Facultad de
INGENIERÍA
Universidad Nacional de Lomas de Zamora

PROYECTO EN INGENIERÍA MECATRÓNICA

Dron autopilotado con detección de fuego y personas mediante procesamiento IA de cámaras

Integrantes:

- FERNANDEZ MALENOTTI, Ignacio DNI: 42.043.463
- MOSCARIELLO, Christian DNI: 37.053.307
- YACONO, Emiliano DNI: 34.485.253

Docentes:

- Ing. Blanca, Ezequiel
- Ing. Lukaszewickz, Cristian
- Ing. Szombach, Juan

Contenido

1. Título del Proyecto

Dron Autónomo para Detección de Incendios y Rescate de Personas

2. Introducción

Durante los meses de verano, diferentes áreas naturales —como parques nacionales, reservas ecológicas y zonas de montaña— se ven afectadas por dos problemáticas críticas y de creciente relevancia: la aparición de focos de incendio y el extravío de personas en entornos de difícil acceso. La vegetación seca y las altas temperaturas favorecen la rápida propagación del fuego, mientras que la amplitud del terreno y la escasez de personal dificultan las tareas de búsqueda, control y patrullaje terrestre.

La detección tardía de un foco ígneo o la demora en localizar a una persona extraviada incrementan significativamente el riesgo para la vida humana y el impacto ambiental. Si bien los helicópteros representan una herramienta efectiva para tareas de vigilancia aérea, su uso implica costos operativos elevados, limitaciones en disponibilidad y una cobertura espacial acotada, lo que restringe su empleo de manera continua o preventiva.

Ante este escenario, surge la necesidad de explorar alternativas tecnológicas más accesibles, flexibles y autónomas, capaces de brindar una respuesta temprana y asistir a equipos de emergencia en situaciones donde cada minuto resulta determinante.

3. Objetivo y Propuesta del Proyecto

El presente proyecto busca desarrollar un prototipo funcional de dron autónomo a escala, orientado a operaciones de monitoreo, detección temprana y apoyo logístico en contextos de incendio o búsqueda y rescate. La propuesta abarca las siguientes capacidades principales:

1. Detección visual y térmica de focos de incendio en etapas iniciales.
2. Localización de personas extraviadas mediante análisis de imágenes RGB e infrarrojas.
3. Transmisión de coordenadas geográficas precisas hacia una estación base para facilitar la intervención.
4. Transporte de pequeñas cargas, tales como elementos de supervivencia, primeros auxilios o materiales para iniciar la contención temprana de un incendio.

El alcance del proyecto no busca reemplazar a sistemas profesionales de respuesta aérea, sino demostrar la viabilidad técnica de una plataforma autónoma liviana, versátil y de bajo costo, capaz de inspeccionar grandes superficies y asistir en tareas que actualmente requieren importantes recursos humanos y económicos.

4. Enfoque del Desarrollo

El desarrollo del proyecto se abordó desde una perspectiva integral de la ingeniería mecatrónica, combinando conocimientos y técnicas de diversas áreas para construir un prototipo funcional capaz de ejecutar tareas complejas de manera autónoma. El enfoque adoptado prioriza la modularidad, permitiendo diseñar, probar e integrar cada subsistema de forma independiente antes de su ensamblaje final.

En materia de percepción del entorno, se implementó un esquema de detección multimodal basado en cámaras RGB y sensores térmicos infrarrojos. Este sistema permite identificar focos de incendio y localizar personas tanto mediante análisis visual tradicional como a través de diferencias de temperatura en la escena. Para potenciar estas capacidades, se incorporan algoritmos de inteligencia artificial entrenados específicamente para las clases relevantes del proyecto (fuego y personas).

El procesamiento de datos se realizó mediante sistemas embebidos, combinando una Raspberry Pi 4 para tareas de visión por computadora y analítica, y una ESP32 encargada de comunicaciones y control de misión. La arquitectura se complementa con la controladora Pixhawk, responsable de la estabilización del dron y la ejecución del vuelo autónomo mediante GPS y el protocolo MAVLink.

En el ámbito de comunicaciones inalámbricas, se adoptó un módulo LoRa de largo alcance para transmitir eventos detectados, garantizando operación en entornos rurales o de baja infraestructura. Esta elección permitió validar la viabilidad de enviar información crítica aún bajo restricciones severas de ancho de banda.

Finalmente, la etapa de integración combinó navegación autónoma, percepción multimodal, procesamiento inteligente y transmisión de alertas en un único flujo de misión. El resultado es un sistema capaz de despegar, desplazarse entre waypoints, detenerse para analizar su entorno, detectar situaciones relevantes, reportarlas a una estación base y continuar su ruta de manera autónoma.

Este enfoque refleja la convergencia de mecánica, electrónica, software, telecomunicaciones e inteligencia artificial, consolidando un prototipo alineado con las tendencias actuales de automatización y apoyo tecnológico en operaciones de emergencia.

5. Análisis del Mercado y Tecnologías Disponibles

En el mercado actual existe una oferta consolidada de vehículos aéreos no tripulados (UAVs) diseñados para tareas de vigilancia ambiental, detección temprana de incendios y operaciones de búsqueda y rescate. Fabricantes como DJI y Autel Robotics han desarrollado plataformas avanzadas —entre ellas el DJI Matrice 30T y el Autel EVO II Dual— equipadas con cámaras térmicas de alta resolución, sensores ópticos de largo alcance, inteligencia artificial embarcada y transmisión de video en tiempo real. Estas aeronaves se utilizan ampliamente en brigadas forestales, bomberos y equipos de emergencia debido a su capacidad para inspeccionar grandes superficies, operar en condiciones climáticas adversas y enviar información crítica a centros de control en cuestión de segundos.

A pesar de la diferencia en escala, prestaciones y costos, estos drones comparten varias funciones esenciales con el propósito central de este proyecto, entre ellas:

- Detección térmica de focos de calor, permitiendo identificar incendios en etapas tempranas.
- Registro visual de alta precisión, útil para tareas de reconocimiento del entorno y búsqueda de personas.
- Transmisión remota de información georreferenciada a una estación base operativa.
- Navegación autónoma o semiautónoma, asistida por GPS y sistemas de piloto automático.

Estas capacidades representan el estándar tecnológico del sector profesional de vigilancia aérea y emergencias.

El prototipo desarrollado en este proyecto réplica y adapta dichas funciones en una plataforma experimental de arquitectura abierta, integrando una cámara térmica, un sistema RGB, algoritmos de inteligencia artificial para detección de fuego y personas, navegación autónoma mediante Pixhawk y un sistema de comunicaciones de largo alcance basado en LoRa. Aunque no compite directamente con drones comerciales de alta gama, el prototipo busca demostrar la viabilidad técnica de un UAV autónomo capaz de cumplir misiones equivalentes en un contexto de bajo costo y orientado a validación académica.

5.1 Comparación con drones comerciales relevantes

A continuación, se presenta una tabla comparativa que resume las características principales de los drones comerciales más utilizados en tareas similares y su relación con el prototipo desarrollado:

Característica	Dron del Proyecto	DJI Matrice 30T	Autel EVO II Dual
Cámara térmica	MLX90640 (32×24)	Radiométrica profesional	Radiométrica profesional
Cámara RGB	PiCam	Zoom óptico + gran angular	Cámara 8K
IA embarcada	Sí (YOLOv8 entrenado para incendios/personas)	Sí (IA propietaria)	Sí (IA propietaria)
Video en vivo	No (limitación técnica de transmisión)	Sí (RGB + térmico)	Sí (RGB + térmico)
Autonomía	4–6 min	40–50 min	~40 min
Navegación autónoma	Pixhawk + GPS	Multisensor + RTK	GPS + sensores avanzados
Comunicaciones	LoRa + telemetría	Enlace digital HD propietario	Enlace digital HD propietario
Costo aproximado	Muy bajo	Muy alto (USD 12.000–30.000)	Alto (USD 4.000–9.000)
Modularidad	Total	Baja	Media

5.2 Diferencias funcionales relevantes

5.2.1 Ventaja del prototipo: IA entrenada específicamente para incendios y personas

A diferencia de los drones comerciales que incorporan modelos de IA genéricos y optimizados para una amplia variedad de escenarios, el prototipo desarrollado en este proyecto utiliza un modelo YOLOv8 entrenado específicamente para:

- Detección de focos de incendio
- Personas en entornos naturales

Este enfoque especializado permite adaptar el rendimiento del sistema a las necesidades concretas de la misión, aprovechando un pipeline completamente local (on-board processing) que no depende de servidores externos ni de enlaces de comunicación de alta velocidad.

5.2.2 Diferencia fundamental: falta de transmisión de video en vivo

Los drones comerciales de emergencia utilizan enlaces digitales de gran ancho de banda (como OcuSync o SkyLink) que permiten enviar video RGB y térmico en tiempo real hacia

un operador remoto.
En contraste, el presente prototipo no transmite video en vivo, debido a:

- la baja capacidad de transmisión del módulo LoRa,
- las limitaciones energéticas del hardware embarcado,
- y la prioridad de diseño puesta en procesar imágenes a bordo y enviar solo alertas relevantes.

Sin embargo, esta restricción no afecta la misión propuesta, dado que el sistema está orientado a:

- capturas puntuales,
- procesamiento local mediante IA,
- y transmisión de eventos georreferenciados.

Este enfoque reduce consumo energético, complejidad del sistema y dependencia de infraestructura de comunicaciones externas.

6. Presupuesto y Costos del Sistema Desarrollado

6.1 Componentes utilizados

El prototipo desarrollado integra una arquitectura abierta basada en sistemas embebidos, sensores térmicos y ópticos, actuadores brushless y una controladora de vuelo de grado hobby.
La siguiente tabla resume los componentes principales utilizados en la construcción del dron, incluyendo módulos electrónicos, sensores, actuadores y estructura portante.

Lista de componentes del sistema

Ítem	Componente	Cant.	Precio unitario (USD)	Subtotal (USD)
1	Módulo ESP32	1	9.55	9.55
2	Raspberry Pi 4	1	90.45	90.45
3	Módulo LoRa SX1278	2	19.50	38.99
4	Cámara térmica M5Stick T-Lite (MLX90640)	1	123.29	123.29
5	Cámara RGB Raspberry	1	11.85	11.85
6	Motores brushless	6	30.13	180.78
7	ESC para motores brushless	6	16.44	98.64
8	Batería Li-Po	1	211.87	211.87
9	Controladora de vuelo Pixhawk 2.4.8	1	159.99	159.99
10	Frame hexacóptero	1	59.99	59.99
11	Servo SG90	1	2.96	2.96
	Total			988.36

Este valor corresponde al costo directo del hardware requerido para el prototipo, excluyendo impresión 3D, consumibles, repuestos, herramientas y horas de trabajo técnico.

6.2 Costos unitarios y costo total del prototipo

El costo total del sistema es de 988.86 USD, lo que lo posiciona como una plataforma de muy bajo costo comparada con los UAV comerciales destinados a tareas de emergencia y vigilancia ambiental.

Algunos puntos clave:

- La Raspberry Pi 4 representa una parte importante del costo, pero permite ejecutar IA embarcada sin necesidad de hardware propietario.
- La Pixhawk provee autonomía en vuelo similar a controladoras profesionales, manteniendo un costo accesible.
- La cámara térmica MLX90640, aunque de baja resolución, es significativamente más económica que las unidades radiométricas integradas en drones comerciales.
- El uso de LoRa reduce drásticamente costos de transmisión a cambio de sacrificar video en vivo.

El resultado es un sistema económicamente viable y orientado al prototipado académico, que integra tecnologías avanzadas a una fracción del costo de las soluciones comerciales.

6.3 Comparación económica con drones comerciales

Para contextualizar el costo del prototipo, se comparó el costo total del dron desarrollado con dos UAV comerciales ampliamente utilizados en operaciones de emergencia:

Modelo	Precio estimado (USD)	Diferencia contra el dron del proyecto
Dron del proyecto	988 USD	—
DJI Matrice 30T	12,000–30,000 USD	12× a 30× más costoso
Autel EVO II Dual	4,000–9,000 USD	4× a 9× más costoso

6.4 Conclusión comparativa y justificación económica del diseño

El análisis económico demuestra que el dron del proyecto representa una alternativa extremadamente accesible frente a los UAV comerciales utilizados en tareas de emergencia. Con un costo total aproximado de 990 USD, el prototipo equivale a:

- entre el 4% y el 10% del valor de un Autel EVO II Dual,

- entre el 3% y el 9% del valor de un DJI Matrice 30T.

A pesar de esta diferencia económica significativa, el sistema desarrollado logra validar funciones esenciales presentes en estos equipos de alta gama, tales como:

- detección térmica,
- inteligencia artificial embarcada,
- navegación autónoma mediante GPS,
- transmisión remota de eventos y coordenadas georreferenciadas.

La viabilidad económica del proyecto se sustenta en decisiones de diseño orientadas a maximizar prestaciones con un presupuesto reducido. La selección de componentes accesibles —como ESP32, Raspberry Pi, Pixhawk y el sensor térmico MLX90640— permite construir un sistema completo sin depender de hardware propietario ni ecosistemas cerrados.

Asimismo, la modularidad de la arquitectura facilita el reemplazo, mejora y ampliación de sensores o procesadores, favoreciendo la experimentación y las iteraciones rápidas propias de entornos académicos. Este enfoque resulta especialmente adecuado para proyectos que requieren:

- múltiples ciclos de prueba y error,
- diagnóstico y modificación de subsistemas,
- integración incremental de nuevas capacidades.

En síntesis, aunque no pretende competir con las plataformas profesionales en términos de robustez o autonomía, el dron del proyecto ofrece una relación costo–prestación altamente favorable, demostrando que es posible integrar tecnologías avanzadas como visión térmica, IA embarcada y vuelo autónomo por una fracción del costo de los sistemas comerciales.

7. Arquitectura General del Sistema

La arquitectura del sistema desarrollado se basa en una estructura modular que integra sensores ópticos y térmicos, procesamiento inteligente mediante inteligencia artificial, comunicaciones de largo alcance y control de vuelo autónomo. Este enfoque permite distribuir las funciones entre distintos módulos especializados, facilitando el diseño, la depuración, la integración y la escalabilidad del prototipo.

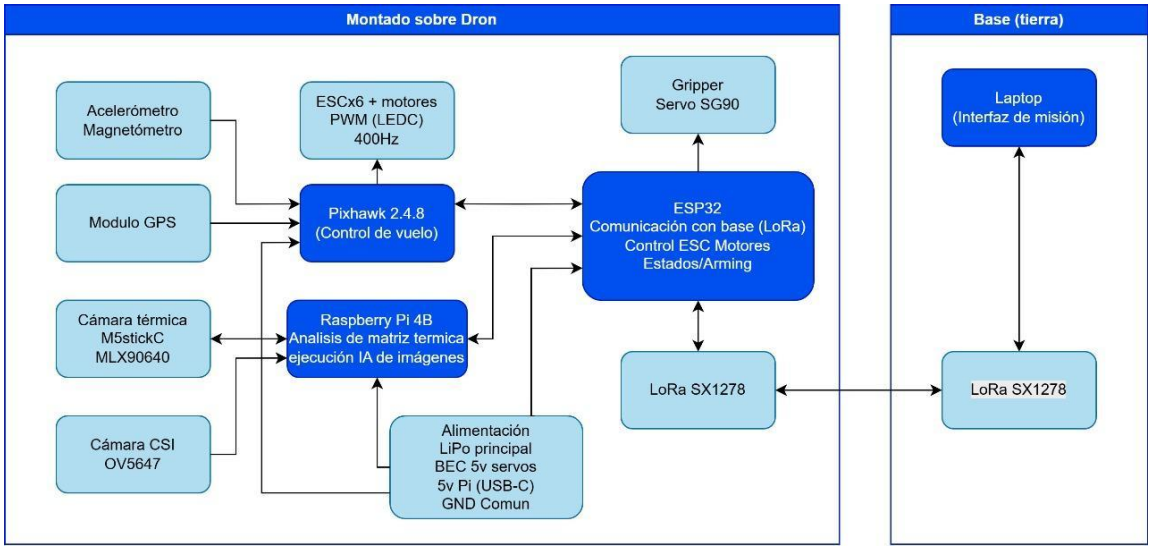
El diagrama de bloques resume el funcionamiento general del sistema, en donde la Raspberry Pi, la ESP32 y la Pixhawk actúan como los tres nodos principales de

procesamiento y control. En conjunto, estos módulos permiten ejecutar el flujo de misión: vuelo autónomo, captura de imágenes, análisis con IA, detección de eventos relevantes y transmisión de alertas hacia la estación base.

7.1 Arquitectura en Bloques del Sistema

El dron se organiza en torno a tres subsistemas principales:

1. **Subsistema de detección (visión RGB + térmica)**
    - Captura de imágenes RGB mediante la PiCam.
    - Lectura de la matriz térmica MLX90640.
    - Procesamiento local por la Raspberry Pi para identificar fuego y personas.
2. **Subsistema de control de misión**
    - La ESP32 recibe detecciones desde la Raspberry Pi.
    - Envía comandos MAVLink hacia Pixhawk.
    - Transmite alertas y coordenadas vía LoRa hacia la Ground Station.
3. **Subsistema de vuelo autónomo**
    - La Pixhawk ejecuta ArduCopter 4.6.2.
    - Controla motores y estabilización.
    - Navega entre waypoints mediante GPS.



Este diagrama permite visualizar cómo fluye la información a través del sistema: desde la captación de datos visuales, pasando por la inferencia de IA, hasta la transmisión de eventos y el control del vuelo autónomo.

7.2.1 Controladora de Vuelo Pixhawk 2.4.8

La Pixhawk constituye el núcleo de control aeronáutico del dron. Ejecuta el firmware ArduCopter 4.6.2, el cual proporciona:

- estabilización y control en tiempo real,
- control de velocidad de motores mediante los ESC,
- gestión de sensores inerciales y de navegación,
- modos de vuelo autónomos (GUIDED, AUTO, LOITER, RTL),
- Recepción de comandos vía MAVLink desde la ESP32.

La Pixhawk se encarga de la ejecución segura de la misión, asegurando que el dron despegue, navegue entre waypoints y aterrice de manera estable.

7.2.2 Módulo ESP32

La ESP32 funciona como unidad de control intermedia entre la Raspberry Pi y la Pixhawk.
Sus funciones principales son:

- interpretar las detecciones enviadas por la Raspberry Pi mediante UART,
- tomar decisiones de misión (detener vuelo, reenviar eventos, continuar ruta),
- transmitir eventos y coordenadas mediante el módulo LoRa SX1278,
- enviar comandos MAVLink a la Pixhawk (takeoff, set waypoint, RTL, land).

Este módulo representa el “cerebro lógico” del dron, encargado de supervisar y coordinar el comportamiento global durante la misión.

7.2.3 Raspberry Pi 4

La Raspberry Pi es la encargada de ejecutar el procesamiento inteligente del sistema.
Sus funciones incluyen:

- captura de imágenes RGB y térmicas,
- preprocesamiento de datos (reescalado, normalización, matrices térmicas),

- ejecución del modelo YOLOv8 en formato ONNX,
- detección de fuego y personas,
- Envío de alertas a la ESP32.

Su capacidad de cómputo permite implementar un esquema de edge computing, procesando la información en tiempo real sin necesidad de transmisión de video continuo.

7.2.4 Módulo LoRa SX1278

El módulo LoRa permite la comunicación de largo alcance entre el dron y la Ground Station, garantizando transmisión aún en zonas rurales o de vegetación densa. Características relevantes:

- bajo consumo energético,
- largo alcance,
- no apto para transmisión de video, solo datos discretos,
- ideal para enviar alertas con coordenadas.

LoRa actúa como el enlace principal para la notificación de detecciones relevantes.

7.2.5 Cámaras RGB e Infrarroja

PiCam (RGB)

- Captura imágenes en puntos específicos de la misión (waypoints).
- Permite identificar personas, fuego visible y humo.

MLX90640 (IR)

- Matriz térmica de 32×24 píxeles.
- Ideal para detectar variaciones de temperatura asociadas a incendios o presencia humana.
- Permite confirmación térmica de detecciones ambiguas en RGB.

7.3 Flujo de Misión del Sistema Completo

El proceso operativo del sistema comienza en la Ground Station y continúa a bordo del dron a través de los módulos ESP32, Raspberry Pi y Pixhawk. El flujo completo se describe a continuación.

1. Definición del polígono o área de interés

En la Ground Station, el operador selecciona un polígono geográfico que delimita la zona a inspeccionar, detallando, además, altura de vuelo, espaciado entre pasadas, espaciado entre detecciones y el comportamiento frente a la detección de un evento (continuar la misión navegando waypoints o volver a home).
El sistema genera automáticamente la secuencia de waypoints necesarios para cubrir el área, mostrando a medida que se realizan ajustes cómo se modifica la ruta y dichos waypoints.

2. Armado del dron y checklist de pre-vuelo

Antes de iniciar la misión, el equipo realiza las verificaciones correspondientes:

- conexión de baterías,
- verificación de sensores,
- bloqueo/armado de motores,
- comunicación Pixhawk–ESP32–Raspberry Pi–LoRa.

Cuando el sistema valida todos los parámetros, el dron queda listo para iniciar la misión.

3. Envío del plan de vuelo a la Pixhawk

La ESP32 recibe desde la Ground Station la lista de waypoints generada por trozos compatible con el tamaño máximo seguro de envío de datos por LoRa, concatena esos mensajes en orden y los transmite de a uno a la Pixhawk mediante MAVLink.
El dron queda configurado en modo GUIDED.

4. Despegue automático

La Pixhawk recibe desde la ESP32 el comando de despegue.
El dron asciende hasta la altura programada.

5. Navegación al primer waypoint

La Pixhawk guía el dron empleando GPS y control de altitud.

6. Detención y estabilización en waypoint

Al alcanzar cada punto, la ESP32 ordena al dron detenerse para realizar captura y análisis.

7. Captura de datos RGB + térmico

La Raspberry Pi adquiere imágenes RGB y una matriz térmica completa del MLX90640.

8. Procesamiento local con IA

La Raspberry Pi ejecuta el modelo YOLOv8 (formato ONNX) para identificar:

- fuego
- personas

9. Envío de detecciones a la ESP32

Si se detecta un evento, la Raspberry Pi informa a la ESP32 mediante UART con la etiqueta de la detección y nivel de confianza. Si ocurre una detección del tipo persona, soltara el paquete que lleva el gripper.

10. Transmisión de alerta a la Ground Station

La ESP32 envía vía LoRa:

- tipo de detección,
- coordenadas del dron,
- timestamp y meta-información.

Esto permite al operador visualizar el evento en tiempo real sobre el mapa y tomar acción respecto del problema detectado.

11. Continuación de la misión

Una vez finalizado el análisis, la ESP32 ordena reanudar el vuelo hacia el siguiente waypoint o bien regresar a la base, según como haya seteado el programa el operador.

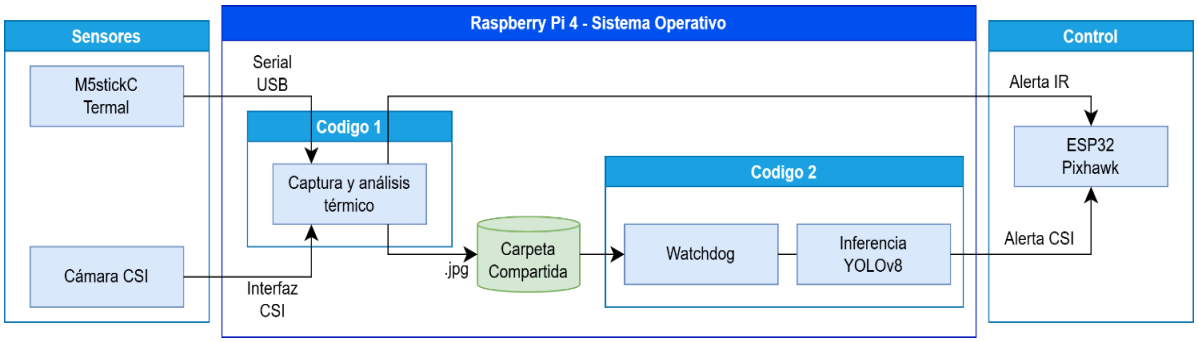
12. Finalización de la misión y retorno automático

Al completar la ruta, la ESP32 envía la coordenada de home a la Pixhawk como si fuese un waypoint más, y luego ejecuta el aterrizaje.

8. Desarrollo de los Subsistemas

El sistema desarrollado se compone de múltiples subsistemas que actúan de manera coordinada para ejecutar la misión autónoma. Cada uno cumple una función específica dentro del flujo percepción–procesamiento–decisión–acción. A continuación, se describen los subsistemas principales, sus características técnicas y su integración dentro de la arquitectura general.

8.1 Sistema de detección autónoma de personas y/o fuego



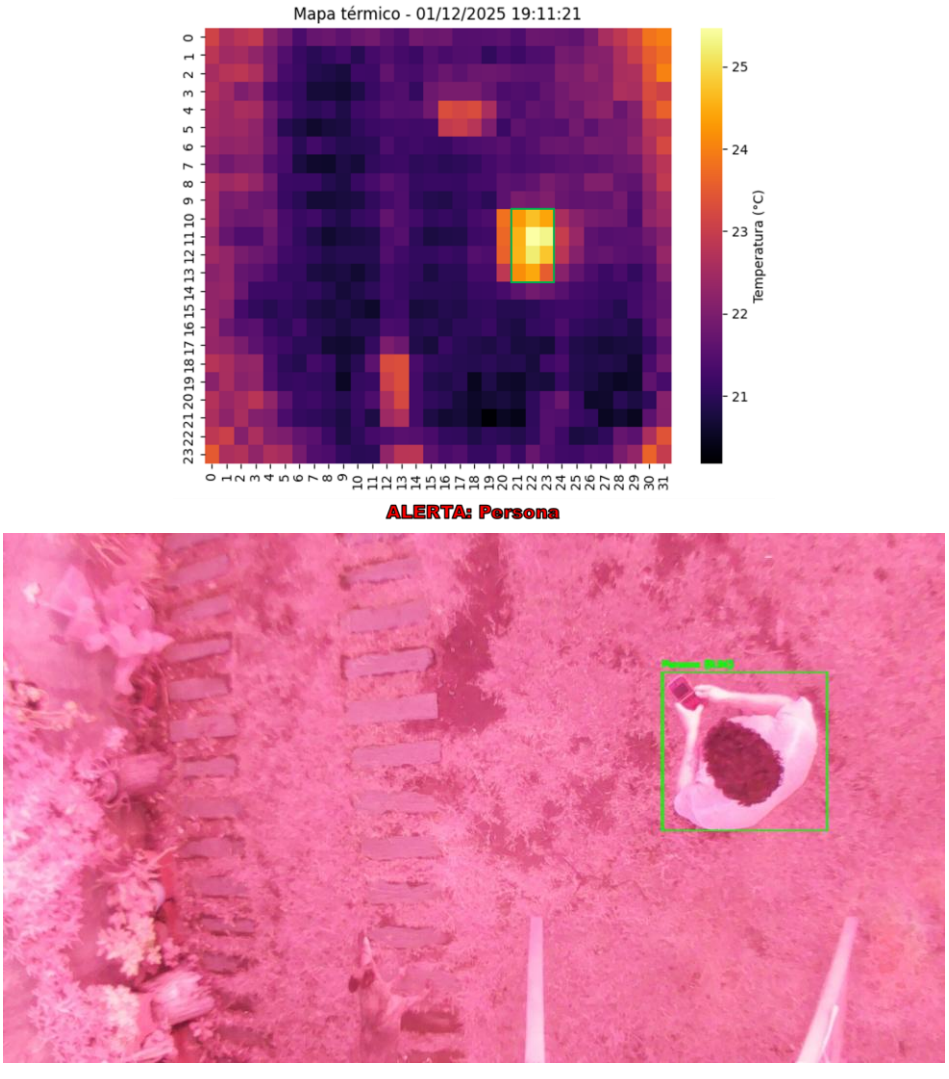
Para la implementación del sistema de visión y detección, se diseñó una arquitectura de software modular distribuida en tres bloques lógicos principales que operan de manera cooperativa. Esta estructura permite separar las tareas de adquisición de datos en tiempo real del procesamiento computacional intensivo, asegurando que el procesamiento pesado de las imágenes no detenga la ejecución del resto del sistema. Esto permite seguir recibiendo datos de los sensores y mantener la conexión fluida con el dron sin tener que esperar a que finalice cada inferencia de la inteligencia artificial.

Adicionalmente, esta división operativa ofrece una ventaja significativa en la gestión del software: el aislamiento de entornos. Cada código se ejecuta en su propio entorno virtual independiente, lo que evita conflictos entre versiones incompatibles de las mismas librerías. De esta manera, cada subsistema cuenta con las dependencias exactas que necesita, sin que las actualizaciones requeridas por el modelo de IA comprometan la estabilidad de los controladores de los sensores.

El flujo de información comienza en el nivel de hardware con el firmware del M5StickC, el cual actúa como un dispositivo de adquisición dedicado para el sensor térmico. En el siguiente nivel, ejecutándose sobre la Raspberry Pi, el Código 1 (Módulo de captura y análisis térmico) funciona como el orquestador principal: solicita los datos térmicos,

captura las imágenes visuales RGB y realiza un primer filtrado de alertas basado en temperatura.

Finalmente, la integración entre la visión clásica y la inteligencia artificial se logra mediante un esquema asíncrono basado en el sistema de archivos. Cuando el Código 1 deposita una imagen en la carpeta compartida, se dispara automáticamente al Código 2 (Análisis con IA). Este último módulo procesa la imagen con el modelo YOLOv8 y, en caso de confirmar una detección positiva, emite las alertas correspondientes al sistema de control, cerrando así el ciclo de vigilancia autónoma detallado en el siguiente esquema global.



8.1.1 Descripción del firmware del M5StickC para adquisición térmica

El M5StickC utiliza su módulo ESP32-PICO interno para capturar y transmitir el cuadro térmico proveniente del sensor MLX90640. El firmware implementa un esquema de operación por solicitud (modo pull), en el cual la captura no se realiza de manera continua, sino únicamente cuando la Raspberry Pi envía el comando correspondiente a través del enlace serial USB. Esta lógica permite minimizar el procesamiento interno del dispositivo y evita generar datos innecesarios cuando no están siendo requeridos.

Inicializar pantalla, serial, bus I2C y sensor MLX90640

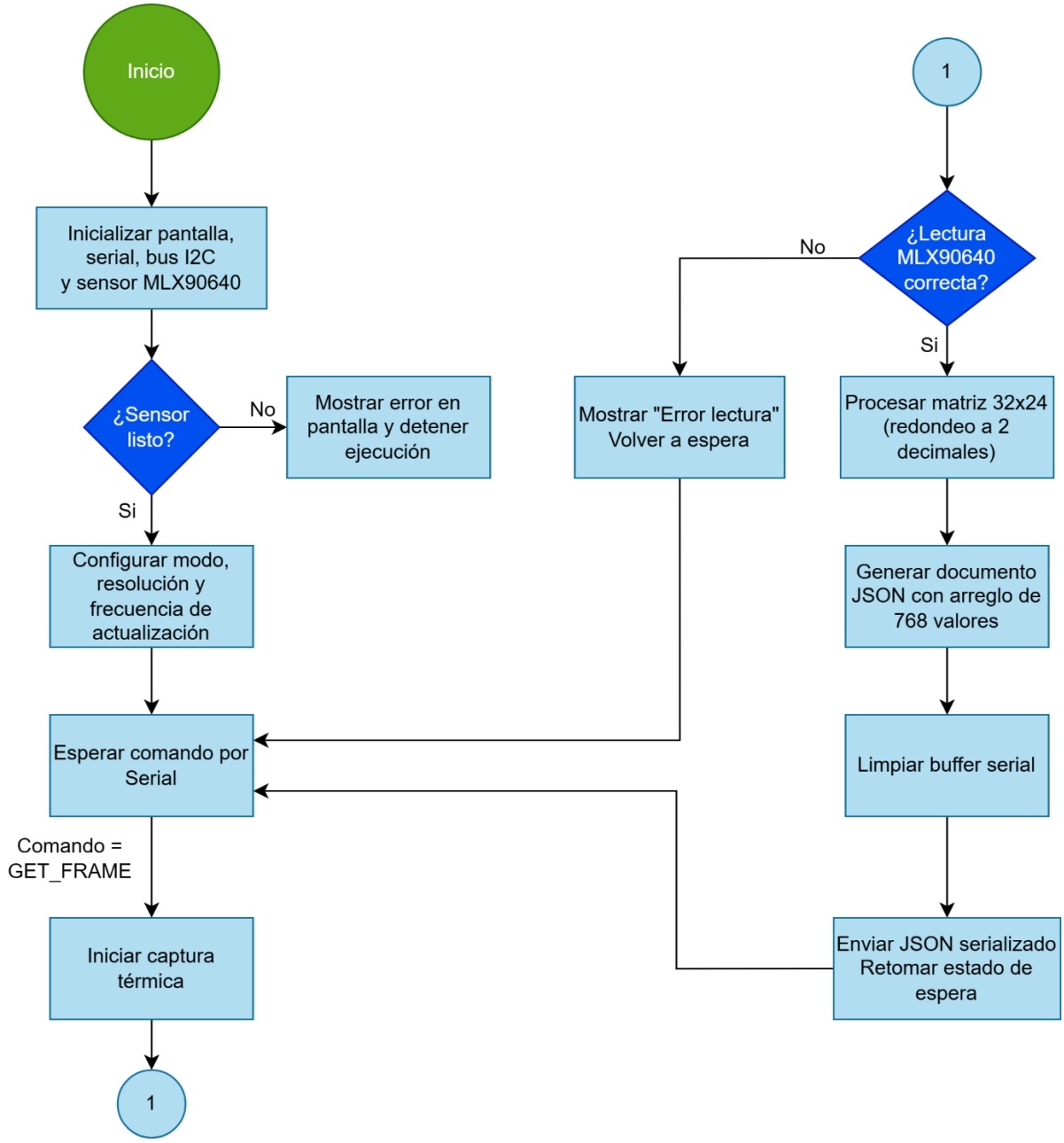
Al iniciar, el sistema configura la pantalla del M5StickC, el bus I2C utilizado para comunicarse con el sensor y la interfaz serial que enlaza el módulo con la Raspberry Pi. También se inicializa el MLX90640 con los parámetros de funcionamiento seleccionados: modo intercalado, resolución de 18 bits y una frecuencia de actualización de 2 Hz. En caso de que el sensor no responda, el dispositivo detiene su ejecución y muestra un mensaje de error para evitar el envío de información incorrecta.

Iniciar captura térmica

Una vez completada la configuración, el M5StickC permanece a la espera de un comando entrante. Si la Raspberry Pi envía la cadena "GET_FRAME", el firmware inicia la lectura completa del sensor y obtiene la matriz de 768 valores de temperatura. Luego, estos datos se redondean a dos decimales y se empaquetan en un documento JSON que contiene un único arreglo con todas las mediciones, manteniendo el orden original de la matriz del sensor.

Limpiar buffer serial y enviar JSON

Para asegurar una transmisión limpia, el firmware limpia el buffer serial antes de enviar la respuesta y encapsula el mensaje JSON entre dos delimitadores (#START y #END). Esto permite que la Raspberry Pi detecte con precisión el inicio y el fin del paquete, incluso si el flujo serial contiene ruido o interrupciones temporales. Finalmente, el dispositivo actualiza la pantalla para indicar el envío y vuelve al estado de espera hasta recibir el próximo comando.


Inicio
Inicializar pantalla, serial, bus I2C y sensor MLX90640
¿Sensor listo?
No
Mostrar error en pantalla y detener ejecución
Si
Configurar modo, resolución y frecuencia de actualización
Esperar comando por Serial
Comando = GET_FRAME
Iniciar captura térmica
1
1
¿Lectura MLX90640 correcta?
No
Si
Mostrar "Error lectura" Volver a espera
Procesar matriz 32x24 (redondeo a 2 decimales)
Generar documento JSON con arreglo de 768 valores
Limpiar buffer serial
Enviar JSON serializado Retomar estado de espera

8.1.2 Descripción técnica del módulo de captura y análisis térmico

El módulo de captura y análisis térmico, implementado en el programa "codigo1_camaras_v2.py", constituye el núcleo funcional del subsistema de visión del dron, integrando captura de imágenes, análisis térmico en tiempo real y comunicación bidireccional con los dispositivos embarcados (M5StickC con sensor térmico y ESP32 para las órdenes de vuelo). En su diseño se prioriza la fiabilidad operativa, la trazabilidad de eventos y la capacidad de ejecutar detección autónoma sin intervención del operador.

Inicialización Serial/Carpetas/Cámara

El código se estructura en torno a una clase de configuración centralizada que define todos los parámetros relevantes para la ejecución: puertos seriales, velocidades de comunicación, características de la cámara CSI, umbrales térmicos para la detección de personas e incendios, y rutas de almacenamiento. Esta centralización permite mantener un control claro sobre los elementos críticos del sistema.
Una vez iniciada la aplicación, el sistema crea y valida automáticamente las carpetas de trabajo y habilita un esquema de registro persistente de métricas y eventos, pensado para documentar tanto condiciones normales de funcionamiento como situaciones de error.

Modo de toma de imágenes

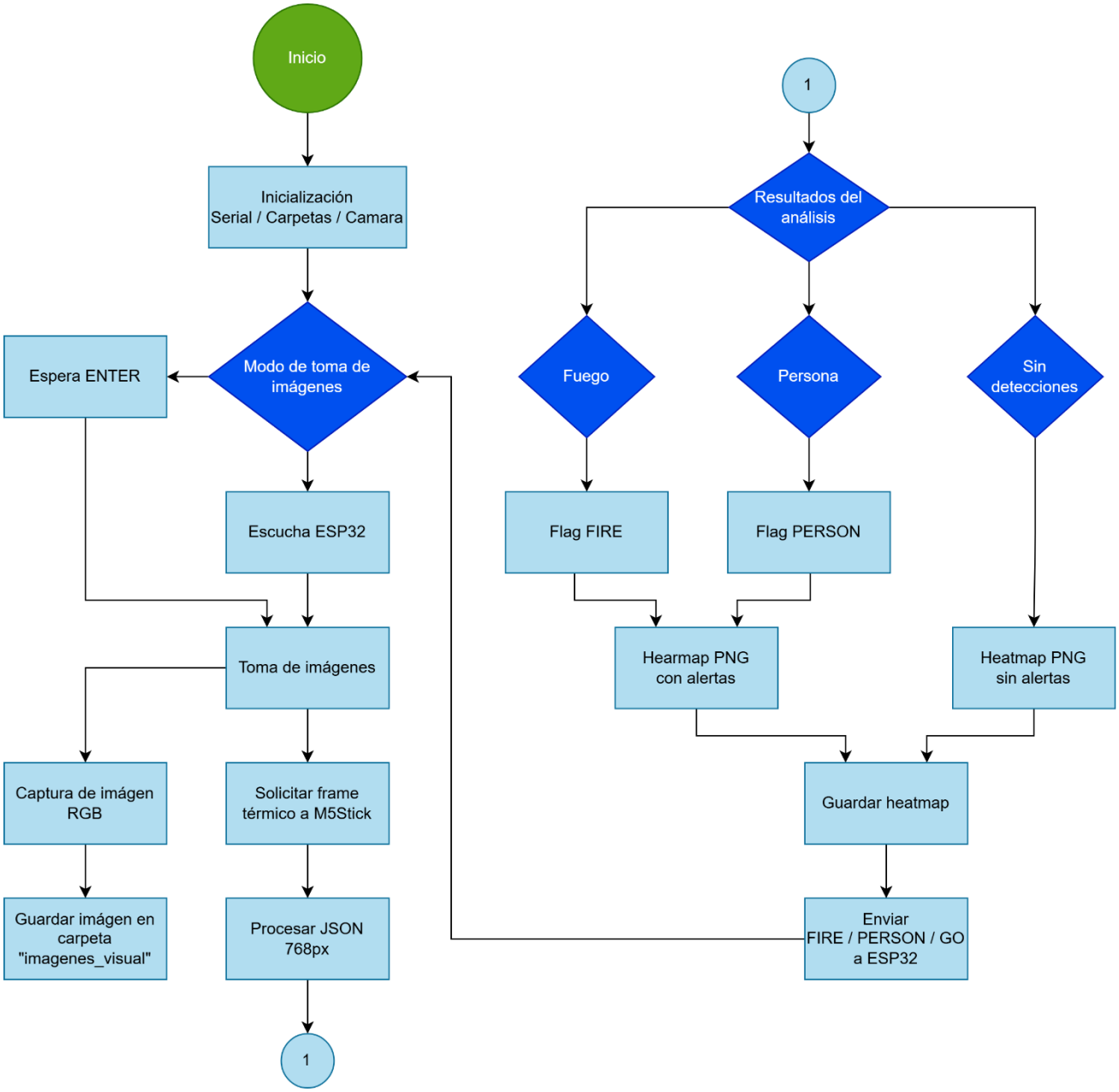
En cuanto a la adquisición sensorial, el módulo establece dos canales de comunicación independientes. Por un lado, se conecta al M5StickC con su sensor MLX90640 integrado, responsable de transmitir matrices térmicas de 32×24 píxeles, encapsuladas en formato JSON. Por el otro, se vincula con la ESP32, que puede iniciar el proceso de captura mediante el comando STABLE o recibir mensajes de alerta cuando el sistema identifica una anomalía térmica. Ambos enlaces operan con tolerancia a fallos y emplean hilos independientes cuando es necesario, garantizando que la recepción de comandos no interfiera con el procesamiento de imágenes.
El procesamiento térmico constituye el corazón computacional del módulo. Cada frame recibido es sometido a un filtrado de mediana para reducir ruido y mejorar la coherencia espacial de la matriz. Luego se ejecuta un análisis basado en segmentación por umbral adaptativo: el sistema determina dinámicamente la temperatura de fondo y la contrasta con rangos característicos para personas y focos de incendio. Los rangos de temperatura y área utilizados para discriminar entre personas y focos de incendio se definen durante

la inicialización del sistema y provienen de pruebas de campo realizadas previamente, lo que permite adaptar el algoritmo a condiciones reales.
En paralelo a la recepción y procesamiento del frame térmico, el software coordina la toma de imágenes RGB con una cámara OV5647 de 8MP y las mismas son guardadas con el mismo sello temporal que la térmica para facilitar su análisis conjunto. Las imágenes RGB se almacenan en la carpeta “imágenes_visual” para que sean relevadas por el código que analizará a las mismas utilizando inteligencia artificial.

Resultados del análisis térmico

Se utilizan técnicas de etiquetado de componentes conectados para identificar regiones candidatas, evaluando su área y delimitándolas con recuadros que representan las detecciones en la matriz térmica. Esta metodología permite operar incluso en condiciones de ruido térmico elevado y evita falsos positivos derivados de objetos ambientales cálidos, pero no relevantes.
Una vez identificadas las regiones de interés, el módulo genera una imagen térmica interpretada mediante un mapa de calor (heatmap), señalando visualmente las detecciones mediante los recuadros codificados por color. Finalmente, cuando el sistema determina la presencia de una persona o un posible foco ígneo, se activa el protocolo de alerta hacia la ESP32, enviando una cadena JSON minimalista que sirve como disparador para acciones de navegación o marcación de eventos en el dron.

El ciclo de operación permite dos modos de activación: captura manual mediante teclado o activación remota desde la ESP32. Esto habilita tanto pruebas en laboratorio como funcionamiento autónomo durante un vuelo real. Cada ciclo concluye retornando un estado de éxito o error y dejando registrada toda la actividad en los archivos de log correspondientes.
En conjunto, este módulo implementa una arquitectura de software orientada a la confiabilidad y la detección temprana, combinando visión térmica, control asincrónico de hardware y generación de evidencia visual. Es el componente responsable de transformar datos brutos del sensor infrarrojo en información estructurada y accionable, funcionando como la primera etapa de interpretación del entorno en el sistema completo del dron.


Inicio
Inicialización
Serial / Carpetas / Camara
Modo de toma de imágenes
Espera ENTER
Escucha ESP32
Toma de imágenes
Captura de imágen RGB
Guardar imágen en carpeta "imagenes_visual"
Solicitar frame térmico a M5Stick
Procesar JSON 768px
1
1
Resultados del análisis
Fuego
Persona
Sin detecciones
Flag FIRE
Flag PERSON
Hearmap PNG con alertas
Heatmap PNG sin alertas
Guardar heatmap
Enviar FIRE / PERSON / GO a ESP32

8.1.3 Descripción técnica del módulo de análisis de imágenes con inteligencia artificial

El segundo módulo del sistema, codigo2_ia_v3.py, cumple el rol de núcleo inteligente del pipeline general. Mientras el Código 1 se ocupa de adquirir imágenes RGB y almacenarlas de manera ordenada, este módulo funciona como un observador activo que espera la llegada de esos archivos para someterlos a un análisis automático mediante un modelo de detección basado en YOLOv8 exportado a ONNX. De esta manera, Código 2 permite cerrar el ciclo completo: captura, procesamiento inteligente y emisión de alertas al dron o al sistema embebido encargado de actuar en consecuencia.

Un aspecto clave del diseño es que todo el análisis de imágenes se realiza de manera local, sin depender de servicios externos ni de una conexión estable a Internet. Esto responde a una limitación propia del sistema: la comunicación entre el dron y la base se establece mediante LoRa, un protocolo de largo alcance, pero muy baja tasa de transferencia, insuficiente para enviar imágenes o ejecutar análisis de imágenes de manera remota. Al ejecutar la inteligencia artificial directamente en la Raspberry embarcada sobre el dron, mismo puede operar de forma autónoma y generar alertas inmediatas sin necesidad de enlaces WiFi o satelitales, lo que mejora la robustez del sistema y lo vuelve apto para entornos aislados o con conectividad limitada.

Configurar sistema (carpetas, paths, clases)

El programa inicia estableciendo una configuración centralizada que define los directorios relevantes, los parámetros y el modelo a utilizar. A diferencia del primer módulo, aquí el objetivo no es solo verificar y organizar carpetas, sino también garantizar un entorno de ejecución preparado para alojar tanto las imágenes procesadas como los registros de funcionamiento.
Esta etapa crea las carpetas necesarias, carga el nombre del modelo ONNX y fija los umbrales mínimos de confianza para descartar predicciones irrelevantes. Al igual que Código 1, este módulo también mantiene comunicación con la ESP32, enviando alertas de detección o indicaciones para continuar la exploración.

Carga de modelo YOLOv8 ONNX

El corazón del módulo se concentra en la clase AnalizadorIA, encargada de cargar el modelo YOLOv8 en formato ONNX y llevar adelante el flujo completo: desde la

preparación de la imagen, pasando por el análisis del modelo, hasta la interpretación final de las detecciones.
Durante el proceso, las imágenes se redimensionan, normalizan y convierten al formato requerido por el modelo. Luego, las predicciones se reescalan al tamaño original y se construyen los recuadros delimitadores junto con sus etiquetas correspondientes.
Una decisión de diseño destacada es la definición explícita de las clases (“Persona”, “Fuego”, “Humo”), agrupando fuego y humo bajo una misma categoría de alerta (FIRE) para simplificar la comunicación operativa con el dron.

Iniciar vigilancia (watchdog en carpeta)

Sobre este analizador se apoya el componente responsable de reaccionar a los eventos del sistema de archivos. Utilizando la biblioteca watchdog, el módulo monitoriza en tiempo real la carpeta de salida del Código1. Cada vez que aparece una nueva imagen, el código procesa el archivo, lo somete al análisis del modelo y genera una imagen anotada con las detecciones encontradas. Esta imagen se guarda con un registro de fecha y hora preciso, lo que facilita la trazabilidad y el alineamiento temporal con los registros del resto del sistema.

Resultados del análisis

La lógica de alertas también reside en este módulo. Cuando no se detectan objetos relevantes, el sistema informa a la ESP32 mediante el mensaje “GO”, manteniendo el flujo de confirmaciones del estado general. Por el contrario, si se identifican personas o fuego, se emiten alertas específicas (“PERSON” o “FIRE”), habilitando al dron a reaccionar de acuerdo con la situación.

Finalmente, el ciclo completo del sistema se encapsula en la clase SistemaIA. Una vez en ejecución, el módulo entra en un estado de escucha permanente y solo se detiene ante una interrupción explícita del usuario. Al finalizar, todos los recursos se liberan de manera ordenada, asegurando un cierre limpio tanto de la conexión serial como del proceso del watchdog.
En conjunto, este módulo observa, interpreta y actúa, conectando el análisis visual avanzado con una lógica operacional liviana pero efectiva. Su diseño apunta a la fiabilidad y a la eficiencia, respetando el flujo general del proyecto y complementando las capacidades del Código1 para conformar un sistema coherente de vigilancia automática.

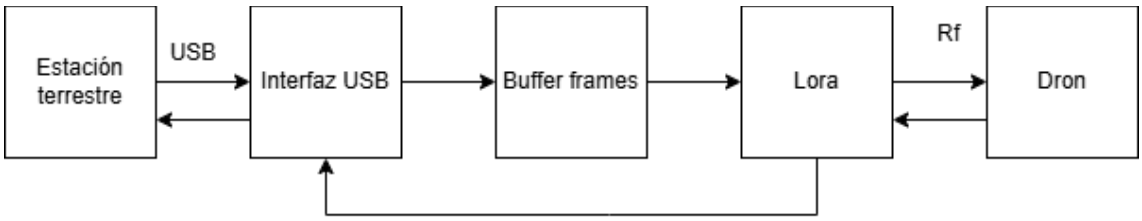
8.4 Sistema de Comunicaciones

El subsistema de comunicaciones combina LoRa para alertas y telemetría entre Pixhawk y Ground Station. Para que el programa perteneciente a la estación terrestre pueda comunicarse por LoRa, es necesario el uso de un adaptador USB serial a LoRa, utilizando otro ESP32 con LoRa sx1278

LoRa (SX1278)

- Enlace de largo alcance.
- Baja tasa de datos.
- Muy bajo consumo de corriente.
- Utilizado para enviar eventos: detección, tipo, coordenadas.
- Ideal para zonas sin infraestructura.

A continuación, el diagrama en bloques del programa



Telemetría (MAVLink)

- Comunicación entre ESP32 y Pixhawk (coordenadas)
- Comandos enviados:
    - takeoff
    - set waypoint
    - RTL
    - land
- Lectura de estado de vuelo, navegación y posición GPS.

Formato JSON

Se eligió utilizar este formato para el envío de datos entre los subsistemas para asegurar compatibilidades de tipos entre diferentes lenguajes de programación, como son C y Python.

Los eventos enviados por LoRa se transmiten en formato JSON, por ejemplo:

{
"event": "fire",
"lat": -34.123456,
"lon": -58.987654,
"time": "2025-11-27 18:42:10"
}

Ground Station

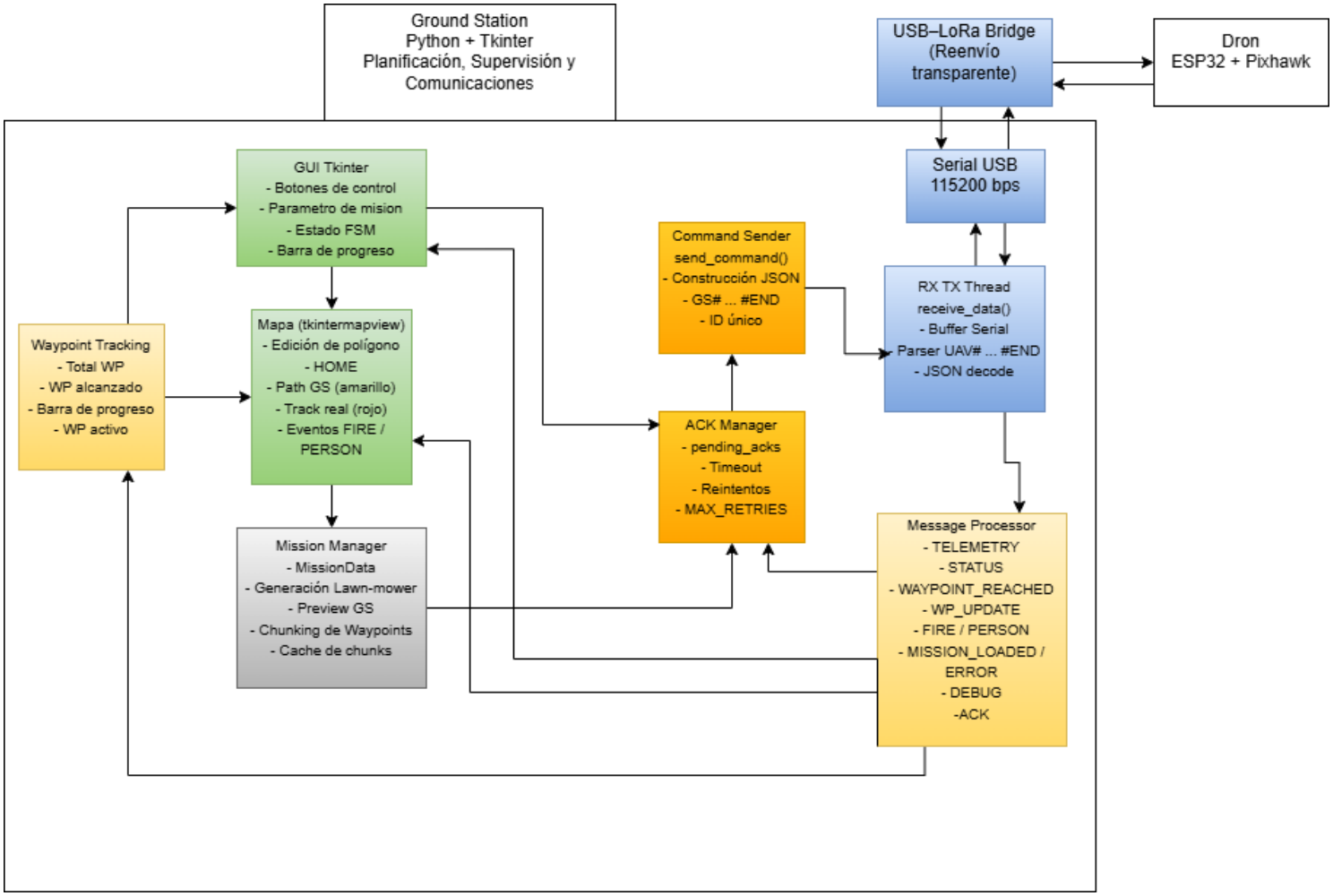
- Recibe JSON por LoRa para telemetría y eventos.
- Muestra detecciones en mapa.
- Registra historial de alertas.
- Permite seleccionar polígonos, generar waypoints y configurar la misión.

La Ground Station se diseñó con una arquitectura modular, separando la interfaz gráfica, la lógica de misión, el procesamiento de mensajes y la capa de comunicaciones.

La confiabilidad del enlace se garantiza mediante un esquema de conocimiento de recibo y reintentos (ACK), implementado en hilos independientes, mientras que la actualización de la interfaz se realiza de forma segura mediante el bucle principal de Tkinter.

Esta organización permite una supervisión clara del estado del dron, una planificación flexible de misiones y una visualización precisa de la información en tiempo real.

Arquitectura del programa de Ground Station



8.5 Navegación Autónoma y Control de Vuelo

El sistema utiliza la controladora Pixhawk ejecutando ArduCopter 4.6.2.

Pixhawk

- Estabilización
- Control de motores
- Gestión de sensores
- Navegación por GPS
- Ejecución de modo de vuelo guided.

MAVLink (entre ESP32 y Pixhawk)

La ESP32 envía:

- Pedido de entrega de datos específicos (telemetría y batería).
- Seteo de modo de vuelo guided
- Armado y desarmado de motores.
- Coordenadas y altura de los waypoint.

Y recibe:

- Posición GPS, altitud barométrica, tensión de la batería.
- Estado de vuelo (confirmación modo de vuelo guided y armado/desarmado)

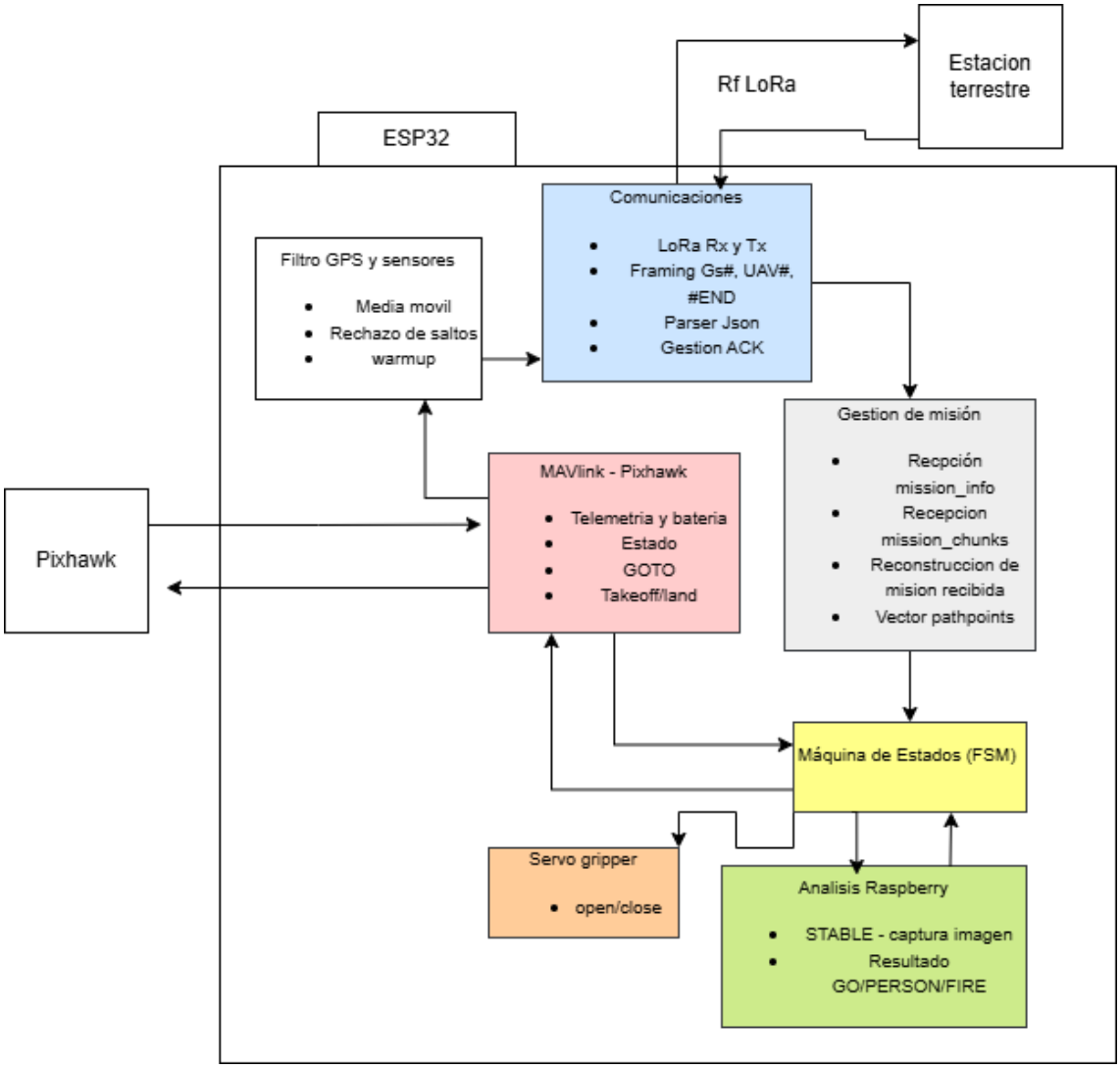
Modos de vuelo utilizados

GUIDED: comandos de navegación puntuales, se envía latitud, longitud y altura y dirige al dron a esa posición.

Relación con el flujo de misión

Este subsistema ejecuta físicamente la ruta definida desde la Ground Station, coordinado con los subsistemas de percepción e IA para detenerse, analizar el entorno y continuar cuando corresponda.

Arquitectura del programa del ESP32 para el vuelo



9. Integración del Sistema Completo

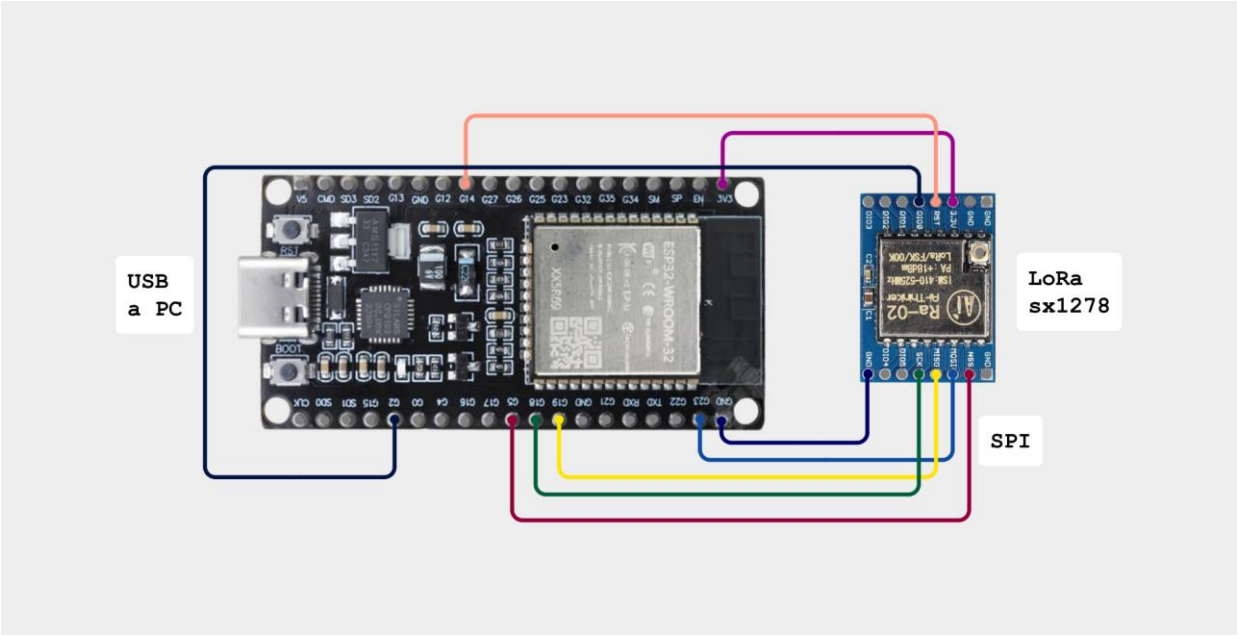
La integración del sistema consiste en unir todos los subsistemas desarrollados —percepción, procesamiento, comunicaciones y control de vuelo— en un flujo de operación continuo y coordinado. Este capítulo describe tanto la integración física (conexiones y arquitectura eléctrica) como la integración lógica, que define cómo interactúan la Raspberry Pi, la ESP32 y la Pixhawk durante la misión autónoma.

9.1 Integración Física del Sistema - Esquemas eléctricos

Detalle de componentes y conexiones según tipo de comunicación.

9.1.1 Enlace USB - LoRa

Como ya se mencionó previamente, la comunicación con el dron se hace por medio de LoRa y debido a que no es un módulo nativo que pueda encontrarse en una computadora, fue necesario crear un enlace con conexión USB serial a protocolo LoRa.

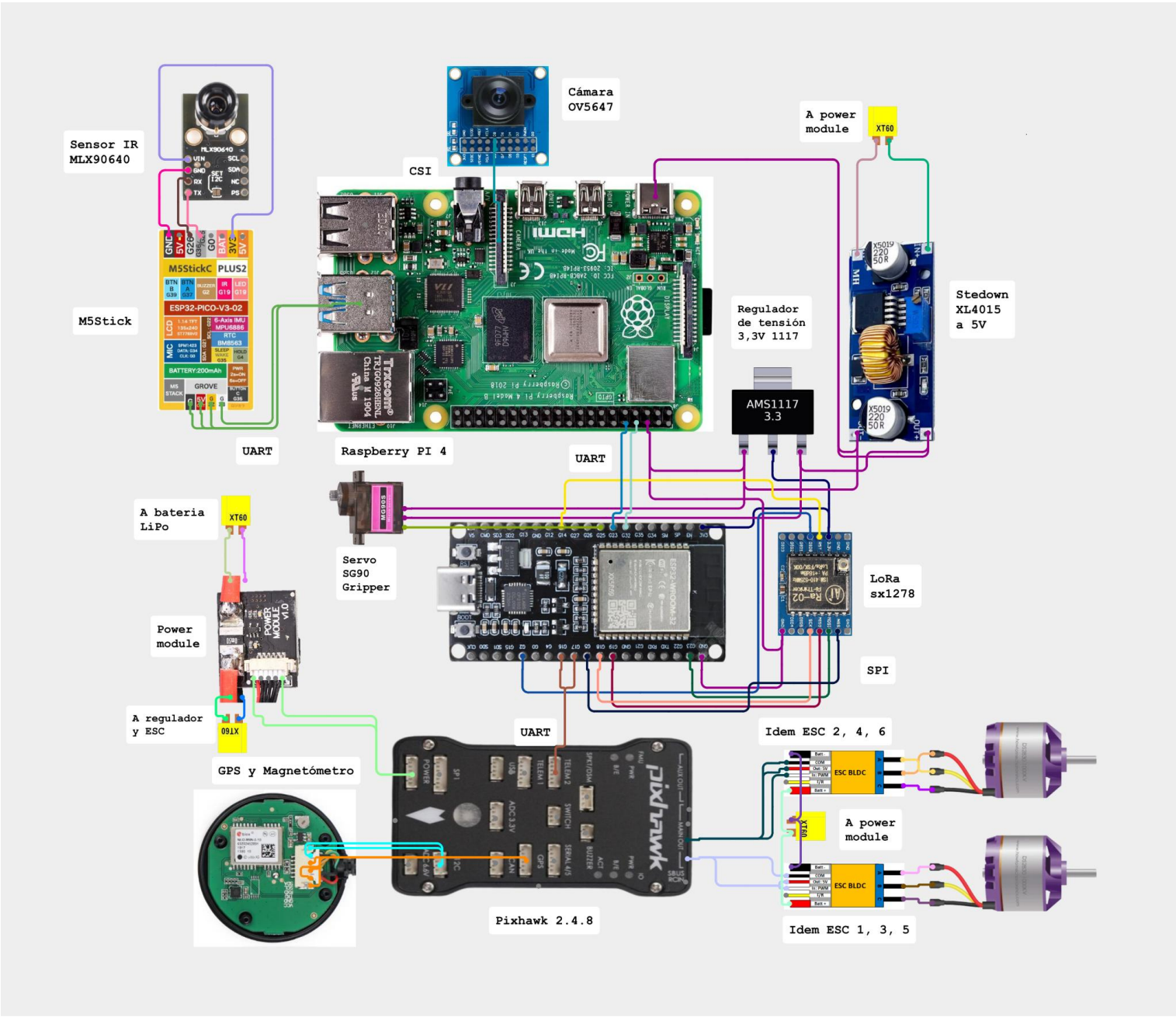


La conexión del módulo LoRa sx1278 se realiza por SPI.

Pines requeridos:

Conexión ESP32 - LoRa sx1278	
Lado ESP32	Lado sx1278
3,3 V	3,3 V
GND	GND
G14	RST
G2	DIO0
G5	NSS
G23	MOSI
G19	MISO
G18	SCK

9.1.2 Dron



Montado en el dron tenemos los tres subsistemas antes descritos, percepción, comunicación y navegación.

Detallado de conexiones y pines requeridos:

Circuito de Potencia	
Conector XT60 batería LiPo	Conector entrada XT60 Power module
Conector salida XT60 Power module	Conector XT60 dron
Alimentación dron	
Conector XT60 dron	Vin XL4015
	6 x ESC 30A
Step down XL4015 5VCC	
Vout XL4015 - 5VCC	Vin LD1117
	USB C alimentación Raspberry pi
Regulador lineal LD1117 3,3 VCC	
Vout LD1117 - 3VCC	ESP32
	lora sx1278

En lo referido al ESP32 nos encontramos con su alimentación de 3,3v que comparte con el módulo sx1278, con la conexión del servo del gripper y las conexiones UART que se dirigen a la Raspberry Pi (UART1) y a la controladora Pixhawk (UART2).

Conexión ESP32 - LoRa sx1278	
Lado ESP32	Lado sx1278
3,3 V	3,3 V
GND	GND
G14	RST
G2	DIO0
G5	NSS
G23	MOSI
G19	MISO
G18	SCK
Conexión ESP 32 - Raspberry Pi 4	
Lado ESP32	Lado Raspberry Pi 4
G33 - RX	G8 - TX
GND	GND
G32 - TX	G10 - RX
Conexión ESP32 - servo sg90	
Lado ESP32	Lado servo sg90
G25	Señal
GND	GND
-	5v → a red 5v XL4015

Conexión ESP32 - Pixhawk	
Lado ESP32	Lado Pixhawk
GND	TELEM 2 GND
G16 - RX	TELEM 2 TX
G17 - TX	TELEM 2 RX

Respecto de Raspberry Pi, nos encontramos con la conexión UART compartida a ESP32 y las conexiones a las cámaras y alimentación.

Conexión Raspberry Pi - cámara ov5647	
Lado Raspberry Pi 4	Lado Cámara
Puerto CSI	Puerto CSI
Conexión Raspberry Pi 4 - ESP 32	
Lado Raspberry Pi 4	Lado ESP32
G8 - TX	G33 - RX
GND	GND
G10 - RX	G32 - TX
Conexión Raspberry Pi - IR MLX90640 con M5Stick	
Lado Raspberry Pi 4	Sensor IR
Puerto USB 3.0	Puerto USB 3.0

EL conexionado de Pixhawk comienza con la alimentación del módulo vía el power module (voltímetro y amperímetro de batería), incluye la conexión UART con el ESP32, los sensores externos magnetómetro HMC5883L y GPS M8N por I2C y la conexión a los ESC, de estos últimos solo se representan dos de los seis existentes en modo de practicidad; a tomar en cuenta que los ESC de los motores 1, 3 y 5 deben ir conectados de forma lineal con los motores y los ESC de los motores 2, 4 y 6 deben ir con una fase invertida, con el fin de que giren en sentido antihorario.

Conexión Pixhawk - power module	
Lado Pixhawk	Lado power module
Puerto power	Salida a puerto power
Conexión Pixhawk - GPS y Magnetómetro	
Lado Pixhawk	Lado GPS y Magnetómetro
Puerto I2C	Magnetómetro
Puerto GPS	GPS
Conexión Pixhawk - ESP32	
Lado Pixhawk	Lado ESP32
TELEM 2 TX	G16 RX
TELEM 2 RX	G17 TX
TELEM 2 GND	GND
Conexión Pixhawk - ESC	
Lado Pixhawk	Lado ESC
MAIN 1	PWM ESC

MAIN 2	PWM ESC INVERTIDO
MAIN 3	PWM ESC
MAIN 4	PWM ESC INVERTIDO
MAIN 5	PWM ESC
MAIN 6	PWM ESC INVERTIDO

9.2 Integración Lógica (Raspberry Pi ↔ ESP32 ↔ Pixhawk)

La integración lógica se basa en la comunicación entre los tres módulos principales del sistema.

Raspberry Pi → ESP32

- Envía detecciones por UART (fire, person).
- Espera autorización para iniciar captura.

ESP32 → Pixhawk

- Envía comandos MAVLink: set mode, set waypoint, request telemetry
- Recibe telemetría, estado de vuelo.

ESP32 → Ground Station (LoRa)

- Envía eventos detectados junto con coordenadas y timestamp.
- Envía telemetría.

Pixhawk → ESP32

- Notifica fallas o pérdida de GPS.
- Envía telemetría

La ESP32 actúa como el orquestador encargado de sincronizar detección, navegación y comunicaciones.

9.3 Integración Funcional: Secuencia de Waypoints

La Ground Station genera una misión automática a partir del polígono seleccionado.
La ESP32 recibe y transmite estos waypoints a la Pixhawk usando MAVLink.

Durante la misión autónoma:

1. La Pixhawk ejecuta navegación GPS entre waypoints.
2. La ESP32 supervisa distancia y confirma llegada.
3. Al llegar, la misión entra en estado “ANALIZANDO”.

La secuencia se repite hasta completar todos los waypoints.

9.4 Lógica de Detención, Captura, Procesamiento y Continuación

Esta es la integración más importante del sistema.
Define cuándo se debe detener el dron y cómo interactúan los subsistemas.

La lógica es:

1. Llegada a waypoint
   La Pixhawk informa a la ESP32.
2. Orden de detención
   La ESP32 envía un comando para mantener el dron estable (LOITER).
3. Captura de imágenes
   La ESP32 indica a la Raspberry Pi que capture RGB + térmica.
4. Procesamiento con IA
   La Raspberry Pi ejecuta YOLOv8 y análisis térmico.
5. Generación de evento (si corresponde)
   Si se detecta un objeto relevante, la Raspberry Pi envía un mensaje a la ESP32.
6. Transmisión de alerta
   La ESP32 envía por LoRa un JSON con:
   - tipo de detección,
   - precisión,
   - hora,
   - coordenadas exactas.
7. Reanudación del vuelo
   La ESP32 envía a la Pixhawk el comando para avanzar al próximo waypoint.

8. Fin de misión
   Cuando se completan todos los puntos, se envía RTL para regresar al origen.

Este flujo garantiza que la misión siga un ciclo coherente:

volar → analizar → decidir → notificar → continuar

10. Pruebas y Resultados

El presente capítulo describe las pruebas realizadas sobre el prototipo desarrollado y los resultados obtenidos durante su funcionamiento. Las pruebas se orientaron a validar el desempeño del sistema en condiciones reales o representativas, evaluando los subsistemas de detección térmica, visión RGB, inteligencia artificial, comunicaciones, vuelo autónomo y operación general del dron.

10.1 Resultados térmicos

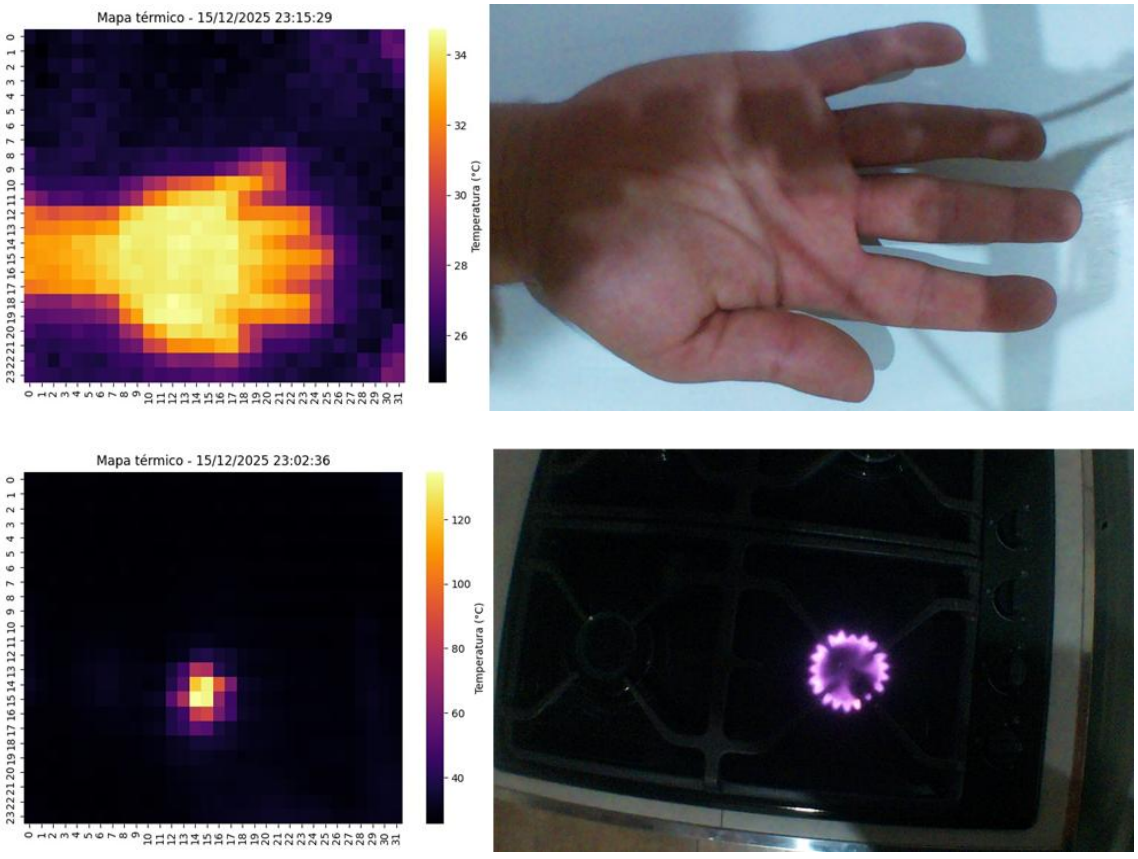
Se realizaron pruebas utilizando la cámara térmica basada en el sensor MLX90640, montada sobre el dron y operada desde la Raspberry Pi.
Las imágenes térmicas obtenidas fueron procesadas mediante interpolación para mejorar la visualización de la matriz original de baja resolución.

Durante las pruebas se analizaron distintos escenarios:

- Presencia de personas en el campo de visión.
- Escenarios sin personas ni fuentes térmicas relevantes.
- Detección de focos de calor localizados (hotspots).

Se validó el umbral térmico seleccionado para la detección, observándose que valores superiores al entorno permitieron identificar correctamente la presencia de una persona o fuente caliente, minimizando falsas detecciones provocadas por el suelo o elementos del entorno.

En la siguiente figura se muestra un ejemplo de imagen térmica interpolada con detección de hotspot.



10.2 Pruebas RGB

Las pruebas del sistema de visión RGB se realizaron mediante capturas tomadas por la cámara Raspberry Pi (OV5647) durante el recorrido del dron por distintos waypoints.

En cada punto de interés, el dron ejecutó la secuencia:

1. Detención en el waypoint.
2. Captura de imagen RGB.
3. Análisis local mediante el modelo de inteligencia artificial.

Las pruebas se realizaron bajo diferentes condiciones de iluminación, incluyendo luz diurna directa y escenarios con sombras parciales.
Las imágenes capturadas fueron utilizadas como entrada para la detección de personas y fuego.

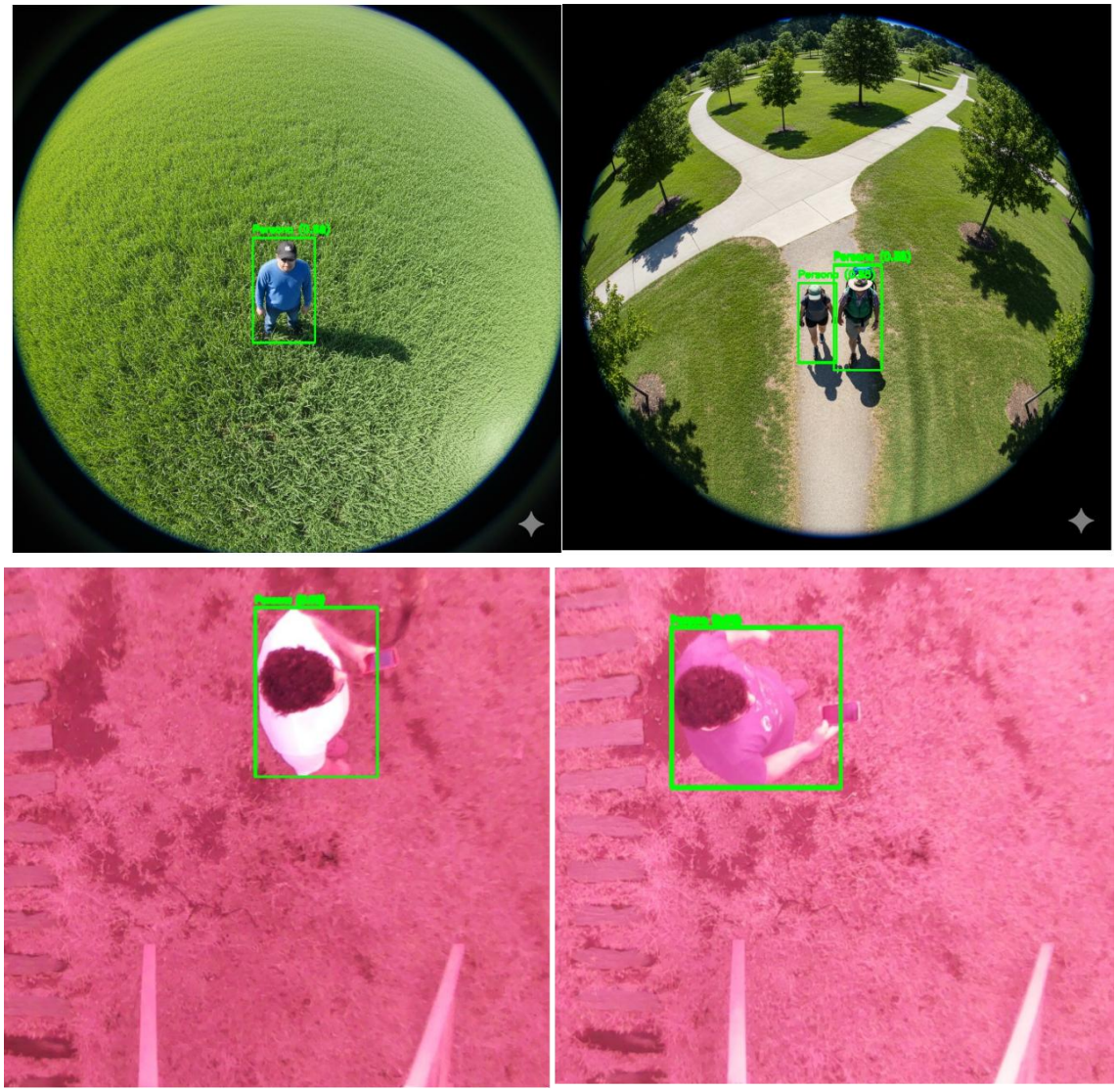
10.3 Resultados de la inteligencia artificial

El sistema de inteligencia artificial fue evaluado utilizando imágenes reales capturadas durante las pruebas de vuelo e imágenes aleatorias de internet.
El modelo procesó las imágenes y generó detecciones representadas mediante bounding boxes, junto con un valor de confianza (score).

Se analizaron los siguientes aspectos:

- Detecciones correctas de personas y fuego.
- Casos de falsos positivos (detección errónea).
- Casos de falsos negativos (elementos no detectados).

Los resultados mostraron un comportamiento adecuado para un prototipo a escala, observándose que la precisión depende fuertemente de las condiciones de iluminación, el ángulo de captura y la distancia al objetivo.



10.4 Pruebas de vuelo

Las pruebas de vuelo se realizaron con el objetivo de validar la estabilidad y la navegación autónoma del dron.

Se evaluaron los siguientes aspectos:

- Estabilidad en vuelo estacionario (hover).
- Respuesta ante comandos de navegación.

- Ejecución de misiones autónomas con waypoints.

El comportamiento observado fue el siguiente:

- El dron logró mantener una estabilidad aceptable durante el vuelo estacionario.
- La navegación autónoma permitió recorrer los waypoints definidos.
- En cada waypoint, el dron se detuvo, realizó la captura de imágenes y continuó con la misión.

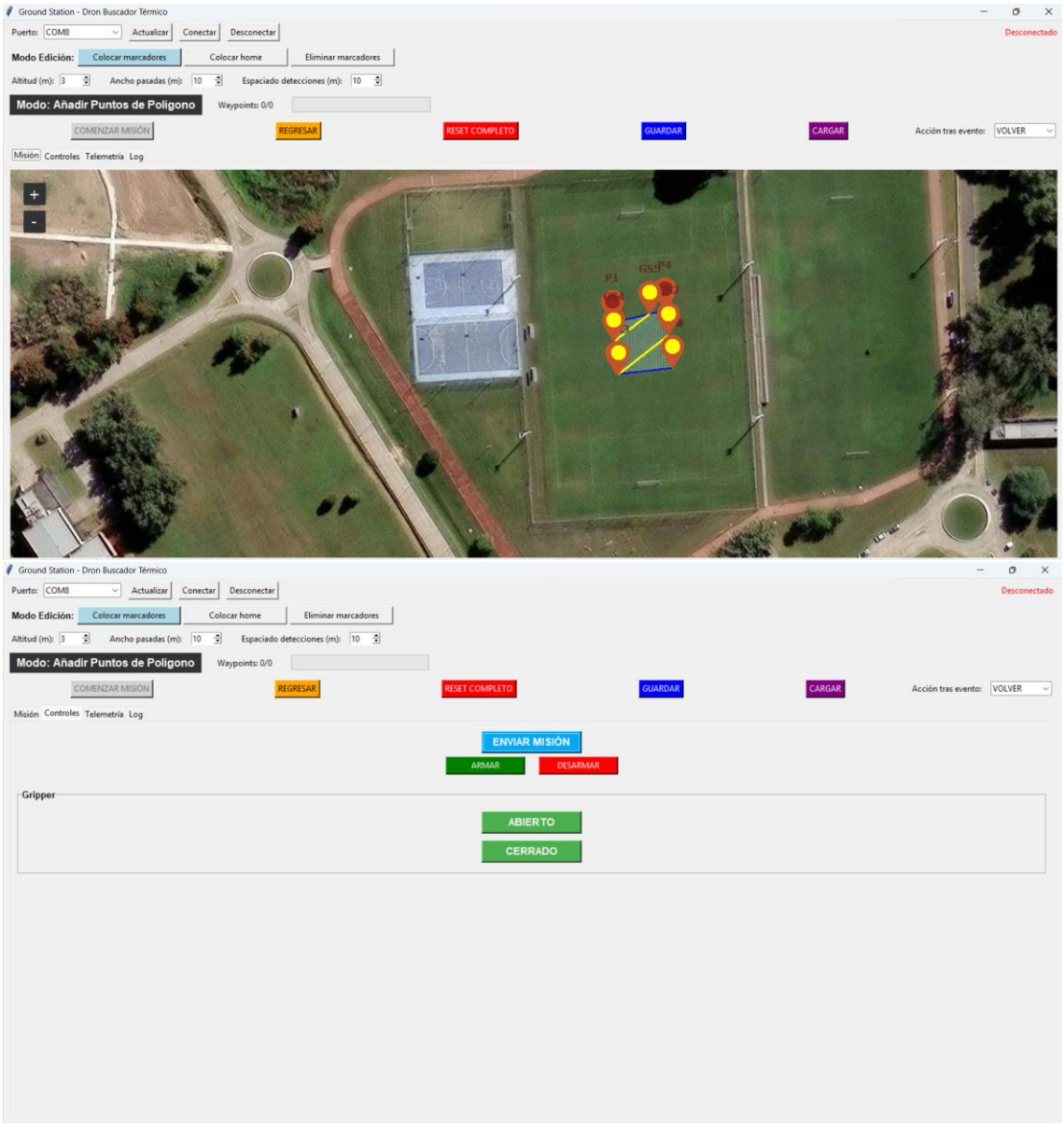
10.5 Capturas de la Ground Station

Durante las pruebas, la Ground Station recibió información enviada desde el dron mediante comunicación LoRa.

Los datos recibidos incluyeron:

- Alertas de detección.
- Coordenadas geográficas del evento.
- Registro de eventos en forma de logs.

Las detecciones fueron visualizadas en el mapa de la Ground Station, permitiendo identificar la ubicación aproximada del evento detectado.


Ground Station - Dron Buscador Térmico
Puerto: COM8
Actualizar
Conectar
Desconectar
Desconectado
Modo Edición:
Colocar marcadores
Colocar home
Eliminar marcadores
Altitud (m): 3
Ancho pasadas (m): 10
Espaciado detecciones (m): 10
Modo: Añadir Puntos de Poligono
Waypoints: 0/0
COMENZAR MISIÓN
REGRESAR
RESET COMPLETO
GUARDAR
CARGAR
Acción tras evento:
VOLVER
Misión
Controles
Telemetría
Log
+
-
P1
P4
Ground Station - Dron Buscador Térmico
Puerto: COM8
Actualizar
Conectar
Desconectar
Desconectado
Modo Edición:
Colocar marcadores
Colocar home
Eliminar marcadores
Altitud (m): 3
Ancho pasadas (m): 10
Espaciado detecciones (m): 10
Modo: Añadir Puntos de Poligono
Waypoints: 0/0
COMENZAR MISIÓN
REGRESAR
RESET COMPLETO
GUARDAR
CARGAR
Acción tras evento:
VOLVER
Misión
Controles
Telemetría
Log
ENVIAR MISIÓN
ARMAR
DESARMAR
Gripper
ABIERTO
CERRADO

10.6 Limitaciones reales encontradas

Durante el desarrollo del proyecto y la etapa de pruebas se identificaron distintas dificultades, las cuales pueden agruparse en dificultades técnicas y dificultades de campo u operativas.

Dificultades técnicas

Una de las principales dificultades técnicas estuvo relacionada con la autonomía limitada de las baterías, lo que redujo el tiempo efectivo disponible para realizar pruebas de vuelo completas. Esta condición obligó a optimizar la planificación de cada ensayo y a priorizar pruebas puntuales por sobre vuelos prolongados.

Otra dificultad relevante fue el establecimiento y validación de la comunicación mediante LoRa entre el dron y la estación base. Fue necesario realizar múltiples pruebas y ajustes para garantizar la correcta transmisión de datos, la recepción de alertas y la estabilidad del enlace, especialmente en condiciones de mayor distancia.

Asimismo, se presentó como desafío la verificación de que el dron respetara correctamente los waypoints calculados antes de iniciar las pruebas de vuelo. Para mitigar riesgos y validar la lógica de navegación, se realizaron pruebas preliminares sin vuelo, desplazando el dron manualmente (caminando con el dron) y observando la respuesta del sistema, la generación de eventos y el comportamiento de la lógica de misión. Este enfoque permitió depurar errores de cálculo y comunicación antes de realizar ensayos aéreos.

Dificultades de campo

Durante las pruebas en exteriores se presentaron diversas dificultades asociadas al entorno. Las condiciones climáticas influyeron de manera directa en la realización de los ensayos, siendo necesario contar con días de clima estable, preferentemente soleados y con bajo nivel de viento.

Otra dificultad importante fue la disponibilidad de espacios amplios, abiertos y seguros para realizar las pruebas. La necesidad de evitar interferencias, obstáculos, edificaciones y la presencia de personas limitó las ubicaciones posibles para los ensayos y condicionó los horarios de prueba.

11. Análisis de Gestión del Proyecto

La gestión del proyecto se estructuró en torno a un cronograma inicial expresado en un diagrama de Gantt y una estimación de horas por tarea. Durante el desarrollo, se registraron las horas reales dedicadas a cada actividad, lo que permitió evaluar desvíos, reasignaciones de trabajo y el grado de complejidad real de cada subsistema.

En total, el proyecto demandó 292 horas reales, superando las 254 horas estimadas, con un desvío global de +38 horas, equivalente a un +15% respecto de la planificación inicial.

11.1 Diagrama de Gantt

El diagrama de Gantt elaborado al inicio del proyecto permitió planificar y organizar temporalmente las actividades necesarias para el desarrollo del dron autónomo, estableciendo una secuencia lógica de tareas y sus respectivas dependencias.

El cronograma contempló las siguientes etapas principales:

- Selección y validación de la idea de proyecto.
- Adquisición de materiales y componentes electrónicos.
- Armado estructural del dron.
- Desarrollo por subsistemas: sistema RGB, sistema infrarrojo, inteligencia artificial, comunicaciones y lógica de vuelo autónomo.
- Integración final de los subsistemas.
- Etapa de pruebas y ajustes.
- Preparación del informe final y presentación del proyecto.

El uso del diagrama de Gantt permitió visualizar las relaciones de dependencia entre los distintos subsistemas, estimar la duración total del proyecto y facilitar el seguimiento del avance a lo largo de la cursada.

El diagrama de Gantt completo se incluye en el Anexo, donde se detallan las actividades, duraciones estimadas y solapamientos entre tareas.

11.2 Tabla de horas estimadas vs horas reales

La siguiente tabla resume el tiempo planificado y el tiempo efectivamente invertido en cada tarea del proyecto.

Nº	Tarea	Horas estimadas (h)	Horas reales (h)	Desvío (h)
1	Selección de proyecto	10	12	+2
2	Selección y compra de materiales	20	24	+4
3	Armado y puesta en funcionamiento del dron	40	35	−5
4	Desarrollo de pilotaje autónomo	25	30	+5
5	Armado y programación IR	25	35	+10
6	Armado y programación del sistema RGB	20	25	+5
7	Integración IR + RGB (Sist. Detección)	15	15	0
8	Comunicación Sist. Detección + Base (GS)	20	25	+5
9	Integración Dron + Sist. Detección	30	20	−10
10	Armado y programación Porta Objetos	10	6	−4
11	Integración Dron + Porta Objetos	14	10	−4
12	Etapa de pruebas	25	55	+30
—	**TOTAL**	**254**	**292**	**+38**

11.3 Gráficos comparativos

Se generaron dos gráficos para visualizar la distribución temporal del proyecto:

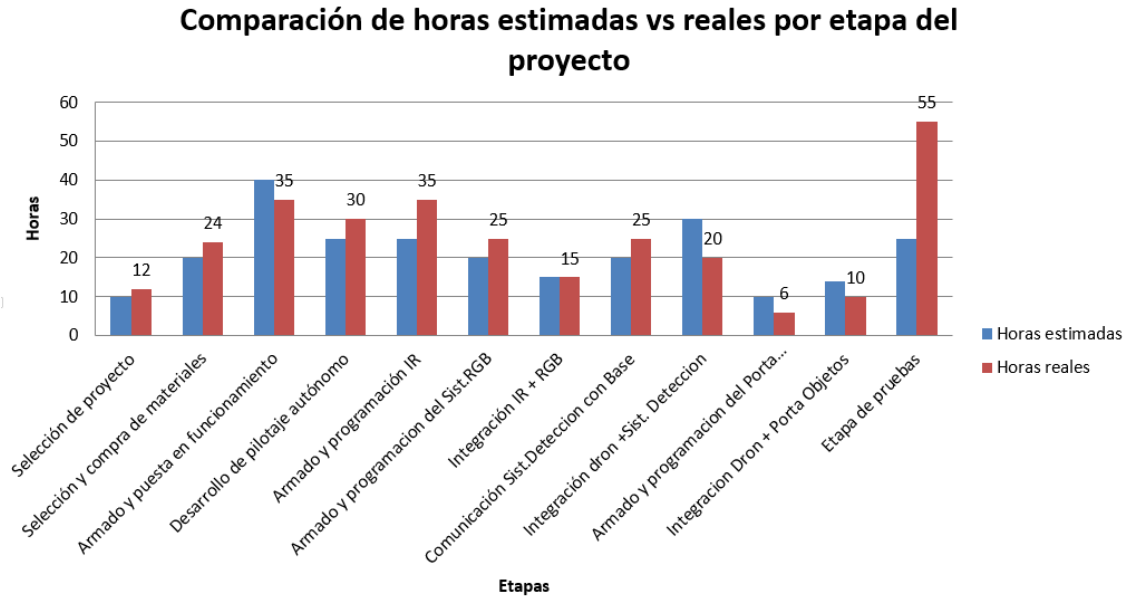
a) Gráfico de barras: horas estimadas vs. reales

Permite observar cuáles tareas tuvieron mayor desvío.

Interpretación esperada:

- Las mayores sobrecargas se encuentran en etapa de pruebas, programación IR, comunicación, pilotaje autónomo y RGB.

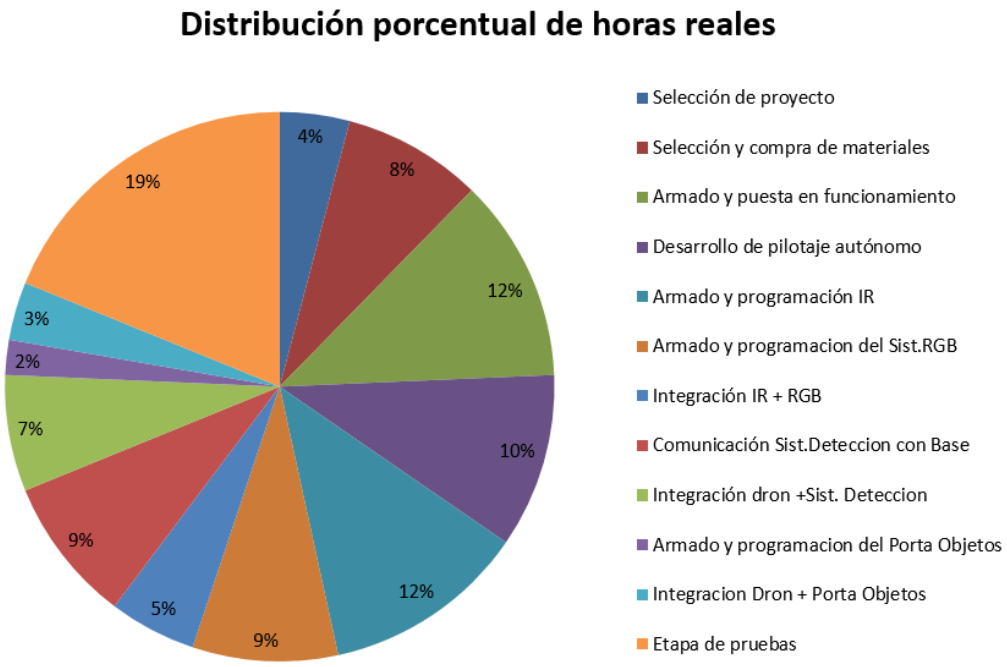
- El armado del dron e integraciones finales consumieron menos tiempo que el previsto.



b) Gráfico de torta: distribución porcentual de horas reales

Refleja los módulos de mayor esfuerzo real:

- Etapa de pruebas: 55 horas (19%)
- Sistema IR: 35 horas (12%)
- Sistema RGB: 25 horas (9%)
- Pilotaje autónomo: 30 horas (10%)
- Comunicaciones + GS: 25 horas (9%)



11.4 Conclusión de gestión

El análisis evidencia que:

- Las tareas orientadas a sensado, IA y pruebas fueron las más demandantes.
- La integración modular facilitó el ensamblado final del sistema.
- La planificación inicial fue adecuada, pero la complejidad del desarrollo de IA y la validación del vuelo autónomo generaron sobrecargas naturales.
- El equipo demostró capacidad de adaptación, reorganización y resolución de problemas para completar el proyecto dentro del cronograma académico.

12. Conclusiones

El proyecto logró desarrollar un prototipo funcional de dron autónomo orientado a la detección temprana de incendios y la localización de personas en zonas de difícil acceso. La integración de visión RGB, sensado térmico, inteligencia artificial embarcada, navegación autónoma y comunicaciones de largo alcance permitió validar experimentalmente el enfoque propuesto y demostrar la viabilidad técnica de una plataforma accesible y modular capaz de ejecutar misiones de monitoreo aéreo.

La arquitectura distribuida —basada en Raspberry Pi, ESP32 y Pixhawk— permitió coordinar captura de datos, análisis local, toma de decisiones y comunicación con la estación base, replicando a escala las funciones presentes en drones profesionales de emergencia. A pesar de las limitaciones propias de un prototipo académico, el sistema logró ejecutar con éxito misiones autónomas, detectando eventos relevantes y comunicándolos de manera confiable.

12.1 Logro del prototipo

El prototipo alcanzó los objetivos planteados:

- Vuelo autónomo por waypoints con despegue, navegación, detenciones y retorno automático.
- Captura de imágenes RGB e infrarrojas en puntos estratégicos de la misión.
- Procesamiento a bordo mediante un modelo YOLOv8 convertido a ONNX.
- Detección de fuego, humo y personas con resultados satisfactorios en pruebas reales.
- Transmisión de eventos vía LoRa hacia la Ground Station con geoposicionamiento.
- Operación coordinada entre Raspberry Pi, ESP32 y Pixhawk bajo un flujo de misión completo.

El prototipo constituye así una prueba de concepto exitosa de un UAV inteligente y autónomo para tareas de apoyo en emergencias.

12.2 Fortalezas del sistema

El desarrollo presenta varias fortalezas técnicas y conceptuales:

Integración completa de subsistemas

Incluye visión, térmica, IA, vuelo autónomo, comunicaciones y GS en un único sistema coherente.

IA embarcada específica para el caso de uso

El modelo YOLOv8 se entrenó para detectar fuego, humo y personas, lo que optimiza su desempeño frente a drones comerciales con modelos más generalistas.

Arquitectura modular y abierta

Permite reemplazar, ampliar o mejorar sensores y módulos sin depender de protocolos propietarios.

Bajo costo en comparación con drones profesionales

El sistema réplica funciones centrales a una fracción del costo de plataformas como DJI Matrice o Autel EVO Dual.

Comunicación robusta en zonas sin infraestructura

El uso de LoRa garantiza alcance aun en áreas montañosas o boscosas donde no hay WiFi ni 4G.

Flujo de misión claro y replicable

Detección → procesamiento → decisión → comunicación → continuación de la misión.

12.3 Debilidades y restricciones

Como todo prototipo académico, el sistema presenta limitaciones que condicionan su desempeño:

Autonomía reducida (4–6 min)

Debido al peso del sistema, tipo de baterías y motores empleados.

Transmisión limitada

El sistema no transmite video en vivo; solo envía eventos y coordenadas por LoRa.

Baja resolución térmica

La MLX90640 funciona bien a corta distancia, pero no reemplaza sensores radiométricos profesionales.

Sensibilidad al viento

El frame utilizado y el peso total afectan la estabilidad en hover prolongado.

Tiempos de procesamiento (1–3 s)

Cada captura requiere detener la misión momentáneamente.

Estas debilidades no comprometen la prueba de concepto, pero sí establecen límites claros frente a aplicaciones operativas reales.

12.4 Mejoras futuras

Existen múltiples caminos para evolucionar el prototipo hacia un sistema más robusto y cercano a un dron operativo:

1. Mejorar la autonomía

- Uso de motores más eficientes (tipo 2216/3508).
- Baterías LiPo de mayor capacidad o Li-Ion.
- Optimización del peso total.
- Optimización del sistema de procesamiento, integrando todo el sistema en un único microcontrolador para reducir consumo de corriente y peso.

2. Incorporar transmisión de video en tiempo real

- Módulo HD digital o WiFi acotado para streaming de baja resolución.
- Integración opcional de un sistema FPV para supervisión.

3. Sustituir el sensor térmico

- Incorporar una cámara térmica radiométrica de mayor resolución.
- Aumentar el alcance térmico y precisión.

4. Mejorar la detección

- Entrenamiento con datasets propios de incendios locales.
- Implementación de fusión RGB–IR para reducir falsos positivos.

5. Robustecer la navegación

- Integración de RTK-GPS para mayor precisión en vuelo autónomo.
- Sensores adicionales (lidar, ópticos) para estabilización en zonas complejas.

6. Incrementar autonomía inteligente

- Implementar mapa de calor de sectores inspeccionados.
- Rescanning automático de zonas sospechosas.
- Ajustes dinámicos de ruta según condiciones ambientales.

13. Anexos

- Códigos
- Planos
- Logs
- Modelos entrenados
- Diagrama