para tener un mejor entendimiento de la forma en la que cada componente recibe/envía la información, además de cómo traducen e interpretan todos estos datos cada uno de ellos.

Para el apartado 3.1: Arquitectura de hardware, se ha desarrollado una figura representativa de la arquitectura de hardware del dron para comprender mejor las diferentes conexiones físicas que existen para su correcto funcionamiento, junto con la posición de los diferentes componentes en la aeronave. Cabe destacar que algunos de los elementos en la figura han sido simplificados para que resulte más sencillo seguir la estructura. En cuanto al apartado 3.2: Arquitectura de software, explicaremos detalladamente los diferentes programas o protocolos que serán necesarios para que todo funcione en conjunto, además de hacer hincapié en algunos casos en los que será necesario un puente que permite que dispositivos con formatos de mensajes distintos, sean capaces de interpretar la información y comunicarse entre ellos.

Por último, hablaremos en profundidad sobre el procesamiento de imagen: explicaremos las diferentes operaciones que se llevan a cabo para obtener la imagen resultado con el índice de vegetación, es decir, toda operación previa o paso intermedio por el que se transcurre desde que se tienen las imágenes de color e infrarrojos almacenadas, hasta que se convierten en la imagen resultado.

3.1 Arquitectura de hardware

Se trata de la representación de las diferentes conexiones físicas que existen entre los diferentes dispositivos electrónicos, para conseguir el funcionamiento de todos ellos en conjunción como si de un único dispositivo se tratase. Para esta parte de la arquitectura, hay que tener en cuenta que en nuestro caso hay 2 principales dispositivos en los que clasificar nuestra arquitectura de hardware: el dron y la estación de tierra. Como ya hemos mencionado previamente, para el caso del dron se ha realizado un esquema gráfico que representa la arquitectura de hardware de la aeronave durante la misión, con algunos componentes simplificados. Dado que el dron en nuestro caso se divide en 2 plataformas de componentes: plataforma inferior y plataforma superior. Realizaremos esta división para explicar con una mejor visibilidad los componentes de cada una de estas plataformas, aunque es importante tener en cuenta que en la realidad todos ellos están agrupados en el mismo vehículo. Por tanto, empezaremos explicando la arquitectura de arriba abajo, empezando primero con la plataforma inferior para finalizar con la plataforma superior.

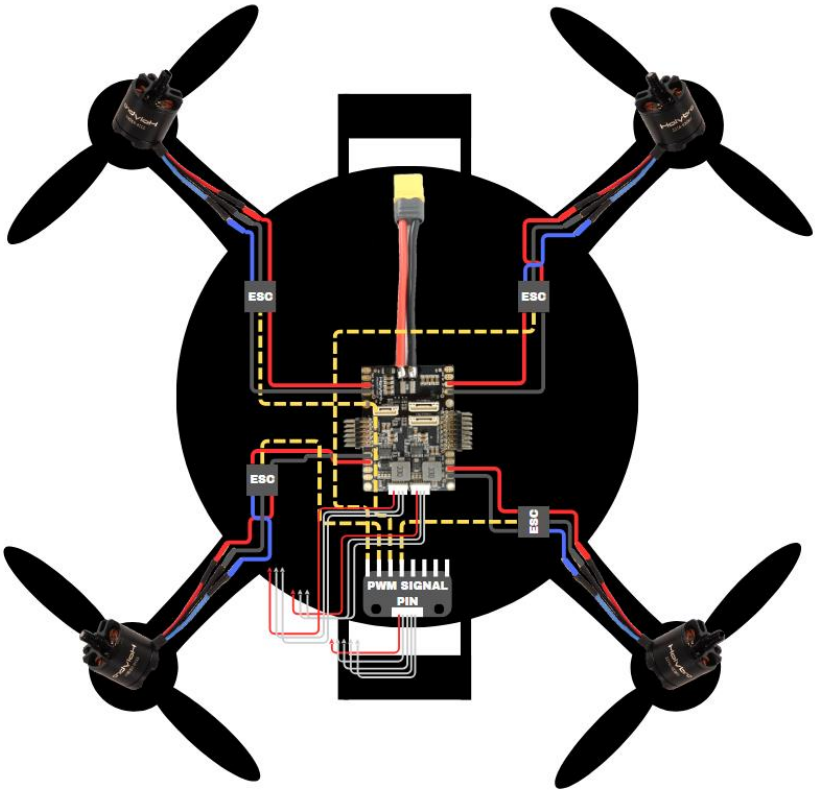


Figura 3-1: Arquitectura de hardware de la plataforma inferior del dron

Como podemos apreciar en la Figura 3-1, la plataforma inferior está formada principalmente por: la placa base de gestión de alimentación en el centro de la plataforma, los 4 motores a las esquinas de la aeronave y el pin que recibe la señal de PWM de cada motor debajo de la placa base. La batería no aparece en este dibujo debido a que se encontraría enganchada debajo de esta plataforma, así que, para mejor visibilidad del resto de componentes, se ha decidido no implementarla en la figura. Sin embargo, desde la placa base podemos apreciar como por la parte superior salen un par de cables: rojo de alimentación (5V) y negro de toma de tierra (GND), que se unen a un puerto XT60 para conectarse con la batería. Esta conexión permitirá que la batería suministre la alimentación a la placa base, la cual contiene diferentes puertos para administrar electricidad al resto de componentes de la aeronave. Además, desde las esquinas de la placa base se suministra un par de cables soldados de voltaje y de toma de tierra a los motores, mientras que desde la parte inferior aparecen 2 puertos JST hembra de color blanco, cuyas conexiones formadas por varios cables no terminan de conectar con ningún elemento de la figura, además de representarse con una flecha al final. Esto es debido a que estos 2 puertos conectarán en la plataforma superior de la aeronave con el controlador PX4, para suministrar alimentación tanto al controlador como a cualquier dispositivo que vaya conectado a este, por ejemplo, el módulo GPS. Existen 2 conexiones de alimentación desde la placa base al controlador como método de redundancia para asegurarse que, si durante el vuelo falla una de las conexiones, existe una segunda conexión que mantenga el suministro de electricidad a los diferentes dispositivos conectados.

Previamente a explicar las conexiones de los motores, es necesario realizar un inciso para tener un contexto previo respecto a la manera en la que estos se encuentran conectados. Según la ley de acción y reacción de Newton, cuando un motor de la aeronave realiza un movimiento circular en un sentido concreto para propulsar el vehículo hacia arriba, se produce una fuerza de reacción en el sentido opuesto que genera un torque aplicado al dron (GCFGlobal, 2023). Es decir, para contrarrestar este torque generado por el impulso de uno de los motores, será necesario que exista otro motor que realice el mismo movimiento circular y con la misma fuerza, pero en el sentido opuesto. De esta manera, un motor anula el torque del otro y viceversa, permitiendo que el dron pueda realizar un vuelo de manera estable. Además, para que siempre se anulen estos torques reactivos será necesario que la fuerza de los propulsores que giran en un sentido sea igual a la fuerza de los motores del sentido opuesto. Una vez tenido en cuenta esto, ahora queda decidir el patrón de posicionamiento de los motores en la aeronave. Hay varias maneras de

solucionar este problema, pero la estructura más estandarizada en este caso por cuestiones de estabilidad del vehículo es la formación en X de los motores.

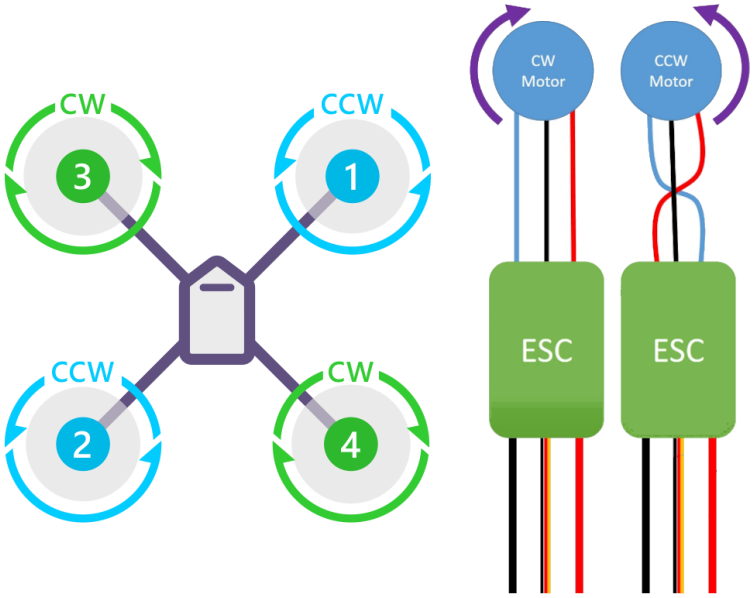


Figura 3-2: Estructura de conexión de los motores del dron

Como podemos apreciar en la Figura 3-2, los motores que giran en un sentido se encuentran colocados de forma diagonal, mientras que los motores que giran en el sentido opuesto se colocan en la diagonal opuesta, dando lugar a una forma similar a la letra X del alfabeto latino. Pero si nos fijamos también detalladamente, los cables no van conectados de la misma forma, sino que para conseguir que unos motores giren en sentido horario (CW) y otros en sentido anti horario (CCW), será necesario modificar la manera en la que los cables van conectados del ESC al motor, tal y como se expresa en la parte derecha de la imagen. Volviendo de nuevo a la figura 3-1, esta estructura de conexión de los motores en X se puede apreciar en el esquema al igual que la conexión de los cables según el sentido de giro. Para que sea más fácil diferenciar qué par de motores gira en sentido horario y qué par gira en sentido anti horario, Holybro añadió unas pegatinas de color plateado (antihorario) y negro (horario) en la parte superior de estos. Este mismo proceso se repite con las hélices acopladas, que tendrán el color de la rosca de encaje según el motor al que van acopladas. Por último, de los motores se transmite el PWM de cada uno de ellos mediante el cable rayado de color amarillo. Todos estos cables del PWM irán conectados al pin de señal de PWM, el cual se conectará al controlador en la plataforma superior para poder supervisar durante el vuelo el estado de los motores.

En la plataforma superior, encontramos los principales componentes que se encargan de la misión, junto con varios complementos para que puedan desarrollar su función adecuadamente. Al igual que con la plataforma inferior, en la Figura 3-3 podemos apreciar un esquema gráfico de la plataforma superior del dron, formada por: el controlador Holybro PX4, la antena de telemetría, el receptor del control remoto, el módulo GPS, la cámara Intel RealSense D435, la Raspberry Pi 4 y una batería portátil para alimentar tanto a la cámara como a la RPi4.

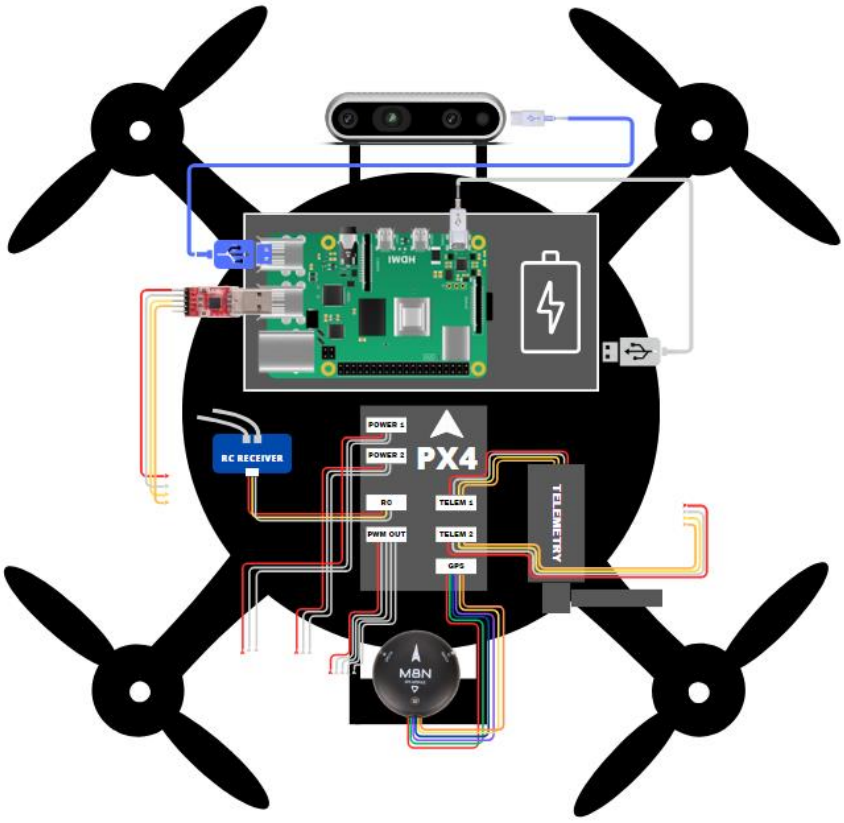


Figura 3-3: Arquitectura de hardware de la plataforma superior del dron

El controlador es el encargado de recopilar la información de todos los componentes que aportan datos de gran utilidad durante la misión. Si nos fijamos en cómo está representado este en la figura, se trata de una simplificación de la realidad respecto de la Figura 2-6. Esta simplificación se debe a que aquellos puertos que no aparecen en el dibujo no serán necesarios para nuestro proyecto, pudiendo así mostrar una versión con más espacio para una mejor visibilidad de las conexiones que nos interesan. Los diferentes puertos se representan con un rectángulo blanco, el cual contiene un texto indicando qué va conectado en ese puerto. En nuestro caso, hay 7 principales puertos que utilizaremos para este trabajo: POWER1, POWER2,

TELEM1, TELEM2, RC, PWM OUT y GPS. Los puertos POWER1 y POWER2 tienen conectados los 2 cables que provienen de la placa base para administrar alimentación al controlador. Como ya mencionamos en la plataforma inferior, conectar 2 cables de alimentación es una medida de seguridad (redundancia). El puerto RC conecta el receptor del control remoto al controlador, para poder comandar la aeronave desde tierra. El puerto PWM tiene conectado el cable del pin de señal PWM, proveniente desde la plataforma inferior también para comandar la velocidad de giro de los motores desde la estación de tierra vía el controlador. Estos 3 puertos tienen conectados cables originarios de la plataforma inferior representada en la Figura 3-1, donde no conectaban con ningún elemento. Los 2 siguientes puertos que podemos apreciar son TELEM1 y TELEM2, correspondientes a donde irán conectados los dispositivos de telemetría para recibir los datos de todos los componentes que vayan conectados al controlador. El puerto de TELEM1 normalmente está conectado a una antena de telemetría que envía los datos a bordo a la estación de tierra, mientras que el puerto TELEM2 suele dejarse en desuso. Pero en nuestro caso, también necesitamos recibir los datos en la Raspberry Pi 4, encargada de almacenar las imágenes de sitios concretos del terreno de cultivo, además de algunos datos que aporta el controlador como la posición y la orientación. Para poder recibir la información del controlador en la Raspberry Pi 4, es necesario adaptar el medio de conexión entre ambos dispositivos, ya que el controlador utiliza puertos JST mientras que la Raspberry Pi tiene varios puertos USB.

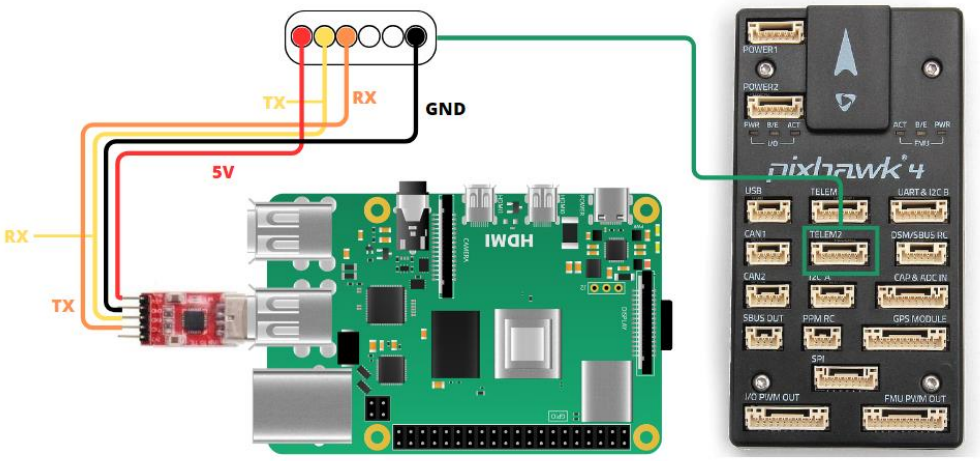


Figura 3-4: Conexión Raspberry Pi 4 con controlador PX4

En la Figura 3-4, podemos apreciar de forma más detallada la conexión entre la Raspberry Pi y el controlador PX4 para que ambos puedan intercambiar información durante la misión. En esta conexión, se está utilizando el protocolo serial UART: un protocolo de comunicación bidireccional entre 2 dispositivos, que permite el intercambio de datos en serie de forma asíncrona. Como podemos ver en la imagen, se requiere de 4 pines para hacer funcionar la

conexión, cada uno de ellos representado con su nombre correspondiente (5V, GND, RX y TX). El pin RX es el encargado de recibir los datos, mientras que el pin TX se encarga de transmitirlos, es decir, que para que la comunicación funcione adecuadamente, desde un extremo el pin TX debe de conectarse con el pin RX y viceversa. Esto se puede ver reflejado en el dibujo, donde los cables amarillo y naranja tienen expresados con letras del mismo color el nombre del pin al que se conectan, apreciando como en ambos extremos del cable se conectan a uno diferente. El puerto GPS tendrá conectado, como su nombre indica, el módulo GPS utilizado durante la misión. En cuanto a la posición del GPS, este dispositivo se encuentra en la parte trasera del vehículo para que tenga la mayor distancia posible con los demás dispositivos electrónicos y así evitar que se produzcan interferencias. Por último, el puerto tendrá conectado un cable procedente del pin de señal de PWM de los 4 motores, para comandar la velocidad de giro de los motores desde la estación de tierra vía el controlador. El controlador irá sujeto a la plataforma mediante una goma espuma adhesiva, que permita que pueda absorber parte de la fuerza que puede sufrir el dispositivo en caso de algún golpe. La antena de telemetría irá sujeta con una brida a la plataforma, que evita que se mueva o deslice cuando el dron realice maniobras en vuelo. En cuanto a la sujeción del GPS, se encuentra acoplado a un poste que le eleva a una altura superior a la de la plataforma, para aumentar aún más la distancia y mejorar la calidad de los datos obtenidos.

Al tener todos los puertos de alimentación ocupados, no teníamos ninguna manera de administrar alimentación a la Raspberry Pi a bordo. Además, conectar este dispositivo a la alimentación proveniente de la batería de la aeronave no sería la mejor solución, ya que provocaría que esta se agotase antes de lo normal, reduciendo el tiempo de vuelo para una misión. Debido a esto, se ha añadido una batería portátil a la aeronave que administrará alimentación a la Raspberry Pi y a todos los periféricos que vayan conectados a esta. La conexión RPi-batería portátil será mediante un cable con puertos USB-C y USB, donde el USB-C irá conectado al puerto de Raspberry Pi dedicado a la alimentación, y el USB irá conectado a cualquiera de los puertos de la batería portátil dedicados para administrar energía. La sujeción de la batería portátil a la plataforma será mediante una cinta adhesiva de doble cara, mientras que la Raspberry Pi utilizará también la cinta adhesiva de doble cara, pero pegada a la parte superior de la batería. Además, para no estropear ninguna parte de la Raspberry Pi con los adhesivos, se ha acoplado a una carcasa protectora de aluminio, la cual permite también proteger al dispositivo de cualquier daño.

En cuanto a la cámara necesitamos que, además de que vaya sujeta a la aeronave, su orientación sea con las lentes apuntando hacia el suelo actuando como cámara ventral, para que cuando el dron realice la misión permita realizar fotos panorámicas del suelo. Para ello, se ha realizado una pieza con impresora 3D que permita cumplir ambos requisitos, partiendo de un diseño original que aporta la propia empresa para acoplar cámaras.

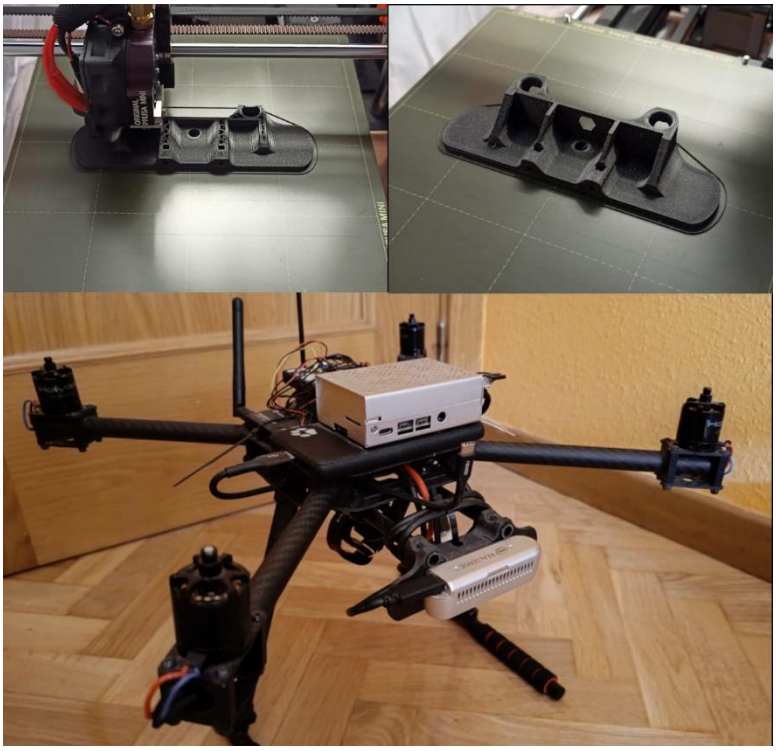


Figura 3-5: Creación de la pieza 3D (arriba) y acople de la cámara al dron (abajo)

Como podemos apreciar en la Figura 3-5, podemos ver en las imágenes de la parte superior el proceso de creación de la pieza, mientras que en la parte inferior se representa el dron con la cámara acoplada. El modelo de la pieza, como ya hemos mencionado, ha sido obtenido de una versión inicial desarrollada por la empresa Holybro a la que ha habido que hacerle una modificación para añadir el orificio en la parte central de la pieza, donde se introduce el tornillo que engancha la cámara. El modelo de impresora utilizado es Prusa MINI+ (PRUSA, 2023) y el material con el que se ha realizado la pieza es filamento PLA: un termoplástico hecho a base de maíz y caña de azúcar, que es un estándar para modelos diseñados en impresoras 3D debido a su fiabilidad y a su asequible precio (el coste de material utilizado para la pieza fue de 20 céntimos) (Mástoner, 2021). La pieza junto con la cámara acoplada irá colocada en la parte frontal de la aeronave, sobresaliendo un poco para que las imágenes no se vean obstaculizadas por partes del vehículo como el tren de aterrizaje. De todas formas, en el apartado 3.3:

Procesamiento de imagen, explicaremos qué podemos hacer en los casos en que las imágenes se ven obstaculizadas por las patas, para eliminar estos fragmentos de la imagen y obtener una imagen resultado libre de elementos ajenos a la vegetación. La cámara irá conectada a la Raspberry Pi con un cable USB-c/USB al igual que la conexión RPi-batería portátil, donde la cámara recibirá la alimentación mediante un conector USB-C y la Raspberry Pi conectará a uno de sus puertos USB el cable para poder suministrarla. Estos cables se encuentran representados en la Figura 3-3 con los colores azul (cámara-RPi4) y gris claro (batería-Rpi4).

Una vez hablado de la arquitectura de hardware del dron en ambas plataformas, el elemento restante sobre el que hay que hablar es la estación de tierra. Al tratarse un portátil comercial, la principal arquitectura viene ya desarrollada por el fabricante (disco duro, microprocesador, ventilación, etc…), por lo que sólo mencionaremos aquellos complementos que se han añadido para que la misión funcione de manera adecuada. En nuestro caso, el único complemento que se ha añadido es la otra antena de telemetría que contiene el kit de desarrollo del dron, la cual permite que la estación de tierra reciba los datos de manera telemática desde la antena a bordo del dron. Además, estos datos recopilados mediante la telemetría nos permitirán ver de forma gráfica el transcurso de la misión desde la estación de tierra utilizando QGroundControl.

3.2 Arquitectura de software

La arquitectura de software define el conjunto de programas y aplicaciones que, con la ayuda de las diferentes conexiones explicadas en el apartado anterior, permite mantener las comunicaciones necesarias entre componentes activas y ejecutar la misión de manera satisfactoria. En todo este conjunto encontramos: aplicaciones con interfaz de usuario para que interactúe el operario, programas que procesan los datos para tomar una decisión y puentes digitales entre diferentes programas para que puedan entenderse y comunicarse. Como en el apartado anterior, se ha realizado una figura en la que se expresa de forma gráfica qué software se ejecuta en cada uno de los dispositivos, además de qué información envía cada uno de ellos y cuál de ellos la recibe. En esta figura, no aparece ningún elemento de la plataforma inferior del dron debido a que los componentes que se encuentran en esta parte no requieren de ningún software específico para poder realizar las funciones que ya hemos explicado previamente.

En la Figura 3-6, podemos apreciar el esquema de la arquitectura de software durante la ejecución de la misión. En ella podemos ver: la cámara, el controlador, la Raspberry Pi y la estación de tierra o portátil. De cada dispositivo aparece una conexión a un rectángulo, el cual nos indica los programas o aplicaciones que se están ejecutando dentro de cada uno de ellos. Por otro lado, las flechas indican el envío de información de un dispositivo a otro, indicando el emisor y el receptor con la dirección de la flecha, y la información enviada contenida en una nube del mismo color que la flecha. El medio por el que se envía esta información aparece representado justo encima de la flecha, ya sea mediante una conexión física en el caso de los componentes a bordo, o mediante una comunicación a distancia cuando sea una conexión tierra-aire.

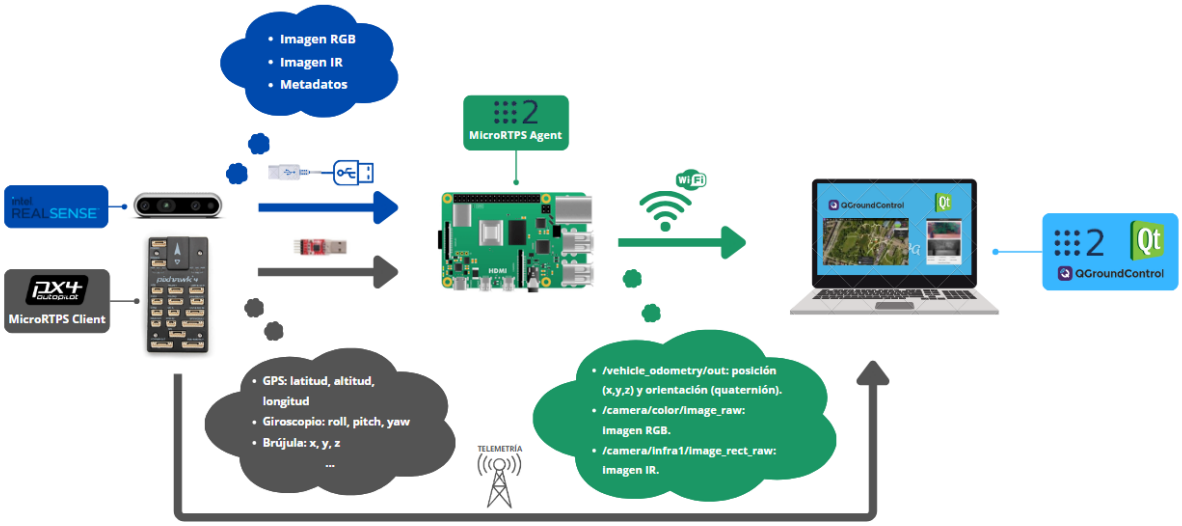


Figura 3-6: Arquitectura de software durante la misión

A la izquierda de la figura podemos ver la cámara Intel RealSense D435, encargada de tomar imágenes y enviarlas a la Raspberry Pi a tiempo real durante la misión. El software que se ejecuta en dicho dispositivo es la tecnología Intel RealSense, desarrollada por la propia empresa y que se encarga de tomar imágenes y mejorar su calidad con diferentes post-procesamientos. La cámara enviará a la Raspberry Pi las imágenes a color e infrarrojos, junto con una serie de metadatos como pueden ser: la distancia focal de la lente, la inclinación de la cámara, resolución de las imágenes. etc… Este envío de información se puede realizar gracias a la conexión física entre ambos dispositivos mediante un cable con puertos USB/USB-C, representado en la figura 3-3 con un cable de color azul. El controlador se encarga de recibir la información de todos los componentes conectados a él mismo y, con ayuda de PX4, publicarla periódicamente en topics

uORB para poder consultarla en todo momento. Como podemos apreciar en la imagen, el controlador manda su información tanto a la estación de tierra como a la Raspberry Pi. La información enviada a ambos dispositivos son todos los datos de los componentes acoplados al controlador, pero en nuestro caso los más importantes son: posición GPS (latitud, altitud y longitud) de la aeronave, orientación tomada con el giroscopio (roll, pitch, yaw) y expresada mediante un cuaternión, y posición de odometría (x, y, z) respecto de la posición inicial (home position) de la misión. A la estación de tierra, se enviará dicha información mediante telemetría con ayuda de las 2 antenas que contiene el kit Holybro X500: una conectada al controlador y otra conectada al portátil. En cambio, la información se enviará a la Raspberry Pi mediante el adaptador UART serial /USB, representado en la Figura 3-4. Pero esta conexión no sólo necesita un adaptador hardware para poder permitir el paso de información, sino que como ya hemos mencionado previamente, PX4 utiliza uORB topics para publicar la información, mientras que ROS 2 utiliza sus propios topics también, por lo que será necesario utilizar un intermediario software que se encargue de que ambos extremos se entiendan. A este tipo de programas se les conoce en informática como puentes entre 2 arquitecturas diferentes (PX4 AutoPilot, 2021), donde se utilizará un protocolo de comunicación común entre ambos, para adaptar el formato de mensajes de un dispositivo al formato del otro.

En la Figura 3-7 (PX4 AutoPilot, 2021), podemos apreciar el esquema el cual explica el funcionamiento del puente entre PX4 y ROS 2. A la izquierda rodeado por un rectángulo negro tenemos la parte ejecutada en el controlador, mientras que en el lado derecho podemos observar un rectángulo de color azul marino que engloba la parte ejecutada en la Raspberry Pi. Ambos dispositivos utilizarán para poder comunicarse microRTPS: un protocolo de comunicación que permite una solución publicador-subscriptor entre 2 programas que utilizan una red con protocolo DDS para comunicarse (FIWARE Zone, 2023). Desde el lado del controlador se ejecutará un cliente microRTPS, el cual se encargará de mandar la información proveniente de los topics uORB al otro extremo de la conexión. Al otro extremo de la conexión, estará un agente microRTPS ejecutándose en la Raspberry Pi que se encargará de comprobar si existen actualizaciones del lado del cliente, para recopilar esa nueva información (PX4, 2021). Para que microRTPS funcione, es necesario utilizar una red UDP o una conexión serializada mediante UART, como podemos apreciar en la figura representada con un rectángulo de línea discontinua de color morado que engloba ambos programas. En nuestro caso y como ya hemos explicado en el apartado 3.1: Arquitectura de hardware, se utilizará una conexión serializada

UART mediante el adaptador físico representado en la Figura 3-4. De esta manera, podemos ver desde el lado de ROS 2 como ya sería capaz de transformar los uORB topics en ROS 2 topics, para que los demás nodos puedan acceder a ella. Si fuera necesario, se puede hacer el caso inverso: publicar desde un ROS topic información, que luego recibirá el dron durante el vuelo como uORB topic. En cuanto a su instalación, se ha seguido el procedimiento de **este enlace** (PX4 AutoPilot, 2021).

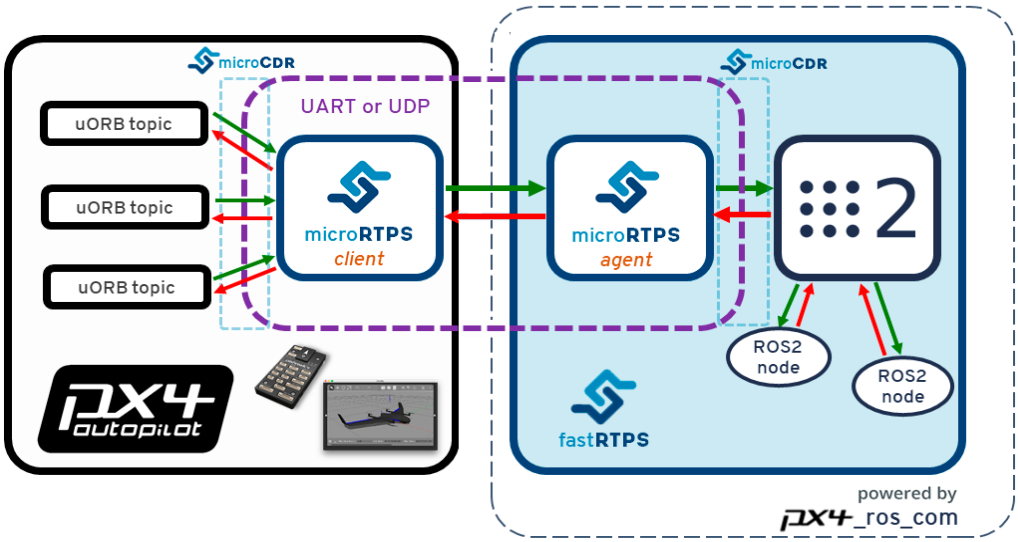


Figura 3-7: Puente PX4-ROS 2 (PX4 AutoPilot, 2021)

La Raspberry Pi se encargará de: recopilar la información de la cámara y el controlador, utilizar el puente PX4-ROS 2 para unificar el formato de los datos, y ejecutar un nodo de ROS 2 el cual será el encargado de tomar las decisiones durante la misión. Para recibir la información proveniente del controlador, se estará ejecutando un agente microRTPS, mientras que para recibir la información de la cámara se ejecutará un nodo de ROS 2 procedente del paquete realsense-ros creado por Intel: se encarga de realizar un puente entre la tecnología RealSense y ROS 2 para convertir los datos de la cámara e imágenes en ROS 2 topics (GitHub, 2023). Esta información unificada en ROS 2, será visible en el dominio de ROS 2 en el que se encuentra trabajando la Raspberry Pi. En nuestro caso, la red será visible para toda máquina conectada al ser el dominio por defecto. Debido a esto, para que la información a bordo pueda ser recibida en tierra mediante ROS 2, se creará una red privada Wi-Fi en la que ambos dispositivos pertenecen a ella, siendo la estación de tierra el creador de la red utilizando un punto de acceso con contraseña (Hotspot Wi-fi), y la Raspberry Pi un dispositivo conectado a esta nueva red. Este tipo de red se puede realizar debido a que ambos dispositivos tienen una tarjeta de red acoplada, pero si alguno de ellos no la tuviera sería necesario instalar una.

Por último, la estación de tierra se encargará principalmente de aportar herramientas de supervisión a tiempo real al operario, para poder saber lo que ocurre de primera mano durante el vuelo. Para ello, se ejecutarán 2 aplicaciones diferentes: QGroundControl y Qt. QGroundControl recibe la información procedente de la conexión controlador-estación de tierra mediante las antenas de telemetría. Así el operario será capaz de visualizar los datos de cada componente, aparte de observar el progreso de la misión planificada. La aplicación de Qt mostrada en la Figura 2-10, mostrará al operario las imágenes a tiempo real en ambos espectros junto con la posición y orientación del dron. Esta información será recibida por la aplicación con la ayuda de los topics de ROS 2, mediante la red privada creada previamente.

3.3 Procesamiento de imagen

Una vez finalizado el vuelo y después de haber recopilado toda la información necesaria, el siguiente paso es utilizar los datos recopilados para poder obtener un diagnóstico del terreno de cultivo. Este diagnóstico consiste principalmente en realizar un tratamiento específico de las imágenes obtenidas de la misión planificada, con el objetivo de conseguir un análisis completo de las plantas pertenecientes a la actual cosecha. En este apartado, hablaremos sobre los diferentes métodos de procesamiento de imagen utilizados en nuestro proyecto, explicando su función junto con una breve explicación de los pasos que realizan cada uno de ellos. Las principales operaciones utilizadas son: registro, segmentación y modificación de imágenes. Todas ellas han sido implementadas utilizando la librería OpenCV, la cual incluye varios tutoriales que explican todos los pasos que vamos a realizar de manera exhaustiva.

3.3.1 Registro de imágenes

Proceso de transformación entre un conjunto de varias imágenes que busca superponer los patrones similares de todas ellas en una (Escobar, 2019). Este proceso se puede utilizar tanto para crear fotos panorámicas a partir de varias fotos independientes, como para alinear una imagen respecto de la otra en orientación y escala. En nuestro caso, cuando capturamos en un waypoint las imágenes infrarrojos y RGB, existe un desplazamiento entre ambas, como

podemos apreciar en la Figura 3-8. Por tanto, para poder calcular nuestro índice necesitamos que los píxeles de ambas imágenes coincidan para extraer la información de manera adecuada.

Figura 3-8: Imágenes RGB (izquierda) e infrarrojos (derecha) tomadas desde la misma posición

El registro o alineamiento de imágenes el proceso por el cual se encuentran patrones similares de una imagen respecto de otra, permitiendo relacionar los píxeles de una imagen con los de otra. De esta manera, será posible alinear ambas imágenes para conseguir que los píxeles coincidan. El proceso consiste primero en conseguir los puntos característicos o puntos clave de una imagen: aquellos puntos que son estables a transformaciones de imagen, es decir, que su aspecto no varía con estas modificaciones (Mallick, 2018). Normalmente, estos puntos son los píxeles que delimitan los bordes de un objeto. Los puntos característicos están formados por:

- Localizador: punto de la imagen (píxel) que es estable a transformaciones de imagen, es decir, que su representación no varía con estas modificaciones, aunque se encuentre en distinta posición.

- Descriptor: Array de números que describe la apariencia de los alrededores del punto localizador.

Tras obtener los puntos característicos de cada una de las imágenes de manera independiente, el siguiente paso es recorrer ambas listas de puntos característicos para encontrar aquellos que coinciden y almacenarlos en 2 nuevas listas. En la Figura 3-9, podemos ver representado un ejemplo de relación de los puntos característicos de la imagen infrarrojos con la imagen a color.

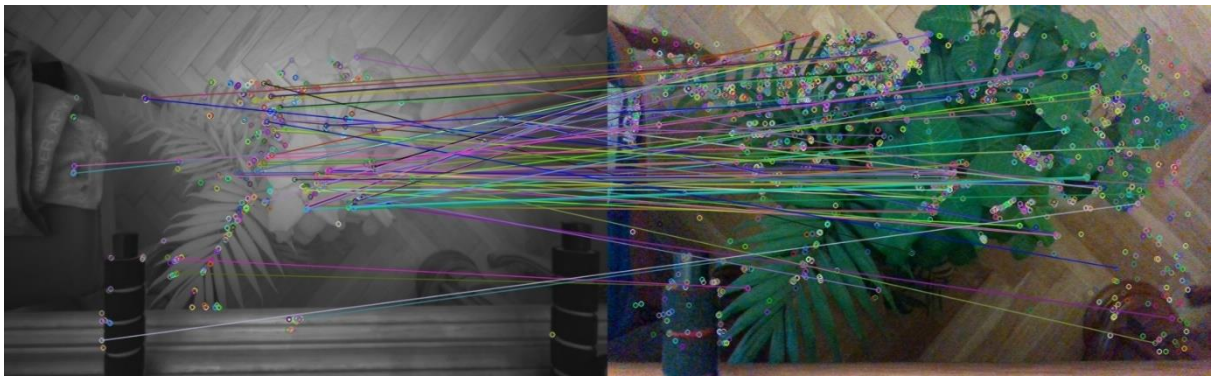


Figura 3-9: Relación de puntos característicos de la imagen infrarrojos (izquierda) con la imagen RGB (derecha)

Si nos fijamos detenidamente, hay puntos característicos de una imagen que no han encontrado su representación en la opuesta. Por tanto, será necesario ordenar los puntos característicos por mayor porcentaje de coincidencia, para posteriormente descartar aquellos que se encuentren al final de la lista (normalmente se descarta un 10-20% de las medidas obtenidas). Una vez finalizado este paso, ya sólo queda utilizar los puntos característicos comunes para calcular la matriz de homografía.

En la Figura 3-10, podemos ver el resultado del ejemplo con el que hemos trabajado durante toda esta explicación. En ella, podemos ver que el resultado no es exacto pero que se ha conseguido una gran mejora en la alineación de las imágenes.

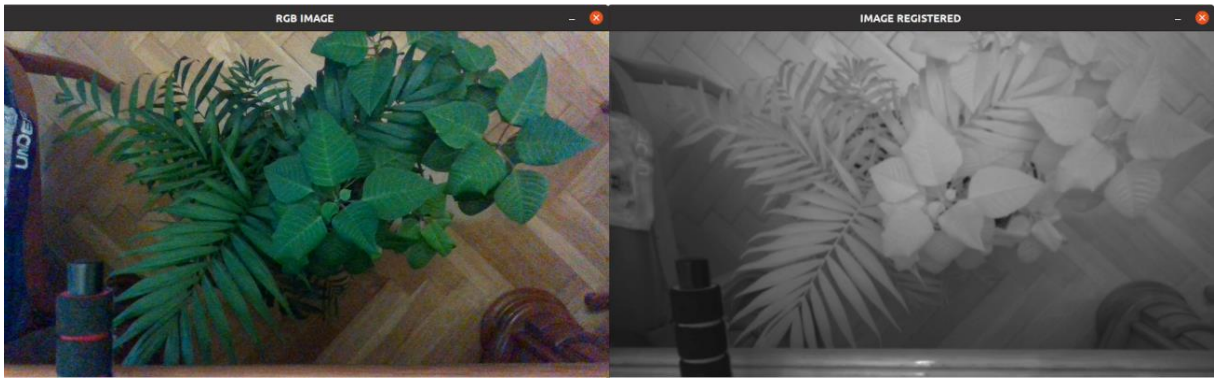


Figura 3-10: Imágenes RGB (izquierda) e infrarrojos (derecha) de la cámara alineadas

3.3.2 Recorte de imágenes

Tras fijarnos en los resultados del registro de las imágenes, podemos ver que una de las patas del tren de aterrizaje del dron actúa como obstáculo en la imagen. La solución más sencilla al encontrarse siempre en la misma posición es cortar un trozo del lado izquierdo de la imagen en ambas fotos, para obtener una foto con la información que necesitamos.

Se conoce como recorte de imagen al proceso mediante el cual se selecciona un rectángulo de píxeles, para descartar los píxeles restantes que queden fuera de la zona delimitada (Knott, 2021). Existen otras soluciones para este tipo de problemas que inicialmente detectan el objeto específico, calculan el área que ocupa para eliminarlo de la imagen y después rellenan el hueco que se ha creado con la información de los píxeles circundantes. Esta solución requiere de una implementación compleja utilizando inteligencia artificial para conseguir una buena detección del objeto, además que el hueco rellenado sería información inventada que no aporta nada a nuestro diagnóstico del cultivo (con información inventada nos referimos a que se rellenan los píxeles eliminados con la información de los píxeles colindantes). La segmentación de una imagen sólo necesita como información el rango de filas y de columnas que se desean cortar, por lo que, al encontrarse la cámara fija siempre en la misma posición, se puede determinar un valor constante. En la Figura 3-11, podemos apreciar el recorte del tren de aterrizaje aplicado en ambas imágenes.

Figura 3-11: Imágenes a color (arriba) e infrarrojos (abajo) con recorte aplicado

3.3.3 Aplicación de máscara a una imagen (ExG)

Una vez hemos alineado ambas imágenes y recortado el tren de aterrizaje, tenemos todo preparado para poder aplicar el índice de vegetación. En la Figura 3-12, podemos observar un ejemplo de imagen con índice de vegetación NDVI aplicado. Este índice ha sido calculado utilizando las imágenes de la Figura 3-10, en las que podemos apreciar que gran parte de la imagen no contiene vegetación y se está calculando el índice en toda ella. Por tanto, necesitaremos de alguna forma filtrar ciertas zonas de la imagen para que se vea de forma más clara las zonas con índice aplicado que realmente existe vegetación.

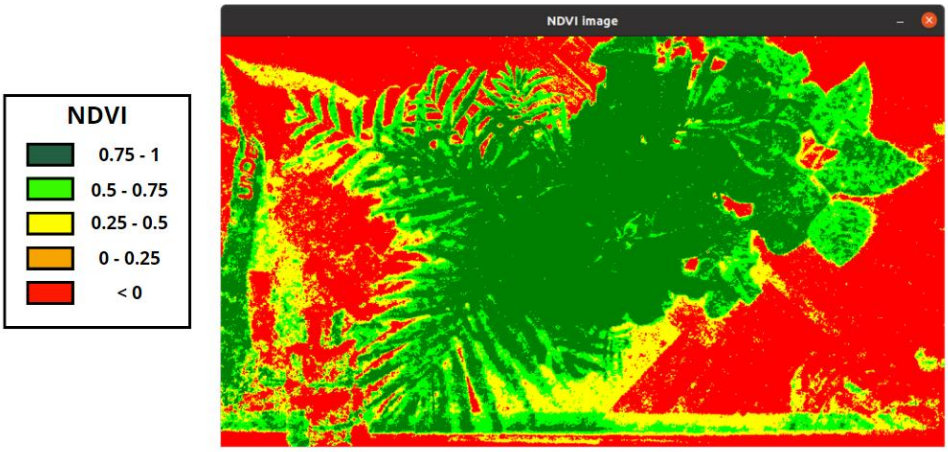


Figura 3-12: imagen con índice NDVI aplicado

ExG es un índice de detección de vegetación que se encarga de medir el verdor de cada uno de los píxeles de la imagen, utilizando sólo una imagen RGB. El resultado del índice es una imagen en escala de grises en la que el valor de cada píxel se ve representado por la siguiente fórmula (BioMed Central, 2018):

$$ExG = 2 \cdot g - r - b$$

Donde:

$$g = \frac{G_n}{(R_n + G_n + B_n)} \qquad r = \frac{R_n}{(R_n + G_n + B_n)} \qquad b = \frac{B_n}{(R_n + G_n + B_n)}$$

Rn = Valor del canal rojo de la imagen RGB normalizado [0-1]

Gn = Valor del canal verde de la imagen RGB normalizado [0-1]

Bn = Valor del canal azul de la imagen RGB normalizado [0-1]

Pero para poder filtrar ciertos píxeles de una imagen necesitamos generar una máscara. Una máscara en tratamiento de imagen es una matriz binaria del tamaño de la imagen original, en la que los valores de los píxeles son 0 o 1. Si el valor del píxel es 0 (negro), significa que ese píxel va a ser descartado en la imagen original, mientras que si el valor del píxel es 1 (blanco), ese píxel se mantendrá en la imagen (Martínez J. , 2022). Por tanto, necesitamos binarizar la imagen del índice ExG para poder crear una máscara que aplicar a nuestra imagen con el índice aplicado.

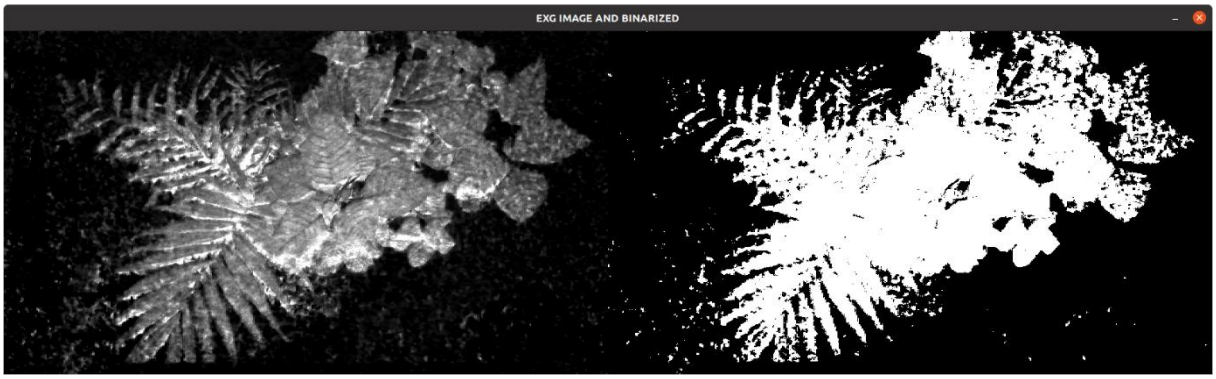


Figura 3-13:Imagen con índice ExG (izquierda) e imagen ExG binarizada (derecha)

El método de Otsu, creado por Nobuyuki Otsu, consiste en un proceso por el que se binariza una imagen de forma automática. El algoritmo se encarga de buscar el umbral que maximiza la varianza inter-clase, dando como resultado un valor constante entre 0-255, el cual se encargará de dividir los valores existentes entre 2 clases (Wikipedia, 2024). Es decir, la imagen se binariza siguiendo el siguiente criterio:

$$Clase\ 1 \rightarrow (Valor\ del\ píxel \geq\ valor\ de\ Otsu) \rightarrow\ 1\ (blanco)$$

$$Clase\ 2 \rightarrow (Valor\ del\ píxel\ <\ valor\ de\ Otsu) \rightarrow\ 0\ (negro)$$

En la Figura 3-13, podemos apreciar el índice ExG aplicado a la imagen RGB de la Figura 3-10 a la izquierda, junto con la imagen resultado binarizada a la derecha. Si nos fijamos en la imagen binarizada, existe algo de ruido dentro de la máscara: existen píxeles detectados como vegetación que son falsos positivos, además de ciertas secciones de las hojas que no han sido detectadas. Existen una serie de transformaciones morfológicas para imágenes binarias que buscan eliminar el ruido dentro de estas (OpenCV Docs, 2024). Los principales operadores morfológicos son:

- Erosión: Elimina los píxeles sobrantes dentro de los diferentes objetos o regiones de píxeles blancos detectados dentro de la imagen. Utiliza un kernel (matriz cuadrada de tamaño mínimo 3x3), para determinar qué píxeles blancos se considera que deben de ser eliminados en la imagen. Un píxel se considerará de valor 1(blanco) si todos los píxeles circundantes que ocupan el tamaño del kernel son 1, mientras que en caso contrario el píxel es erosionado (valor 0). Además, se añade un parámetro de iteraciones para seleccionar el número de veces que se aplica la operación: a más iteraciones, mayor será la erosión y viceversa. En la Figura 3-14 (OpenCV Docs, 2024), podemos apreciar un ejemplo de erosión aplicado a una imagen binaria.

- Dilatación: Operación opuesta a la erosión, mostrada en la Figura 3-15 (OpenCV Docs, 2024). Se considerará un píxel valor 1 si al menos uno de los píxeles que contiene el kernel vale 1. Se encarga de aumentar la región blanca denotada por los diferentes objetos en la imagen binaria. Utiliza también un parámetro de iteraciones para decidir cuántas veces se repite el proceso.

Figura 3-14: Erosión de imagen binaria (OpenCV Docs, 2024)

Figura 3-15: Dilatación de una imagen binaria (OpenCV Docs, 2024)

Una vez conocidas los 2 principales operadores, existen 2 variantes que podemos utilizar en nuestro trabajo para eliminar el ruido de las máscaras:

Apertura: Consiste en el proceso de primero erosionar la imagen binaria, para después calcular la dilatación de la imagen erosionada. Esta variante se encarga de eliminar el ruido externo a los objetos de la imagen (eliminar pequeñas zonas blancas aisladas). Un ejemplo de esta operación se encuentra representado en la Figura 3-17 (OpenCV Docs, 2024).

1. Cierre: Consiste en realizar la dilatación de la imagen binaria, para posteriormente erosionar la imagen dilatada. Esta variante se encarga de eliminar el ruido interno de los objetos de la imagen binaria (rellenar pequeños huecos negros). En la Figura 3-16 (OpenCV Docs, 2024), podemos apreciar un ejemplo de la operación aplicada.

Figura 3-17:Apertura de una imagen binaria (OpenCV Docs, 2024)

Figura 3-16: Cierre de una imagen binaria (OpenCV Docs, 2024)

Siguiendo las transformaciones morfológicas explicadas y viendo el resultado de la imagen binaria en la Figura 3-13, si aplicamos a nuestra máscara primero una operación de cierre seguida de una apertura, obtenemos una máscara que engloba mejor la zona delimitada por la vegetación. Para estas 2 operaciones se ha utilizado un kernel de 9x9 con sus valores 1, junto a un número de iteraciones igual a 3. Ahora que ya tenemos nuestra máscara de la imagen original creada, el siguiente paso es calcular la imagen resultado utilizando las imágenes del índice NDVI y la máscara. Para filtrar la imagen con ayuda de la máscara utilizaremos la operación lógica AND entre ambas imágenes. La operación binaria AND es verdadera sólo si ambos píxeles tienen un valor mayor que 0, es decir, que utilizando la ayuda de la máscara podemos descartar los píxeles que no queremos analizar (Rosebrock, 2021). En la Figura 3-18, se encuentra representado el resultado del índice NDVI con la máscara aplicada. En la imagen superior vemos el índice aplicado junto a una máscara que no ha recibido trasformaciones morfológicas, mientras que en la imagen inferior podemos observar el índice aplicado junto a la máscara que se ha realizado primero una operación de cierre seguida de una apertura. Según los resultados de la figura, podemos apreciar que las transformaciones morfológicas ayudan a englobar mejor en nuestra máscara la zona denotada por vegetación, dando una mayor legibilidad a nuestro análisis del terreno de cultivo. En la sección de Resultados, explicaremos con las imágenes tomadas durante el vuelo todo el tratamiento de imagen, mostrando los pasos intermedios durante el proceso completo.

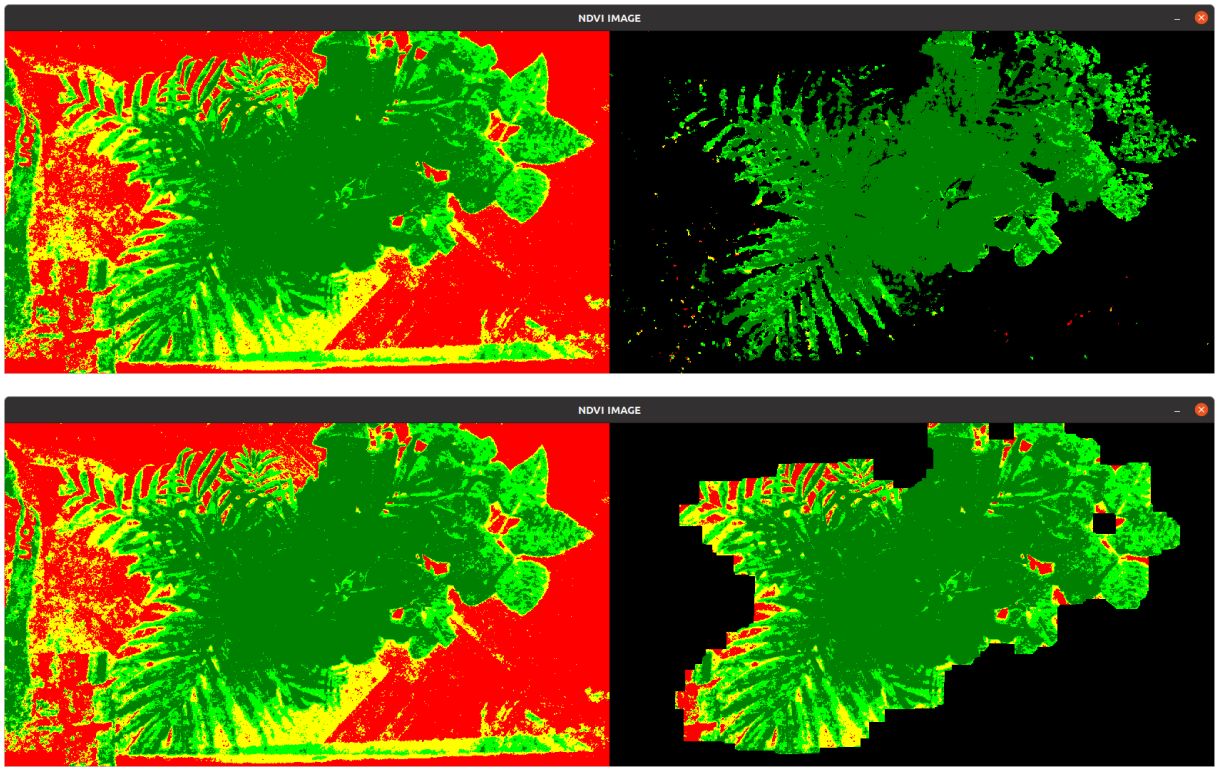


Figura 3-18: Índice NDVI con máscara sin transformaciones morfológicas (arriba) e índice NDVI con máscara con transformaciones morfológicas aplicadas (abajo)

4 Pruebas en vuelo

Consiste en un conjunto de ensayos en los que se busca comprobar que el proyecto se puede traspasar de la teoría al mundo real. Este proceso es muy importante porque es donde se descubren las desviaciones que existen, corrigiéndolas e implementándolas en nuestro trabajo de forma correcta. Para nuestro proyecto, hay cosas como el funcionamiento de la Raspberry Pi 4 o el procesamiento de imagen, que se pueden probar sin necesidad de volar la aeronave. Sin embargo, cualquier aspecto relacionado con la precisión de la ejecución de la misión del dron, necesitará ser probado en vuelo. Además, al haber añadido varios componentes extras al kit inicial del vehículo, habrá que comprobar que tanto la maniobrabilidad como el consumo de batería por el aumento de peso no se ven afectados gravemente. Por tanto, lo primero será planificar el conjunto de pruebas que vamos a realizar, en las que se va adaptando la teoría al mundo real de forma progresiva.

En esta sección, se detallan: las diferentes pruebas realizadas, qué busca comprobar y verificar cada una de ellas, y los ajustes realizados post-ensayo en caso de que se detecte una desviación de la teoría respecto del mundo real. Las primeras pruebas consistirán en probar que la calibración del dron ha conseguido que la aeronave funcione de la manera adecuada, además de comprobar que la comunicación estación de tierra-dron funciona correctamente a la hora de enviar/recibir, tanto comandos como datos recopilados a tiempo real durante la misión. Las siguientes pruebas consistirán en añadir los componentes externos al vehículo, para ir comprobando el comportamiento de este junto con dispositivos como la cámara o la Raspberry Pi 4. Por último y una vez hayamos conseguido que todos los componentes funcionen correctamente, las pruebas que nos quedarán por realizar serán aquellas que buscan mejorar los resultados obtenidos durante la misión. Durante todo este conjunto de pruebas, se utilizará QGroundControl para supervisar el estado y el transcurso de las misiones. Además, todas las

pruebas se realizarán en el aeródromo de aeromodelismo de la Universidad Rey Juan Carlos en Fuenlabrada. De esta forma, podemos realizar todas nuestras pruebas en una zona segura y poco transitada, para evitar cualquier daño causado en caso de fallo durante el vuelo.

4.1 Pruebas en vuelo sin componentes externos

Una vez realizada la calibración del dron, podemos empezar a realizar nuestras primeras pruebas en vuelo. Estas primeras pruebas consistirán básicamente en comprobar que todo componente procedente del kit de nuestra aeronave funciona adecuadamente. Nuestra primera prueba consistirá en realizar una misión de corta duración, en la que manejaremos el dron manualmente mediante un control remoto. En esta prueba, realizaremos una serie de trayectorias donde comprobaremos que la estabilidad del vehículo y la maniobrabilidad se comportan de forma esperada, es decir, que no haya ningún movimiento impreciso que llegue a realizar una trayectoria peligrosa para la integridad del vehículo.

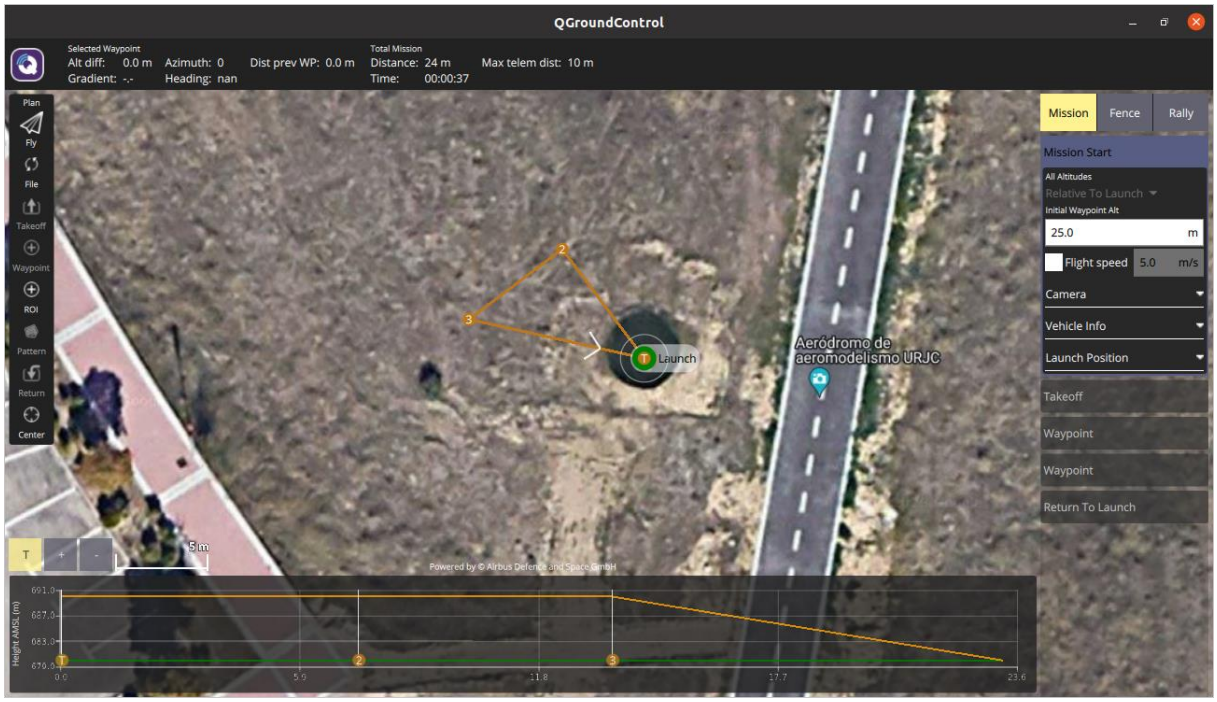


Figura 4-1: Plan de misión con trayectorias manualmente definidas visto desde QGroundControl

Una vez comprobado que todo funciona correctamente en esta primera prueba, la siguiente prueba que vamos a realizar consiste en realizar una misión en la que el dron vuela automáticamente a unos puntos determinados por un plan creado previamente por el operario

de vuelo. Esta prueba nos ayudará a verificar para futuros ensayos que es posible automatizar la misión, mientras el operario supervisa el estado durante el transcurso de esta. En la Figura 4-1, podemos apreciar dicho plan representado en QGroundControl, donde podemos apreciar: el comienzo de la misión, las trayectorias que va a realizar, los puntos que va a visitar y dónde va a finalizar (en este caso, la misión vuelve al punto inicial). Además, se pueden ajustar otros parámetros como: la velocidad a la que viaja a cada waypoint o la altura a la que se sitúa en él. En el caso de esta primera misión, estos 2 parámetros pueden dejarse con los valores predeterminados dado que independientemente de los valores que asignemos, no influirán en nuestro principal objetivo que es comprobar que el sistema automático de misiones funciona.

Después de haber realizar esta prueba y asegurarnos que podemos realizar misiones automatizadas, el siguiente paso es comprobar que estas misiones se pueden abortar en cualquier momento. Con abortar la misión nos referimos a que el operario sea capaz manualmente desde tierra (ya sea vía QGroundControl o mediante el control remoto), forzar que el dron vuelva al inicio antes de que la misión haya finalizado. Esto puede sernos muy útil en ciertas situaciones: la batería del dron se está agotando y puede caerse antes de finalizar la misión, las condiciones climáticas han cambiado repentinamente (rachas de viento fuerte o lluvia) y pueden dañarse algunas piezas, o ya hemos conseguido los objetivos de la prueba y queremos ahorrar batería en vez de completar la misión. En esta tercera prueba, utilizaremos el mismo plan representado en la Figura 4-1. La única diferencia es que, durante el transcurso de la misión, activaremos el modo Return para que el dron detecte que el estado de la misión ha cambiado y de esta forma sepa que se le ha comunicado desde tierra que debe volver al inicio. La asignación de los modos de vuelo al control remoto se realiza desde QGroundControl. En el apartado 2.2.6, se puede apreciar con mayor detalle los modos utilizados durante este trabajo junto con su función. Una vez hayamos completado la tercera prueba y sepamos con certeza que abortar la misión manualmente funciona, estamos preparados para empezar a hacer las pruebas con los componentes externos acoplados a la aeronave. Debido a que estos componentes externos pueden afectar al funcionamiento del vehículo, esta tercera prueba nos será de gran utilidad para el resto de comprobaciones.

4.2 Pruebas en vuelo con componentes externos

Con las primeras pruebas realizadas, hemos comprobado que la funcionalidad básica del dron y su comunicación con QGroundControl se comporta de la forma que esperábamos. Sin embargo, todavía no hemos añadido las piezas que necesitamos para realizar la misión en la que recopilaremos los datos necesarios para el post-procesamiento. En este apartado, vamos a realizar una serie de pruebas que verifiquen, tanto que la aeronave puede volar con el peso añadido de los nuevos componentes, como que los componentes realizan su función: en el caso de la cámara que sea capaz de tomar imágenes, mientras que en el caso de la Raspberry Pi sea capaz de recopilar toda la información necesaria durante la misión. Además, comprobaremos las variaciones en el gasto de batería tras añadir el peso de todos los componentes externos a la aeronave.

El primer paso es montar la cámara, la batería portátil y la Raspberry Pi 4 en la aeronave, tal y como se muestra en la Figura 3-5. Se ha añadido a la Raspberry una carcasa de aluminio protectora, para absorber el golpe y minimizar los daños en caso de que se produjera un fallo durante la misión y el vehículo se estrellase. Como ya hemos mencionado, la cámara irá acoplada a la pieza 3D mediante un tornillo 1/4-20 UNC que será sujeción suficiente durante todo el recorrido. En cuanto a la batería portátil, se sujetará a la plataforma superior del dron mediante cinta adhesiva de doble cara y bridas, con la Raspberry Pi situada encima de esta y con el mismo método de sujeción. Para la primera prueba, acoplaremos sólo la pieza 3D junto con la cámara atornillada para comprobar cómo afecta el aumento de peso de este dispositivo a la maniobrabilidad y el gasto de batería del vehículo. Durante la prueba, se realizará el mismo plan que aparece en la Figura 4-1 para analizar los cambios producidos en estos parámetros. Dado que el peso de la cámara junto con la pieza 3D se estima en ≈ 100g, lo que supone un aumento del 6.66% del peso disponible para la carga de pago (la carga de pago máxima son 1500g), se estima que este aumento de peso no debería de afectar notablemente en la maniobrabilidad y gasto de batería de la aeronave (Holybro, 2023).

Tras finalizar la prueba satisfactoriamente, el dron realizó el recorrido completo de la misión en aproximadamente 50 segundos. Durante el desarrollo de esta, no se detectó ninguna anomalía en los movimientos del vehículo ni un desvío en las trayectorias realizadas respecto a la misión sin componentes externos. En cuanto al consumo de batería, la batería que estamos usando en nuestro caso es una batería LiPo 5000mAh de 4 celdas conectadas en serie (4S) con

una velocidad de descarga 30C. El voltaje con la batería completamente cargada medido previamente al vuelo es 16.8V (4.2V por celda), mientras que el voltaje medido tras realizar la misión es de 16.6V. Este consumo de batería es prácticamente el mismo que se produjo en las misiones sin componentes externos, pero todavía no hemos añadido la batería portátil y la Raspberry Pi al vehículo, que son dispositivos más pesados que la cámara.

Por tanto, nuestra siguiente prueba es comprobar cómo afecta el peso de la batería portátil y la Raspberry Pi al dron durante el vuelo. Cabe estacar que al tener QGroundControl conectado al dron mediante telemetría, en caso de detectar que el voltaje de la batería cae drásticamente podemos abortar la misión manualmente. Además, para que el operario no tenga que estar constantemente pendiente del consumo de la batería, en el apartado Safety de la parte de calibración en QGroundControl, hemos asignado unos porcentajes de batería críticos en los que el dron volverá automáticamente al inicio (25% de batería) o aterrizará donde sea que se encuentre en caso de nivel de emergencia (10% de batería). Una vez acoplada la batería portátil y la Raspberry como hemos explicado previamente, el siguiente paso es realizar el plan de la Figura 4-1 de nuevo, para analizar las consecuencias de añadir un peso extra considerable a la carga de pago: el peso de la batería portátil está estimado en ≈ 200g (Amazon, 2024), mientras que el peso de la Raspberry Pi junto con la carcasa de aluminio es ≈ 80g. Esto convierte el aumento de peso en:

$$P_{total} = P_{cámara} + P_{batería} + P_{Raspberry} = 100_g + 200_g + 80_g = 380_g$$

Este peso ya supone un 25.33% del peso máximo de la carga de pago, lo que supone un aumento considerable respecto sólo al peso de la cámara. Después de haber realizado la prueba con todos los componentes externos acoplados, el resultado ha sido positivo ya que el dron ha conseguido completar la misión y no se ha producido ninguna anomalía en cualquiera de sus trayectorias. En cuanto al consumo de batería, se ha producido como ya esperábamos un aumento midiendo un voltaje de 16.4V al finalizar la misión. Estos resultados nos permiten estimar que el tiempo que nuestra aeronave puede estar realizando una misión de este tipo es de 8-10 minutos aproximadamente (se considera 14V como voltaje mínimo), ya que, si modificamos algunos parámetros como la velocidad de la misión o el tiempo de espera en cada waypoint, el tiempo de vuelo puede verse alterado.

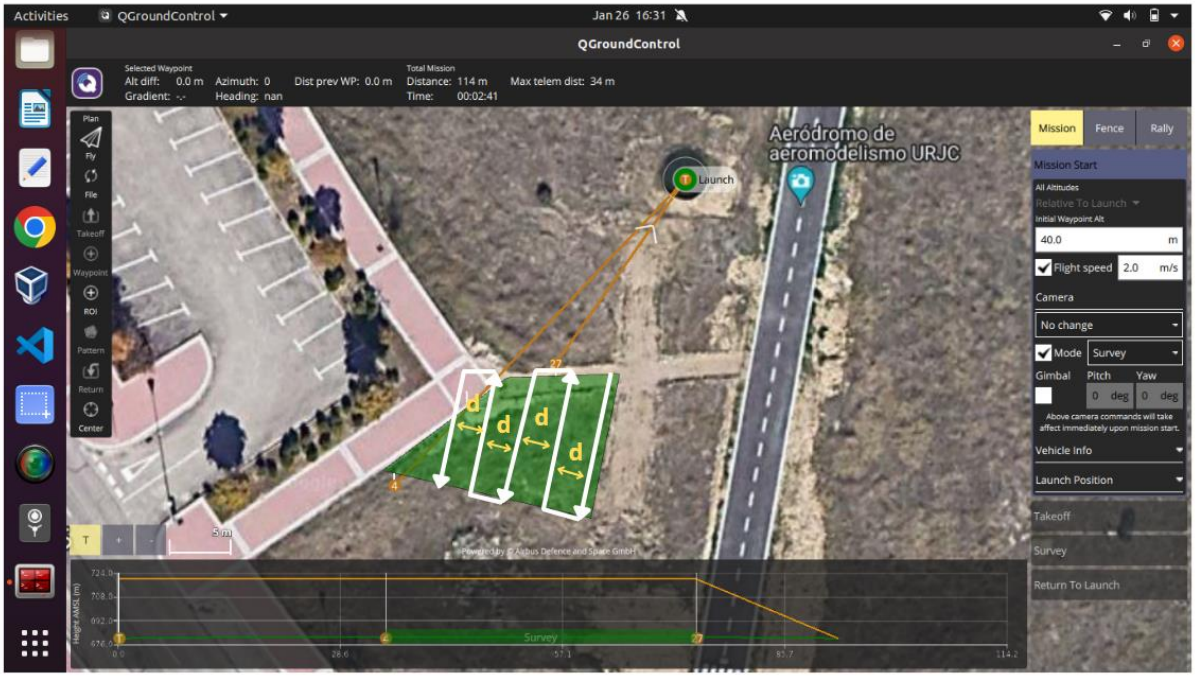


Figura 4-2: Plan de misión con trayectoria de cobertura visto desde QGroundControl

Una vez hemos obtenido los resultados esperados con todos los componentes externos acoplados al dron, podemos empezar a hacer pruebas que sean simulacros de la misión que cumple el objetivo de nuestro proyecto. Esta misión consiste en realizar un plan automatizado en el que el dron recorra una zona de un terreno con vegetación, mientras que periódicamente se van guardando imágenes tomadas por la cámara en tiempo real y algunos datos: posición, orientación y nombre de las imágenes guardadas. Para el caso de esta misión, vamos a desarrollar un plan diferente al presentado en la Figura 4-1, donde además modificaremos la velocidad de vuelo y la altura del dron durante la misión, para comprobar con qué parámetros obtenemos mejor calidad de imágenes. En la Figura 4-2, podemos ver el plan realizado desde QGroundControl donde a simple vista ya podemos apreciar la principal diferencia respecto del plan de la Figura 4-1: el plan ya no recorre unos puntos concretos asignados manualmente por el operador, sino que ahora existe una zona concreta designada por un polígono de color verde, la cual será recorrida por la aeronave durante la misión realizando un algoritmo de trayectoria de cobertura. Desde QGroundControl, es posible ajustar los parámetros de la trayectoria: la altura a la que recorrerá la zona y la distancia en metros para realizar los giros del zigzag. En este caso, la distancia de los giros, representada con una flecha amarilla, se establecerá en 1 metro mientras que la altura de vuelo se irá modificando según se vayan haciendo pruebas para comprobar la calidad de las imágenes. Para finalizar la misión, el vehículo volverá como siempre al punto donde se inició el plan.

Con este plan se realizaron 3 misiones diferentes, donde la diferencia entre ellas fue la altura de vuelo: 10, 25 y 40 metros. En las 3 misiones, la duración aproximada del vehículo en el aire se estima en 3 minutos aproximadamente, midiendo un voltaje de 15.9V al finalizar (siempre se realizó la misión con una batería completamente cargada: 16.8V). Las misiones recopilaron cada una de ellas 70 fotos: 35 fotos RGB y 35 fotos infrarrojos, además de un fichero CSV con los datos correspondientes a cada una de ellas. De todas estas fotos, algunas de ellas serán descartadas debido a que, al tratarse de una captura periódica, existen fotos: tomadas cuando el dron todavía no ha despegado, cuando el dron está ascendiendo a la altura del vuelo y cuando está descendiendo para aterrizar.

Figura 4-3: Fotos clasificadas según la altura a la que fueron tomadas.

En la Figura 4-3, podemos apreciar un ejemplo de la calidad y visión panorámica de las fotos dependiendo de la altura a la que fueron tomadas. Como la mayoría de índices de detección de vegetación han sido desarrollados con el objetivo de ser utilizados con imágenes satelitales, obtendremos mejores resultados si utilizamos las imágenes tomadas a mayor altura. Con estas 3 últimas misiones realizadas, ya hemos recopilado una serie de datos con los que empezar el proceso de post procesamiento y así obtener un análisis del terreno. En la siguiente sección, explicaremos el proceso a seguir con todos estos datos para obtener un diagnóstico de la salud de la vegetación en la zona recorrida durante el vuelo

5 Resultados

Tras haber realizado varias pruebas en vuelo, hemos obtenido una serie de datos procedentes de diferentes misiones con los que podemos realizar un diagnóstico del terreno de cultivo sobrevolado. Para ello, dividiremos el post-procesamientos en varias fases donde explicaremos las operaciones utilizadas para conseguir los resultados del análisis. Todo este post-procesamiento se realizará utilizando un ejecutable desarrollado con C++, utilizando tanto las librerías estándar del lenguaje como la librería de tratamiento de imágenes OpenCV. Para saber más sobre la implementación del código, **véase este enlace al repositorio GitHub** (GitHub, 2024)**.**

La primera fase se encargará de leer los datos recopilados dada una estructura predefinida, además de preparar las imágenes para el cálculo del índice. Esta estructura de almacenamiento de los datos recopilados será explicada al comienzo de esta fase. La segunda fase consistirá en crear las imágenes con los índices aplicados respecto de la imagen RGB e infrarrojos de cada lugar donde se capturaron las imágenes. En esta fase, se aplicará una máscara para filtrar los píxeles que no contenga vegetación y así poder depurar la imagen del índice, mostrando únicamente las zonas de interés para realizar el diagnóstico. En la tercera y última fase, se realizará el análisis de la imagen para detectar los puntos críticos donde el cultivo requerirá de intervención por parte del humano. Para ello, se dividirá la imagen en secciones utilizando una rejilla que nos permita realizar un diagnóstico en zonas más concretas. Tras detectar los puntos críticos, calcularemos la posición de ellos en el plano 3D y los convertiremos en coordenadas geográficas, para poder ser geolocalizados en la Tierra. Por último, pintaremos dichos puntos en la imagen junto con sus coordenadas y concatenaremos la imagen RGB con la imagen del índice aplicado, para poder tener un diagnóstico completo de la zona en una sola imagen.

En esta sección y al igual que en Procesamiento de imagen, se añadirán figuras que expliquen todos los pasos intermedios de este proceso de forma gráfica, para un mayor entendimiento de las operaciones realizadas.

5.1 Primera fase: Lectura del directorio de la misión

Mientras el dron realiza la misión, como ya sabemos existe un programa en ejecución en la RPi4 que se encarga de capturar las imágenes, además de almacenar datos de interés en un fichero. Todos estos elementos recopilados en una misión, se guardan en un directorio con una estructura concreta. El nombre del directorio sigue el siguiente formato: mission_yyyy-mm-dd_hh-mm-ss, donde yyyy-mm-dd es la fecha de la misión y hh-mm-ss la hora a la que comenzó. Dentro de este directorio podemos encontrar: las imágenes RGB, las imágenes infrarrojas y un fichero de metadatos. Este fichero de metadatos guarda información adicional de cada lugar donde se capturan imágenes, siguiendo el siguiente esquema:

Cámara:

- rgb_img_name: nombre de la imagen RGB para los datos asociados.
- ir_img_name: nombre de la imagen IR para los datos asociados.

Odometría:

- x(m): Posición en el eje X respecto al sistema NED desde el punto de inicio.
- y(m): Posición en el eje Y respecto al sistema NED desde el punto de inicio.
- z(m): Posición en el eje Z respecto al sistema NED desde el punto de inicio.
- roll(rad): Alabeo de la aeronave calculado desde el quaternión de orientación.
- pitch(rad): cabeceo de la aeronave calculado desde el quaternión de orientación.
- yaw(rad): Guiñada de la aeronave calculado desde el quaternión de orientación.

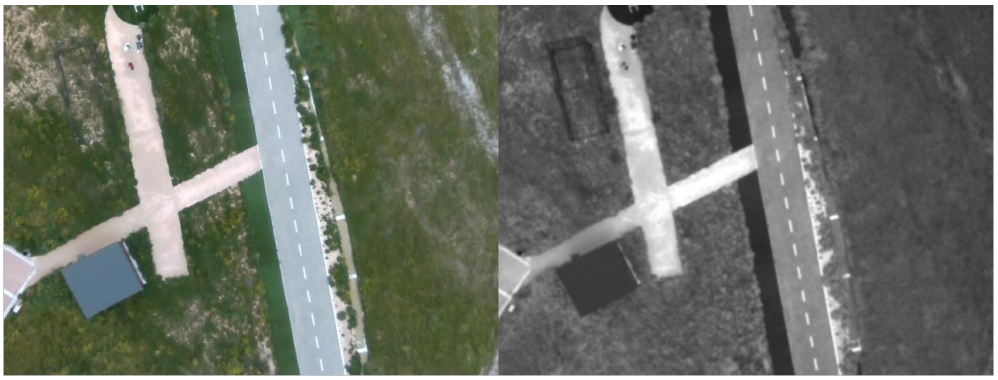
GPS:

- latitude(º): Latitud del punto en la Tierra.
- longitude(º): Longitud del punto en la Tierra.
- altitude(m): Altitud del punto en la Tierra sobre el nivel del mar (AMSL).

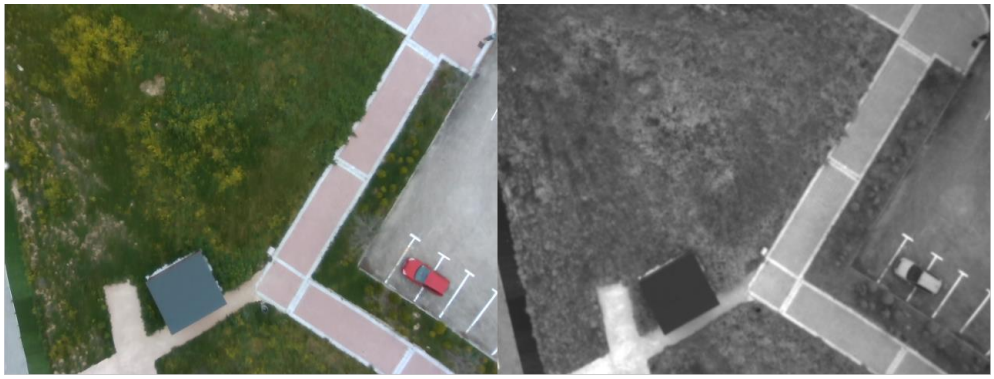
Figura 5-1: Estructura del directorio autogenerado por la misión

En la Figura 5-1, podemos observar un esquema que resumen la estructura del directorio de la misión. La ruta a este directorio será pasada como argumento al programa que se encarga de realizar el diagnóstico, el cual es capaz de interpretar este formato y almacenar los datos en memoria. En el proceso de almacenaje de los datos, cuando se lean las imágenes se les aplicarán 2 operaciones: primero se registrará la imagen infrarroja respecto de la imagen RGB (Figura 3-10), para posteriormente cortar la pata del tren de aterrizaje de ambas imágenes (Figura 3-11).

Una vez hemos almacenado todos los datos de la misión en memoria local, el siguiente paso es procesar todos estos datos juntos parcela a parcela, obteniendo un diagnóstico separado de cada una de las zonas capturadas durante el vuelo. En la Figura 5-2, se pueden apreciar las imágenes que vamos a utilizar de ejemplo para explicar el resto de fases. Cabe destacar que este mismo proceso se realizará con todas las imágenes tomadas, pero en el caso de este documento, se mostrarán todos los pasos intermedios con las imágenes que aparecen en la figura, para un mejor entendimiento.



a)



b)

Figura 5-2: Imágenes de ejemplo para el post-procesamiento

5.2 Segunda fase: Creación de imágenes resultado

Una vez estructurada la información procedente del directorio de la misión dentro del programa, podemos empezar a generar las imágenes de las parcelas de cultivo con el índice de vegetación aplicado. Lo primero es crear una máscara de la imagen a color en la que filtremos aquellos píxeles que se pueden considerar como zonas sin vegetación. Para ello: utilizaremos el índice ExG para obtener una imagen a escala de grises en función del verdor de cada píxel, binarizaremos la imagen con el índice aplicado utilizando el método automático de Otsu, y aplicaremos transformaciones morfológicas para obtener una máscara que diferencie los píxeles que pueden tener vegetación de las zonas que no queremos analizar. Todo este proceso está

explicado detalladamente en el subapartado 3.3.3, además de aportar figuras que explican cada uno de los pasos seguidos.

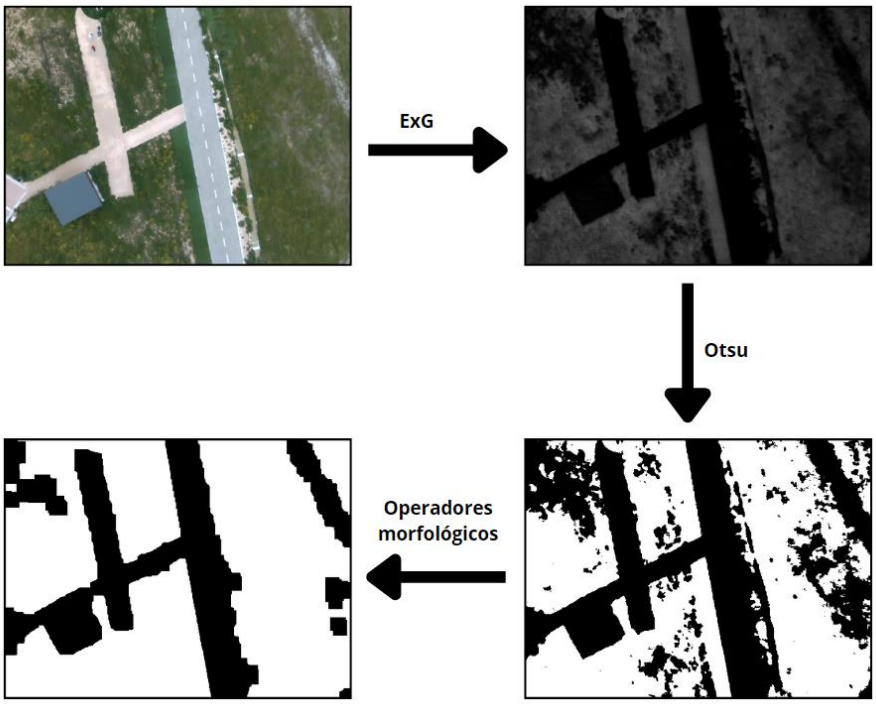


Figura 5-3: Proceso de creación de máscara binaria a partir de imagen RGB

En la Figura 5-3, podemos apreciar el proceso explicado previamente aplicado a la imagen RGB tomada por el dron durante una misión. Todos estos píxeles negros que aparecen posteriormente en las imágenes con índice aplicado, no serán tenidos en cuenta a la hora de realizar el diagnóstico del cultivo. Esto se debe a que la máscara indica que todos estos píxeles carecen de vegetación, por lo que no son de interés para el análisis del terreno.

Las transformaciones morfológicas aplicadas para obtener la máscara mejorada son: una operación de cierre iterada 2 veces y una operación de apertura iterada 1 vez, ambas utilizando un kernel formado por una matriz de valores 1 de tamaño 9x9. Ya obtenida la máscara de la imagen RGB, empezaremos el proceso de creación de imágenes con el índice de vegetación aplicado. Los índices que utilizaremos en el caso de nuestro trabajo son: NDVI, EVI y SAVI, cuyas fórmulas se encuentran explicadas detalladamente en la sección 2.1: Técnicas de detección de vegetación. Para generar la imagen resultado, se aplica la fórmula a cada uno de los píxeles de la imagen obteniendo un análisis del terreno de cultivo capturado por la cámara.

Esta imagen nos permite luego realizar una interpretación de los resultados obtenidos, para poder señalizar las zonas del cultivo que requieren de intervención humana. Tras obtener las imágenes con el índice de vegetación aplicado, el último paso de esta fase será utilizar el operador lógico AND entre las imágenes del índice y la máscara, con el objetivo de filtrar aquellas zonas que carecen de vegetación y son innecesarias para nuestro diagnóstico.

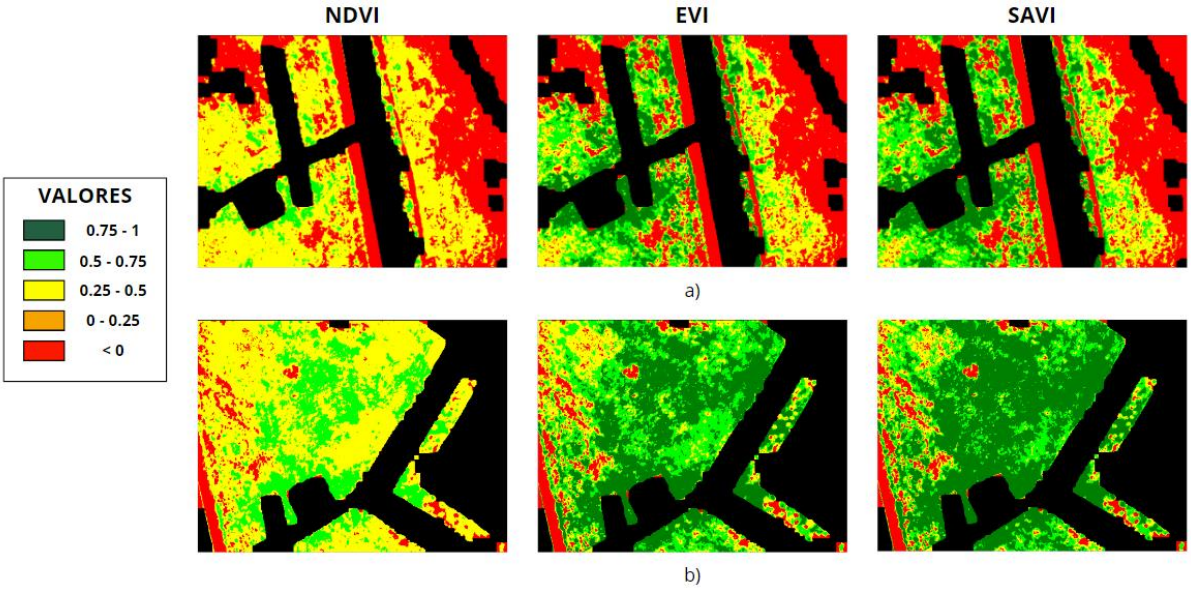


Figura 5-4: Imágenes ejemplo con los índices de vegetación aplicados

En la Figura 5-4, podemos observar las imágenes de índice de vegetación generadas con NDVI, EVI y SAVI en orden de izquierda a derecha. Previamente, vamos a desglosar los valores utilizados en las constantes de los índices EVI y SAVI. Los valores del índice EVI han sido reajustados respecto de los valores utilizados por el gobierno estadounidense con sus imágenes satelitales (USGS, 2024). En cuanto a la constante L del índice SAVI, ha sido reajustada manualmente haciendo ensayos con diferentes valores hasta obtener un resultado similar al índice EVI. Finalmente, Los valores utilizados en este proyecto son:

- EVI $\rightarrow G = 4.5 \quad C1 = 6.0 \quad C2 = 4.5 \quad L = 1.0$
- SAVI $\rightarrow L = 2.0$

A simple vista, podemos observar que todos los índices detectan de forma similar las zonas donde la vegetación es pobre o inexistente, mostrando pequeñas zonas de color rojo. Sin embargo, los resultados de los índices EVI y SAVI, cuyo objetivo es mejorar los resultados obtenidos con el índice NDVI, muestran un diagnóstico del cultivo más preciso respecto a la salud de la vegetación: aparecen con mayor frecuencia los diferentes tonos verdes, para indicar

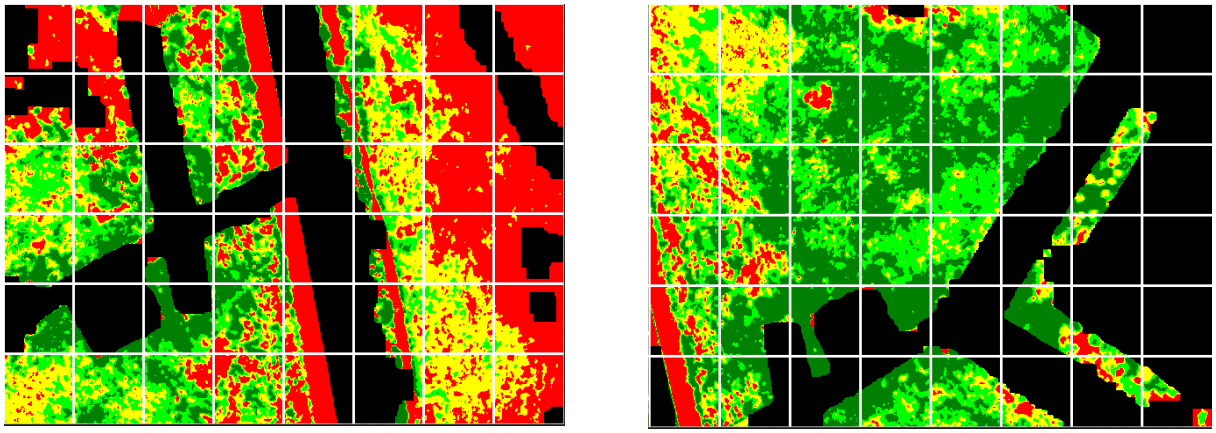
aquellas zonas que la vegetación tiene un estado de salud adecuado respecto de las zonas amarillas, cuya salud está cerca de ser peligrosa. En la imagen NDVI, se pierden prácticamente la mayoría de los detalles de la salud de las diferentes parcelas, simplificando el diagnóstico del terreno de cultivo a un estado de salud unificado sobre toda su superficie. Por tanto, para la realización de la siguiente fase se recomienda utilizar tanto el índice EVI como el SAVI, para detectar los puntos críticos del área. En el caso de nuestro trabajo, se seguirá explicando el resto del procedimiento utilizando la imagen EVI.

5.3 Tercera fase: Análisis de imágenes resultado

Después de haber obtenido las imágenes resultado con los índices de vegetación aplicados, la tercera y última fase se encarga de interpretar los datos recopilados para detectar las zonas donde el cultivo se encuentra en estado crítico, además de marcarlas en la imagen con sus coordenadas GPS. Lo primero será delimitar en la imagen diferentes zonas para poder realizar un diagnóstico de áreas más concretas. Para ello, generamos sobre la imagen una rejilla imaginaria que dividirá todo el terreno en pequeñas parcelas. Estas parcelas serán las áreas que el programa se encargará de analizar una a una, diagnosticando si requieren de intervención humana o no. El tamaño de las parcelas es decidido por el usuario previo a la ejecución del programa. En nuestro caso, al tratarse de imágenes de tamaño 840x480 hemos decidido utilizar un cuadrado de 80 píxeles de lado. En la Figura 5-5, podemos observar la rejilla dibujada sobre la imagen con el índice EVI aplicado.

Previamente a explicar el criterio de detección de puntos críticos, es necesario desglosar los posibles colores que aparecen en las imágenes con índice de vegetación aplicado para entender mejor el significado de cada uno de ellos. Los colores que generan los índices son:

- Planta saludable → Verde oscuro, verde y amarillo → (0.25 – 1].
- Planta estresada → Naranja y rojo → [-1 – 0.25]
- No existe vegetación → negro.



a) *b)*

Figura 5-5: Imágenes de ejemplo con índice de vegetación y rejilla imaginaria

Con los colores ya detallados, el criterio para decidir si un área del terreno se considera punto crítico sería: la parcela contiene más píxeles de planta estresada que de planta saludable y el número de píxeles de planta estresada es mayor que el número de píxeles sin vegetación. El criterio se ve resumido en la siguiente fórmula:

$$(p_{pe} > p_{ps}) \cap ((p_{pe} + p_{ps}) > p_n)$$

$p_{pe} = número\ de\ píxeles\ de\ planta\ estresada$
$p_{ps} = número\ de\ píxeles\ de\ planta\ saludable$
$p_n = número\ de\ píxeles\ \text{sin}\ vegetación\ (negro)$

La primera condición es obvia, ya que la forma de saber si una parcela requiere de intervención humana es debido a que existen más píxeles de mala salud que de buena. La segunda condición busca eliminar aquellas parcelas en las que la mayoría del terreno no se considera vegetación, ya que si no existirían casos de falsos positivos por toda la imagen.

Una vez detectadas las parcelas consideradas como puntos críticos, guardaremos las coordenadas del centro de la parcela dentro de la imagen: fila y columna del píxel que denota el centro del cuadrado. Con todos estos puntos críticos almacenados, ya sabemos dónde debería actuar el humano respecto de la imagen, pero, ¿cómo sabemos en qué lugar de la Tierra se encuentra el punto crítico? Como al realizar la misión se han guardado varios datos mientras se capturaban las imágenes, tenemos las coordenadas geográficas del centro de la imagen almacenadas. Previamente cabe destacar que, para todo este proceso, hay que tener en cuenta una suposición: el dron ha tomado las imágenes estando aproximadamente paralelo al suelo, es

decir, siendo $roll = 0, \quad pitch = 0$. Por tanto, lo primero que necesitamos es transformar el punto crítico 2D de la imagen al sistema de referencia 3D. La fórmula a utilizar se resume en:

$$(d_{x_{cam}}, d_{y_{cam}}, d_{z_{cam}}) = \left(\frac{(x - c_x) \cdot h_{cam}}{f_x}, \frac{(y - c_y) \cdot h_{cam}}{f_y}, h_{cam}\right)$$

x = coordenada X de un píxel en el sistema de coordenadas de la imagen.

c_x = coordenada X del píxel del centro de la imagen en el sistema de coordenadas de la imagen.

f_x = Distancia focal de la cámara en el eje X.

y = coordenada Y de un píxel en el sistema de coordenadas de la imagen.

c_y = coordenada Y del píxel del centro de la imagen en el sistema de coordenadas de la imagen.

f_y = Distancia focal de la cámara en el eje Y.

h_{cam} = Altura de la cámara respecto del suelo.

$d_{x_{cam}}$= Distancia al punto crítico en el eje X respecto del sistema de referencia de la cámara.

$d_{y_{cam}}$= Distancia al punto crítico en el eje Y respecto del sistema de referencia de la cámara.

$d_{z_{cam}}$= Distancia al punto crítico en el eje Z respecto del sistema de referencia de la cámara.

Al encontrarse la cámara a la altura aproximadamente del centro de masa, se cumple que:

$$h_{cam} = h_{odom}$$

Con esta fórmula, obtendremos la distancia al punto 3D respecto del centro de la imagen en el sistema de referencia de la cámara. Sin embargo, hay un detalle que no hemos tenido todavía en cuenta: las coordenadas GPS las recoge el módulo GPS en vez de la cámara. Estábamos suponiendo que la posición guardada en el fichero era exactamente la posición del centro de la imagen, pero existe un desplazamiento desde la cámara al GPS. Si nos fijamos en la Figura 3-5, el GPS se encuentra elevado respecto de la posición de la cámara además de estar posicionado detrás del vehículo. Por tanto, a los valores obtenidos con la fórmula anterior será necesario añadirles el desplazamiento existente entre la cámara y el GPS. Al no tener un modelo URDF del dron, estas medidas de las transformadas del GPS se han obtenido midiendo manualmente. Los desplazamientos en cada eje son:

$$GPS_{off_x} = 0\ cm, \qquad GPS_{off_y} = 20\ cm = 0.2\ m, \qquad GPS_{off_z} = 15\ cm = 0.15\ m$$

Finalmente, la fórmula de conversión de un punto de 2D a 3D se define como:

$$(d_{x_{GPS}}, d_{y_{GPS}}, d_{z_{GPS}}) = \left((d_{x\,cam}), \left(d_{y\,cam} - GPS_{off_y}\right), h_{odom} + GPS_{off_z} \right)$$

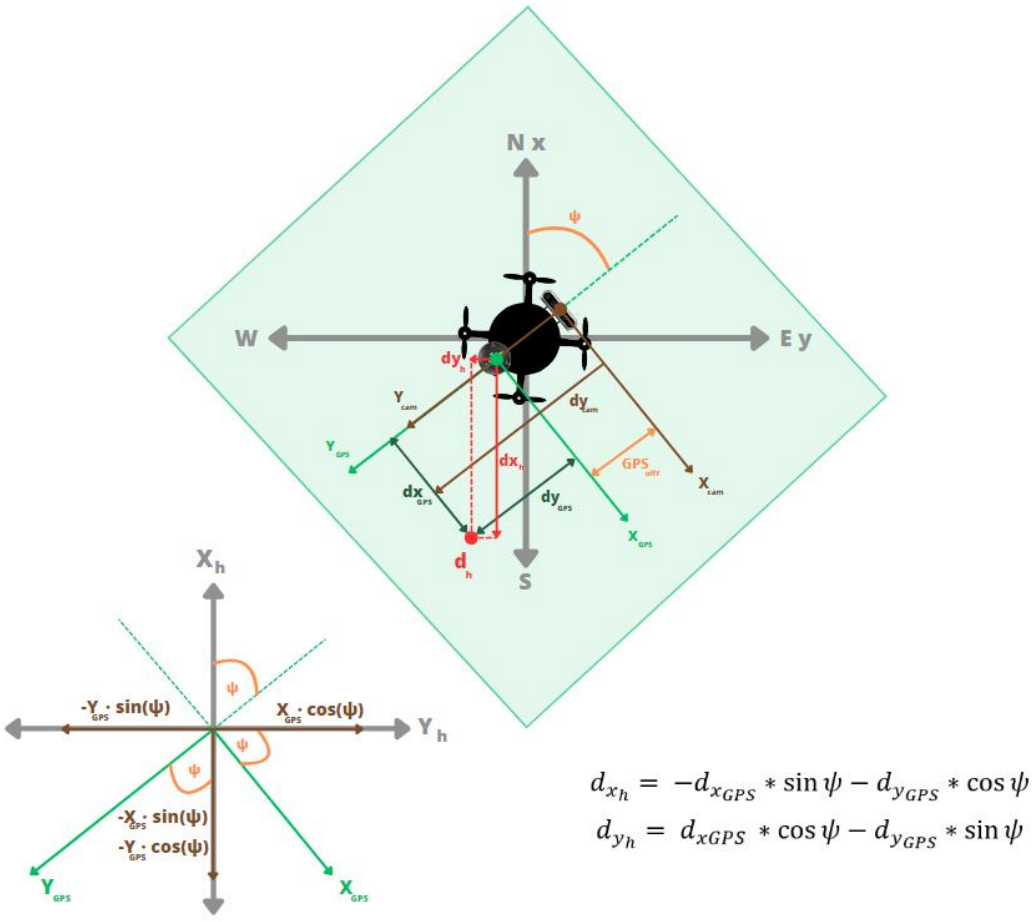
$d_{x_{GPS}}$= Distancia al punto crítico en el eje X respecto del sistema de referencia del GPS.

$d_{y_{GPS}}$= Distancia al punto crítico en el eje Y respecto del sistema de referencia del GPS.

$d_{z_{GPS}}$= Distancia al punto crítico en el eje Z respecto del sistema de referencia del GPS.

h_{odom} = Altura del centro de masa de la aeronave respecto al suelo.

Al conocer la posición del punto respecto al GPS del propio dispositivo, sólo queda calcular la posición del punto en el sistema de la Tierra. Para sacar dicha posición en sistema de coordenadas geográficas, es necesario proyectar la posición del sistema de referencia del GPS en los ejes del sistema de referencia NED. Para conseguir esta proyección, se requiere saber el yaw del dron en el momento en el que se capturó la imagen, el cual se encuentra guardado en el fichero de metadatos recopilado durante la misión. En la Figura 5-6, se aprecia un esquema que explica el procedimiento para llegar a la proyección deseada.



$$d_{x_h} = -d_{x_{GPS}} * \sin\psi - d_{y_{GPS}} * \cos\psi$$

$$d_{y_h} = d_{xGPS} * \cos\psi - d_{y_{GPS}} * \sin\psi$$

Figura 5-6: Representación del sistema de referencia de los dispositivos junto con el cálculo de la distancia en sistema de referencia del horizonte

Con este último cálculo, se han obtenido las distancias en cada uno de los ejes del sistema 3D en el sistema de referencia del horizonte en metros. Por tanto, una vez obtenidas las distancias alineadas con el sistema de referencia final, el último paso para obtener las coordenadas finales del punto crítico es sumar la distancia en metros a las coordenadas geográficas tomadas por el GPS. Lo primero de todo cabe destacar que, en el caso de este trabajo, no es necesario saber la altitud final del punto para localizarlo. Por tanto, se calcula la nueva latitud y longitud del punto crítico sobre el planeta Tierra, para así ser capaces de geolocalizar aquellas zonas que necesitan ser intervenidas. La fórmula que calcula la nueva latitud y longitud dado un desplazamiento en metros es:

$$Lat_{P_c} = \left(Lat_{GPS} + \left(\frac{d_{x_h}}{R_T} \right) \cdot \left(\frac{180}{\pi} \right) \right)$$

$$Lon_{P_c} = \left(Lon_{GPS} + \left(\frac{d_{y_h}}{R_T} \right) \cdot \frac{\left(\frac{180}{\pi} \right)}{\cos\left(Lat_{GPS} \cdot \frac{\pi}{180} \right)} \right)$$

Lat_{P_c} = Latitud del punto crítico.

Lat_{GPS} = Latitud tomada por el GPS.

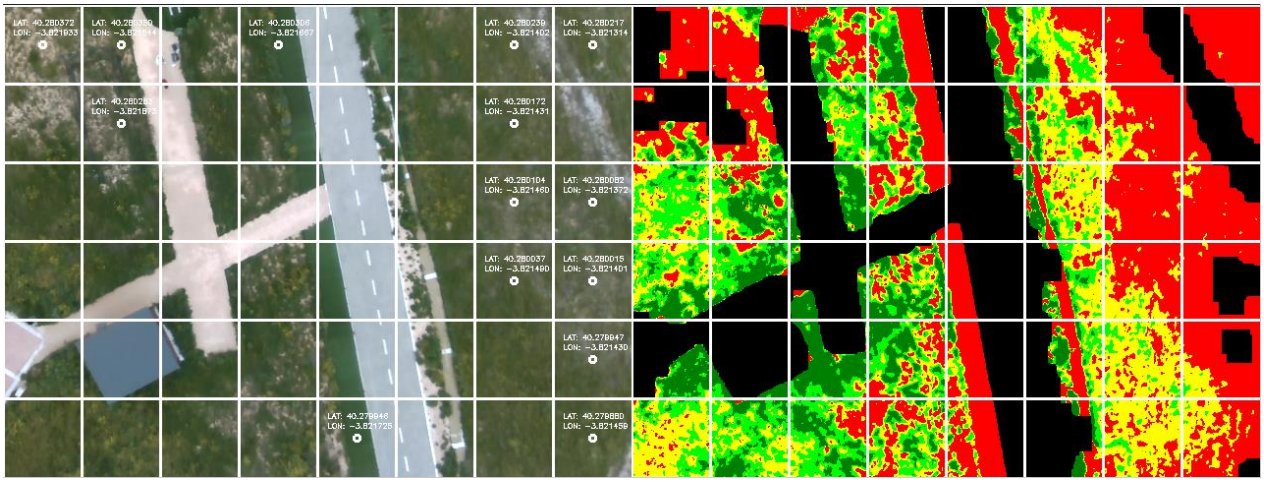
Lon_{P_c} = Longitud del punto crítico.

Lon_{GPS} = Longitud tomada por el GPS.

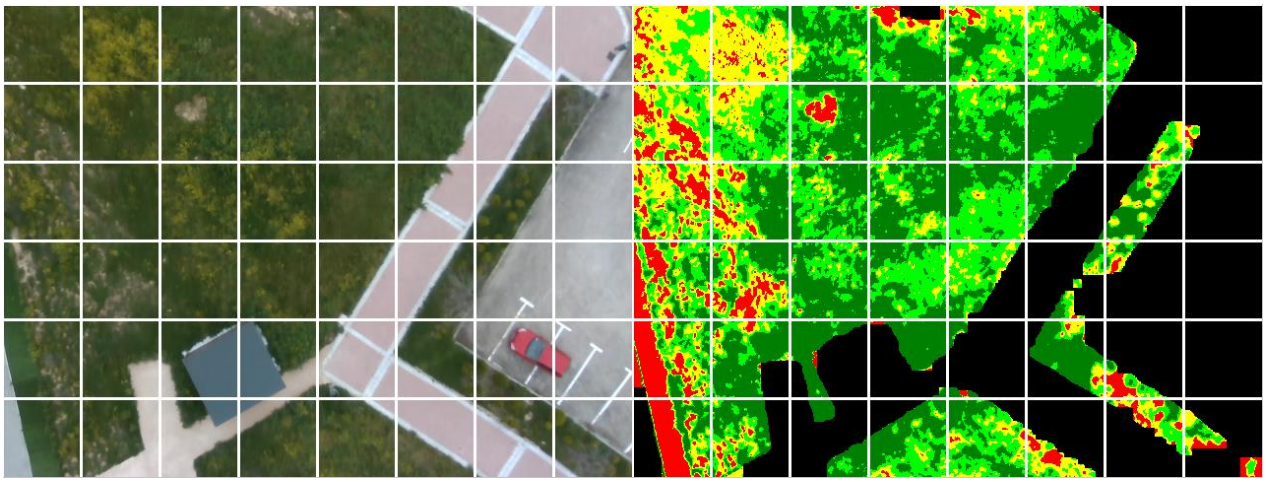
R_T = Radio de la Tierra = 6378 km aprox.

Después de obtener las coordenadas finales del punto crítico, sólo queda representar la información. Finalmente, se muestran en la imagen todos los puntos críticos con sus coordenadas ligadas a ellos. Para mayor legibilidad de los resultados obtenidos, los puntos críticos junto con sus coordenadas se pintarán en la imagen RGB en vez de en la imagen del índice. Además, se concatenará la imagen RGB con los puntos críticos junto con la imagen del índice de vegetación, ambas con la rejilla imaginaria superpuesta para encontrar mejor las parcelas a las que pertenecen los puntos.

En la Figura 5-7, se aprecian las imágenes resultado obtenidas utilizando las imágenes iniciales de la Figura 5-2. Analizando los diagnósticos obtenidos, la imagen a) no contiene ningún punto crítico en todo su terreno, mientras que la imagen b) tiene varias zonas críticas. Por ello, vamos a fijarnos principalmente en la imagen b) a la hora de comprobar los resultados de puntos críticos obtenidos. Si nos fijamos en esta imagen, la zona derecha principalmente contiene puntos críticos en todas sus áreas, algo que concuerda con la falta de vegetación y de verdor de toda esta zona. En la parte superior de la imagen se encuentra el norte, en la parte inferior el sur, a la izquierda el oeste y a la derecha el este.



a)



b)

Figura 5-7: Imágenes resultado del análisis

Por tanto, si un punto crítico se acerca a la parte superior de la imagen, la latitud aumentará dado que nos encontramos en el hemisferio norte. Siguiendo este sistema de referencia, cualquier punto crítico que se acerque al extremo derecho de la imagen, verá su longitud aumentada dado que se acerca al este. Finalmente, aquellas zonas que contienen menos píxeles

de vegetación que píxeles negros, no se considerarán parte del análisis. Un ejemplo de esto es el cuadrante que ocupa la mayor parte de la caseta en la imagen b), el cual no se ha considerado como punto crítico, aunque carezca de vegetación. En cuanto a la precisión de la geolocalización de los puntos, existe un error medio de 8 metros para la posición exacta de las áreas que requieren de intervención, siendo la medida con más error de 18 metros. En la Figura 5-8, se puede apreciar la comparación de la imagen b) junto con la representación de sus puntos críticos en Google Earth introducidos manualmente. En el apartado 6.3: Líneas de investigación futuras, explicaremos con más profundidad a que se debe en gran parte este error, además de aportar alguna solución para mejorar la precisión de los resultados.

Figura 5-8:Comparación de la posición de los puntos críticos respecto de su representación en Google Earth

6 Conclusiones

Una vez procesados los datos recopilados en las misiones planificadas del dron, hemos obtenido una serie de resultados respecto al análisis de la salud del terreno de cultivo. Sin embargo, todavía no hemos realizado una comprobación de si hemos obtenido los objetivos iniciales del trabajo, ni de las capacidades aprendidas durante el transcurso de este. En este capítulo, vamos a sacar conclusiones de todo el proceso explicado previamente en este documento, es decir, vamos a indagar en si la aplicación de nuestro trabajo se ha desviado en algunas partes de la implementación de nuestra idea inicial, aparte de verificar qué otras se han cumplido según lo esperado.

En cuanto a las desviaciones que hayan existido en nuestro trabajo, desglosaremos ciertas mejoras que se pueden implementar en futuras investigaciones relacionadas con este trabajo, para poder acercarnos más a una implementación más precisa. Todas estas mejoras irán acompañadas de una explicación en la que se detecta el problema existente en el proyecto actual, junto con la solución que se aporta.

6.1 Evaluación de objetivos

Una vez desarrollado el proyecto completamente, es hora de volver a repasar los objetivos iniciales de nuestro trabajo:

a) Captura de imágenes y datos de importancia: se ha desarrollado una arquitectura, tanto hardware como software, que ha permitido la cohesión del vehículo aéreo no tripulado con los diferentes dispositivos externos:

a. Se ha añadido una cámara al vehículo con ayuda de una pieza 3D personalizada, para poder tomar imágenes con una posición fija durante el vuelo.

b. Se ha añadido una Raspberry Pi a la aeronave para poder unificar la información de la cámara y el controlador del dron en un solo dispositivo, además de almacenar todos estos datos para su posterior análisis. Ha sido necesario incorporar una batería portátil que soporte alimentación a esta unidad de procesamiento.

c. Se han instalado diferentes librerías y herramientas software con drivers incluidos, que permiten la comunicación entre dispositivos.

b) Estudio y aplicación de técnicas de detección de vegetación: se han expuesto diferentes técnicas de detección de vegetación: desde índices que requieren sólo de imagen RGB como el índice EXG, hasta índices que requieren de espectro de luz visible e infrarrojo como NDVI, EVI o SAVI. Se han mostrado los resultados con cada uno de los índices explicados en el documento.

c) Geolocalización de los puntos críticos del cultivo: durante el análisis de los datos obtenidos, se han procesado las imágenes y obtenido los puntos críticos del área analizada. Se han utilizado un algoritmo para convertir esas distancias 2D de la imagen en distancias 3D, se han corregido desplazamientos entre mediciones y se han calculado las nuevas coordenadas GPS del punto crítico en la Tierra. Se ha detectado que existe un error en las coordenadas representadas en la imagen respecto del lugar en el mundo real. La causa de este error se explicará detalladamente en el apartado 6.3: Líneas de investigación futuras.

d) Análisis y validación de los resultados: Se han procesado los datos obtenidos en las misiones planificadas y se han procesado para generar un análisis del terreno sobrevolado. Se ha generado un formato de representación de los resultados que sea legible para el usuario, además de representar la información de mayor importancia.

6.2 Capacidades adquiridas

Después de haber evaluado la consecución de los objetivos presentados inicialmente en este documento, el proceso de materializar nuestro planteamiento del proyecto en el resultado buscado ha conseguido que desarrollemos una serie de capacidades gracias al diseño realizado.

Para poder implementar el objetivo a), se ha desarrollado una arquitectura tanto hardware como software, que permite conectar a la aeronave con los dispositivos externos añadidos con el objetivo de capturar los datos necesarios durante la misión. Al buscar una solución para acoplar la cámara al dron en una posición fija, se han adquirido conocimientos en modelado e impresión 3D para crear la pieza que se encarga de unir ambos dispositivos. Además, se ha obtenido valiosa información de los diferentes materiales posibles para utilizar durante la impresión del modelo. En cuanto a la conexión física entre la Raspberry Pi y el controlador PX4, se han adquirido conocimientos tanto de electrónica como de soldadura, al conseguir soldar varios cables Dupont hembra a cada uno de los cables que van conectados al puerto de telemetría del controlador. Dentro de este campo, se han obtenido conocimientos de los diferentes pines utilizados en electrónica, además de protocolos seriales como UART o USB. Respecto del software utilizado, se han obtenido conocimientos sobre ROS 2 a la hora de tanto publicar como suscribirse a topics para obtener información. También se han desarrollado habilidades tanto para la compilación de librerías, como de su instalación en diferentes dispositivos. Finalmente, se han implementado puentes de comunicación entre diferentes softwares que utilizan el mismo protocolo de comunicación, como en el caso de ROS 2-PX4.

Respecto del objetivo b), se han obtenido conocimientos sobre diferentes técnicas de vegetación: desde técnicas que sólo utilizan el espectro de luz visible como el índice ExG, hasta índices que utilizan 2 espectros diferentes como los índices NDVI, EVI y SAVI. Además, se han obtenido capacidades en tratamiento de imagen con la librería OpenCV a la hora de aplicar la fórmula de los índices previamente mencionados y obtener un diagnóstico del terreno.

Finalmente, para la consecución de los objetivos c) y d) se han obtenido capacidades de conversión de coordenadas del plano 2D al 3D. Además, se han aplicado conversiones de unidades respecto de diferentes sistemas de coordenadas, como por ejemplo de coordenadas cartesianas a coordenadas geográficas a la hora de obtener las coordenadas GPS de un punto crítico. Para el análisis del cultivo, se han utilizado diferentes funciones de representación de

información en imágenes con la librería OpenCV, mostrando los datos finales del diagnóstico de forma accesible al usuario, facilitando la lectura de los resultados obtenidos.

6.3 Líneas de investigación futuras

Al principio de este documento, expusimos los objetivos que se buscaban cumplir con el desarrollo de este proyecto. Durante el apartado 6.1: Evaluación de objetivos, hemos comprobado cuáles de estos objetivos se cumplen y de qué forma se han implementado en nuestro proyecto. Sin embargo, existen otras implementaciones similares a la nuestra que pueden dar un punto de vista diferente del obtenido con nuestro resultado, además de ciertas mejoras que pueden aportar una solución más pulida. Algunas de ellas son:

a) Mejora de la precisión de la geolocalización de los puntos críticos: existe un error entre la posición calculada por el algoritmo y la posición del punto crítico en la Tierra. Esto se debe a 2 problemas que existen actualmente: la posición medida por el GPS tiene un error de metros y el dron no toma las imágenes en posición horizontal perfecta. Con la siguiente mejora que se expone, se conseguirá mejorar en este ámbito.

b) Incorporación del gimbal acoplado a la cámara: un gimbal es un dispositivo capaz de estabilizar la cámara, contrarrestando los movimientos del dron que afectan a la orientación de esta. Esta mejora, permite aumentar la precisión de la geolocalización de los puntos críticos al permitir tomar imágenes del terreno con la cámara perfectamente estabilizada horizontalmente.

c) Análisis de cultivo fusionado: durante el comienzo del desarrollo de este proyecto, se planteó la idea de fusionar todas las imágenes capturadas del terreno, para realizar un análisis unificado de todo el terreno de cultivo. Sin embargo, este proceso se trata de un

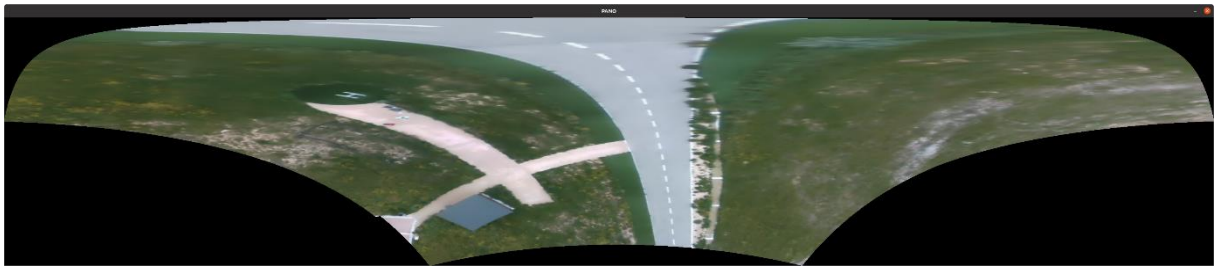


Figura 6-1: Imagen panorámica del cultivo mal fusionada

sistema complejo que requiere de una buena metodología para unificar todos los fragmentos. En la Figura 6-1, se puede observar un ejemplo de mala elección de metodología de fusión de imágenes aplicada a las fotos tomadas durante una de las misiones.

d) Fichero con los puntos críticos almacenados: Al guardar todos los puntos críticos en un fichero, unificamos todas las áreas de intervención recopiladas de todas las imágenes en un mismo sitio. Este fichero se puede utilizar, tanto para ir con ayuda de un dispositivo GPS a los lugares que requieren de intervención, como para representarlos en alguna aplicación como Google Maps o Google Earth.

REFERENCIAS

AliExpress. (23 de Mayo de 2024). *AliExpress*. Obtenido de AliExpress: https://es.aliexpress.com/i/32898066295.html

AliExpress. (4 de junio de 2024). *AliExpress*. Obtenido de AliExpress: https://es.aliexpress.com/item/1005006756353921.html?src=google&src=google&albch=shopping&acnt=439-079-4345&slnk=&plac=&mtctp=&albbt=Google_7_shopping&gclsrc=aw.ds&albagn=888888&isSmbAutoCall=false&needSmbHouyi=false&src=google&albch=shopping&acnt=439-079

Amazon. (25 de Enero de 2024). *Amazon*. Obtenido de Amazon: https://www.amazon.es/gp/product/B07PNL5STG/ref=ppx_yo_dt_b_asin_title_o00_s00?ie=UTF8&th=1

BBVA. (24 de Agosto de 2023). *¿Qué es la agricultura de precisión? La gestión digital del campo*. Obtenido de https://www.bbva.com/es/sostenibilidad/que-es-la-agricultura-de-precision-la-gestion-digital-del-campo/

BioInteractive. (11 de Febrero de 2020). *BioInteractive*. Obtenido de BioInteractive: https://www.biointeractive.org/es/classroom-resources/fotosintesis#:~:text=La%20fotos%C3%ADntesis%20convierte%20la%20energ%C3%ADa,productores%20y%20nutrir%20los%20ecosistemas.

BioMed Central. (22 de Mayo de 2018). *BioMed Central*. Obtenido de BioMed Central: https://plantmethods.biomedcentral.com/articles/10.1186/s13007-019-0402-3/tables/6

DroneCode. (10 de Octubre de 2023). *DroneCode*. Obtenido de DroneCode: https://dronecode.org/

DST. (28 de Septiembre de 2023). *dstfotografia.com*. Obtenido de dstfotografia.com: https://dstfotografia.com/ndvi/

Editors, H. (12 de Enero de 2018). *History.com*. (A&E Television Networks) Recuperado el 2 de Julio de 2023, de https://www.history.com/topics/pre-history/neolithic-revolution

Escobar, R. (22 de Septiembre de 2019). *Dunas de Cydonia*. Obtenido de Blogspot: https://stg-pepper.blogspot.com/2019/09/registro-de-imagenes-con-sift-en-opencv.html

FastGrowingTrees. (23 de septiembre de 2023). *fastgrowingtrees.com*. Obtenido de fastgrowingtrees.com: https://www.fast-growing-trees.com/pages/soil-ph

FIWARE Zone. (2 de Noviembre de 2023). *FIWARE Zone Docs*. Obtenido de FIWARE Zone Docs: https://fiware-zone.readthedocs.io/es/latest/fast-rtps-micro-rtps.html

FlyingTech. (23 de Mayo de 2024). *flyingtech.co.uk*. Obtenido de flyingtech.co.uk: https://www.flyingtech.co.uk/product/flysky-fs-i6x-10ch-transmitter-fs-x6b-2-4ghz-ibus-receiver/

GCFGlobal. (19 de Octubre de 2023). *GCFGlobal*. Obtenido de GCFGlobal: https://edu.gcfglobal.org/es/fisica/tercera-ley-de-newton-accion-y-reaccion/1/

GitHub. (10 de Junio de 2023). *GitHub*. Obtenido de GitHub: https://github.com/IntelRealSense/realsense-ros

GitHub. (5 de junio de 2024). *GitHub*. Obtenido de Github: https://github.com/DiegoURJC/post_mission_processing

GitHub. (6 de Junio de 2024). *GitHub*. Obtenido de GitHub: https://github.com/IntelRealSense/librealsense

GitHub. (7 de junio de 2024). *GitHub*. Obtenido de GitHub: https://github.com/DiegoURJC/camera_trigger_qt

Gladys Toribio. (1 de mayo de 2020). *cursosteledeteccion.com*. Obtenido de cursosteledeteccion.com: https://www.cursosteledeteccion.com/ndvi-que-es-y-para-que-sirve/

Holybro. (24 de Septiembre de 2023). *Holybro*. Obtenido de Holybro: https://holybro.com/products/px4-development-kit-x500-v2?variant=43018371530941

idorobotics. (24 de marzo de 2018). *idorobotics.com*. Obtenido de idorobotics.com: https://idorobotics.com/2018/03/24/how-the-intel-realsense-r200-camera-works/

Intel RealSense. (27 de Septiembre de 2023). *Intel RealSense*. Obtenido de Intel RealSense: https://store.intelrealsense.com/buy-intel-realsense-depth-camera-d435.html

Intel RealSense. (27 de Septiembre de 2023). *Intel RealSense*. Obtenido de Intel RealSense: https://www.intel.la/content/www/xl/es/architecture-and-technology/realsense-overview.html

Intel RealSense. (23 de Mayo de 2024). *Intel RealSense*. Obtenido de Intel RealSense: https://www.intelrealsense.com/depth-camera-d435/

Intel RealSense Support. (7 de junio de 2020). *Intel RealSense Support*. Obtenido de Intel RealSense Support: https://support.intelrealsense.com/hc/en-us/community/posts/360051296993-Ubuntu-Compatibility

Knott, R. (10 de Agosto de 2021). *Techsmith*. Obtenido de Techsmith: https://www.techsmith.com/blog/image-cropping-101-basics/#:~:text=To%20%E2%80%9Ccrop%E2%80%9D%20an%20image%20is,image%20by%20removing%20unnecessary%20parts.

Linazasoro, I. (9 de Mayo de 2023). *Linazasoro optika*. Obtenido de Linazasoro optika: https://linazasoro-optika.eus/la-luz-azul-deberia-importarte/

Linazasoro, I. (9 de Mayo de 2023). *Linazasoro Optika*. Obtenido de linazasoro Optika: https://linazasoro-optika.eus/la-luz-azul-deberia-importarte/

Lozano, R. (2001). *Unmanned Aerial Vehicles Embedded Control*. Krieger Publishing Company.

Mallick, S. (11 de Marzo de 2018). *Learn OpenCV*. Obtenido de Learn OpenCV: https://learnopencv.com/image-alignment-feature-based-using-opencv-c-python/

Martínez, J. (15 de Diciembre de 2021). *DataSmarts*. Obtenido de DataSmarts: https://www.datasmarts.net/como-rotar-imagenes-en-opencv/

Martínez, J. (17 de Enero de 2022). *DataSmarts*. Obtenido de DataSmarts: https://www.datasmarts.net/como-usar-mascaras-en-opencv/

Mástoner. (1 de Noviembre de 2021). *Mástoner*. Obtenido de Mástoner: https://mastoner.com/blog/diferencia-entre-filamento-pla-y-abs/#:~:text=Se%20denomina%20filamento%20PLA%20al,del%20tipo%20FDM%20del%20mercado.

McCormick. (24 de Noviembre de 2021). *McCormick*. Obtenido de McCormick: https://www.mccormick.it/es/agricultura-4-0-que-es-y-cuales-son-sus-herramientas-y-beneficios/#Tecnolog%C3%ADas

OpenCV Docs. (28 de Junio de 2023). *OpenCV Docs*. Obtenido de OpenCV Docs: https://docs.opencv.org/4.8.0/d1/dfb/intro.html

OpenCV Docs. (9 de Mayo de 2024). *OpenCV Docs*. Obtenido de OpenCV Docs: https://docs.opencv.org/3.4/db/df6/tutorial_erosion_dilatation.html

OpenCV Docs. (17 de Mayo de 2024). *OpenCV Docs*. Obtenido de OpenCV Docs: https://docs.opencv.org/3.4/db/df6/tutorial_erosion_dilatation.html

Pedro Castillo García, L. E. (2016). *Indoor Navigation Strategies for Aerial Autonomous Systems*. Butterworth-Heinemann.

PRUSA. (26 de Octubre de 2023). *PRUSA Research*. Obtenido de PRUSA Research: https://www.prusa3d.com/es/producto/original-prusa-mini-semi-montada-4/?country=ES&currency=eur&gad_source=1&gclid=CjwKCAjwnOipBhBQEiwACyGLusq5XqL0d0eqeUYhedwWRIqn6bsGS-OcmHLZxu8rFSqs8XAPU3fRgBoC33wQAvD_BwE

PX4. (3 de Junio de 2021). *PX4 Docs*. Obtenido de PX4 Docs: https://docs.px4.io/v1.12/en/middleware/micrortps.html

PX4 AutoPilot. (22 de Junio de 2021). *PX4 AutoPilot Doc*. Obtenido de PX4 AutoPilot Doc: https://docs.px4.io/v1.12/en/ros/ros2_comm.html

PX4 Autopilot. (13 de Abril de 2023). *PX4 Autopilot*. Obtenido de PX4 Autopilot: https://docs.px4.io/main/en/frames_multicopter/holybro_x500_pixhawk4.html

PX4 Autopilot. (28 de septiembre de 2023). *PX4 Autopilot*. Obtenido de PX4 Autopilot: https://docs.px4.io/main/en/getting_started/px4_basic_concepts.html

Rivas, A. (20 de Marzo de 2023). *Normas APA*. Obtenido de Normas APA: https://normasapa.in/estado-del-arte/#%C2%BFQue_es_el_estado_del_artehttps://normasapa.in/estado-del-arte/#%C2%BFQue_es_el_estado_del_arte

ROS 2. (14 de Ocutbre de 2023). *ROS 2 Documentation*. Obtenido de ROS 2 Documentation: https://docs.ros.org/en/foxy/Tutorials/Beginner-Client-Libraries/Writing-A-Simple-Cpp-Publisher-And-Subscriber.html#

ROS 2. (7 de Enero de 2024). *ROS 2 Docs*. Obtenido de ROS 2 Docs: https://docs.ros.org/en/rolling/Concepts/Intermediate/About-Quality-of-Service-Settings.html

ROS Answers. (15 de Enero de 2020). *ROS Answers*. Obtenido de ROS Answers: https://answers.ros.org/question/341629/ubuntu-1604-or-raspbian-jesse-for-raspberry-pi-4/

Rosebrock, A. (19 de Enero de 2021). *pyimagesearch*. Obtenido de pyimagesearch: https://pyimagesearch.com/2021/01/19/opencv-bitwise-and-or-xor-and-not/

Ruiz, P. R. (16 de Octubre de 2019). *Osbodigital: Portal de información y análisis de los incendios forestales*. Recuperado el 15 de Agosto de 2023, de https://osbodigital.es/2019/10/16/los-drones-una-herramienta-con-muchas-aplicaciones/

Sembralia. (14 de Octubre de 2020). *Suelos Alcalinos, te mostramos cómo bajar el pH*. Obtenido de https://sembralia.com/blogs/blog/suelo-alcalino-ph

Toribio, G. (25 de Octubre de 2019). *CursosTeledetección.com*. Obtenido de CursosTeledetección.com: https://www.cursosteledeteccion.com/ndvi-que-es-y-para-que-sirve/

USGS. (5 de junio de 2024). *USGS*. Obtenido de USGS: https://www.usgs.gov/landsat-missions/landsat-enhanced-vegetation-index

Wikipedia. (7 de septiembre de 2023). *Wikipedia*. Obtenido de Wikipedia: https://es.wikipedia.org/wiki/Estaci%C3%B3n_de_Control_Terrestre_de_UAV

Wikipedia. (9 de Agosto de 2023). *Wikipedia*. Obtenido de Wikipedia: https://es.wikipedia.org/wiki/OpenCV

Wikipedia. (17 de Abril de 2024). *Wikipedia*. Obtenido de Wikipedia: https://en.wikipedia.org/wiki/Otsu%27s_method