
Integración del dron DJI Air 2S con el ecosistema Robotat empleando el mobile SDK del fabricante DJI

Cristopher René Sagastume González



UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería



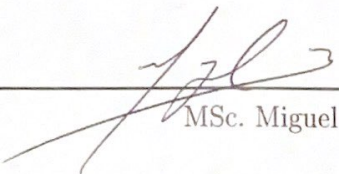
**Integración del dron DJI Air 2S con el ecosistema Robotat
empleando el mobile SDK del fabricante DJI**

Trabajo de graduación presentado por Christopher René Sagastume
González para optar al grado académico de Licenciado en
Ingeniería Mecatrónica

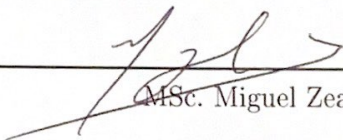
Guatemala,

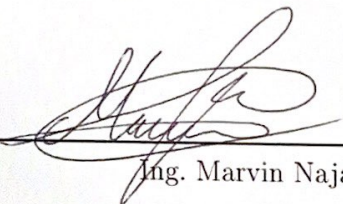
2024

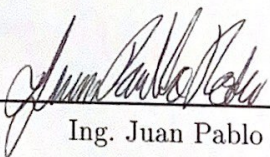
Vo.Bo.:

(f) 
MSc. Miguel Zea

Tribunal Examinador:

(f) 
MSc. Miguel Zea

(f) 
Ing. Marvin Najarro

(f) 
Ing. Juan Pablo Rodas

Fecha de aprobación: Guatemala, 6 de enero de 2024

El presente trabajo de graduación se centra en el desarrollo de una aplicación mediante el uso del *Software Development Kit* (SDK por sus siglas en inglés) de Da Jiang Innovations, en sus siglas DJI, para el dron DJI Air 2S. La finalidad es la adquisición y manipulación de datos provenientes de los sensores del dron, como la IMU, y su integración, por medio de comunicación inalámbrica, con el ecosistema Robotat.

Este proyecto es el fruto de un exhaustivo proceso de investigación, diseño, desarrollo e implementación, donde se enfrentaron numerosos retos tanto en el ámbito técnico como teórico. Cada etapa fue crucial para lograr los objetivos planteados, con especial atención en la usabilidad y fiabilidad de la aplicación, y a la exactitud y validez de los datos obtenidos. Además de la creación de un entorno para el uso del SDK.

La motivación subyacente de este trabajo es aportar al campo de la tecnología de drones y robótica, aspirando a que sirva como referencia e inspiración para futuros trabajos e investigaciones en áreas afines. Este proyecto abre nuevas posibilidades y perspectivas para la integración y la utilización de drones en diferentes ámbitos, potenciando su alcance y funcionalidad.

Quisiera expresar mi agradecimiento primeramente a Dios por darme la oportunidad de estudiar en esta casa de estudios. También agradezco a mis padres y hermana por todo el esfuerzo que han dedicado para lograr este objetivo en mi vida. A pesar de todas las dificultades, siempre han estado para apoyarme y no dejar que dude de mis capacidades de salir adelante. A mi tío, Erick Noguera, pues fue parte de ese esfuerzo que me sacó adelante en mis estudios y siempre me ha dejado saber lo orgulloso que está de mí y alentarme a lograr mis metas. A mis mejores amigos, Marco Trujillo y Alejandro Ortega, por acompañarme durante todo mi proceso de estudio tanto de colegio como de universidad y compartir tanto logros como derrotas, pero siempre con una sonrisa y llevando con nosotros momentos inolvidables. También, quisiera expresar un sincero agradecimiento a mi asesor, MSc. Miguel Zea, cuya orientación y conocimientos fueron indispensables para la realización de este trabajo.

Es mi esperanza que este trabajo sea de utilidad para aquellos interesados en el desarrollo de aplicaciones para drones y en la exploración de nuevas tecnologías. Además de que inspire a otros a contribuir y expandir los horizontes en estos innovadores campos de estudio.

Prefacio	III
Lista de figuras	VI
Resumen	VII
<i>Abstract</i>	VIII
I. Introducción	1
II. Antecedentes	2
A. Usos del DJI Mobile SDK del fabricante DJI para distintas aplicaciones	3
B. Ecosistema Robotat	4
III. Justificación	6
IV. Objetivos	7
A. Objetivo general	7
B. Objetivos específicos	7
V. Alcance	8
VI. Marco teórico	10
A. SDK del fabricante DJI	10
B. Android Studio	11
B.1. Estructura de proyectos en Android Studio	11
B.2. Sistemas de compilación	12
B.3. Terminología de <i>gradle</i>	12
C. Sistema de captura de movimiento OptiTrack	13
D. Drones	14
E. Dron DJI Air 2S	16
F. Protocolo de comunicación TCP/IP	18
G. Formato JSON	19

VII. Desarrollo de una aplicación Android para obtener datos de sensores y cámara del dron DJI Air 2S.	20
A. Pruebas iniciales del dron	20
B. Creación de entorno específico para el desarrollo de aplicaciones con el SDK del fabricante DJI	21
C. Requisitos de la máquina virtual	22
D. Herramienta de desarrollo	22
E. Configuración del IDE	22
F. Configuración del SDK del fabricante DJI	26
F.1. Configuración de permisos para la aplicación	26
F.2. Funcionalidades utilizadas	26
G. Proceso de obtención de datos	26
H. Movimiento autónomo del dron	28
I. Resultados	30
VIII. Integración del dron con Robotat usando OptiTrack y comunicación inalámbrica	35
A. Pruebas iniciales de comunicación	35
B. Integración completa y resultados	36
IX. Conclusiones	40
X. Recomendaciones	41
XI. Anexos	42
A. Repositorio de código	42
B. Permisos solicitados por la aplicación	42
XII. Referencias	44

Figura 1. Cuadricóptero DJI Air 2S	2
Figura 2. Aplicacion Skycatch para drones DJI	3
Figura 3. Esquema de funcionamiento de tecnología Azure FarmBeats	4
Figura 4. Ecosistema Robotat	5
Figura 5. Vista de proyecto Android Studio	11
Figura 6. Sistema de captura de movimiento	13
Figura 7. Cámara <i>Prime</i> ^{x41}	14
Figura 8. Movimiento autónomo	15
Figura 9. Cuadricóptero DJI Air 2S	16
Figura 10. Estructura del formato JSON	19
Figura 11. Vistas de la aplicación DJI.	21
Figura 12. Vistas de la aplicación de ejemplo	23
Figura 13. Vistas de la configuración del SDK	24
Figura 14. Funcionamiento virtual	25
Figura 15. Vistas de la aplicación	27
Figura 16. Flujo de obtención de datos	27
Figura 17. Flujo de movimiento autónomo del dron	29
Figura 18. Altura inicial	30
Figura 19. Movimiento autónomo	31
Figura 20. Movimientos con autonomía (rotación)	32
Figura 21. Movimientos con autonomía (altura)	33
Figura 22. Recepción de datos	35
Figura 23. Métodos de autonomía del dron	36
Figura 24. Implementación con Robotat	37
Figura 25. Autonomía con Robotat (Rotación)	38
Figura 26. Autonomía con Robotat (Altura)	39

En este proyecto, se desarrolló una aplicación en Android, utilizando la versión 4.16.4 del SDK del dron DJI Air 2S, con el objetivo de captar datos de los sensores de posición, orientación, altitud y cámara del dron. El entorno de desarrollo, establecido en una máquina virtual Windows 10 Home, se implementó mediante *Android Studio* versión Dolphin | 2021.3.1 Patch 1.

Se obtuvieron los datos de altura y orientación de los sensores como la Inertial Measurement Unit (IMU, por sus siglas en inglés) y la cámara del dron. Estos datos se enviaron por medio del protocolo de comunicación Protocolo de control de transmisión/Protocolo de internet (TCP/IP, por sus siglas en inglés) al ecosistema Robotat, utilizando como cliente común una computadora con el software Matlab. De igual manera, se lograron rutinas de movimiento autónomas que realizaron movimientos simples como rotaciones en *pitch*, *yaw* y modificaciones de la altura. Esto fue posible gracias al acceso a los valores de posición y orientación del dron utilizando el sistema de captura OptiTrack, que integra al ecosistema Robotat para enviar comandos desde el servidor al dron.

In this project an Android application was developed, using version 4.16.4 of the *DJI Air 2* drone SDK, with the objective of capturing data from the drone's position, orientation, altitude and camera sensors. The development environment, set in a virtual machine running Windows 10 Home, was implemented using *Android Studio* version Dolphin 2021.3.1 Patch 1.

The height and orientation data were obtained from sensors such as the IMU and the drone camera, which were sent via TCP/IP communication protocol in JSON format to the Robotat ecosystem. Similarly, autonomous movement routines were achieved by performing simple movements such as *pitch*, *roll* and *yaw* rotations, by accessing the drone's position and orientation values using the OptiTrack capture system that integrates the Robotat ecosystem to send commands from the server to the drone.

La tecnología de drones ha experimentado un desarrollo y expansión significativos, abriendo diversas posibilidades en diversos sectores. Los drones, equipados con una variedad de sensores y sistemas avanzados, han demostrado su potencial en tareas que van desde la inspección y monitoreo hasta la captura de imágenes aéreas de alta resolución.

Este trabajo de graduación comprende el desarrollo de una aplicación específica en la plataforma Android, utilizando el SDK proporcionado para el dron DJI Air 2S. El objetivo principal fue adquirir y procesar la información recopilada de los sensores, como la IMU y la cámara del dron. Los datos que se obtuvieron fueron posición, orientación y altitud para integrarlos con el ecosistema Robotat mediante comunicación inalámbrica TCP/IP y lograr un movimiento autónomo.

La aplicación desarrollada se implementó en una máquina virtual con un sistema operativo Windows 10 Home, utilizando *Android Studio* (versión Dolphin | 2021.3.1 Patch 1) como entorno de desarrollo integrado (IDE, por sus siglas en inglés) (capítulo 7). Este entorno se configuró específicamente para el uso eficiente del SDK de DJI en la versión 4.16.4, cumpliendo con los requisitos de *hardware* y *software* necesarios para garantizar un desarrollo y ejecución óptimos de la aplicación.

Este proyecto no solo buscó optimizar la adquisición y manipulación de los datos de los sensores del dron, sino que también tuvo la aspiración de sentar las bases para el funcionamiento autónomo del dron. En el capítulo 9, se planteó la autonomía lograda, la cual se basa en movimientos simples modificando la altura, el ángulo de *pitch* para un movimiento hacia enfrente y el ángulo de *yaw* para hacer girar sobre el eje z al dron, mediante una única pulsación de un botón.

A lo largo de este documento, se presentarán detalladamente los procesos de investigación, desarrollo, implementación y pruebas realizados, así como los retos enfrentados y los resultados obtenidos. El trabajo realizado aspira a contribuir al creciente campo de la tecnología de drones y servir como un recurso valioso para futuras investigaciones y desarrollos en la robótica aérea y la integración de sistemas.

Los drones son una tecnología emergente en el campo de la robótica y la aviación que han experimentado un rápido avance en los últimos años. Como ejemplo de estos se puede mencionar el dron *DJI Air 2S* (DJI, 2023), el cual ha destacado como una herramienta versátil y potente que ha encontrado aplicaciones en diversos sectores. En la actualidad, los drones han sido utilizados en una amplia gama de áreas, como la agricultura, inspección de infraestructuras, cartografía y topografía, fotografía y el cine, y en operaciones de búsqueda y rescate, entre otros.



Figura 1: *Cuadricóptero DJI Air 2S.*

Nota. Adaptada de DJI (2023).

Los drones han demostrado su capacidad para realizar tareas de forma eficiente y segura, brindando beneficios en términos de ahorro de tiempo, costos y recursos, así como en la obtención de datos precisos y en la mejora de la seguridad en diversas aplicaciones. El uso de drones ha revolucionado la forma en que se realizan ciertas actividades, abriendo nuevas oportunidades y posibilidades en diferentes campos. Sin embargo, a pesar de los avances y beneficios que han traído consigo, aún existen retos y limitaciones en su implementación, lo cual justifica la necesidad de investigar y analizar su uso.

A. Usos del DJI Mobile SDK del fabricante DJI para distintas aplicaciones

La empresa Skycatch, fundada en 2013, ha utilizado drones DJI con el SDK de DJI para llevar a cabo proyectos de cartografía y topografía de forma eficiente y precisa demostrando que la integración del SDK de DJI ha permitido a Skycatch optimizar sus operaciones de cartografía y topografía, brindando soluciones innovadoras para la industria geoespacial (Skycatch, 2023).



Figura 2: Aplicación Skycatch para drones DJI.

Nota. Adaptada de Skycatch (2023).

Los drones DJI con SDK han sido utilizados para crear mapas en 3D, modelos digitales de elevación y ortomosaicos de alta resolución para aplicaciones cartográficas y topográficas. La empresa Pix4D ha desarrollado una aplicación llamada *PIX4Dcaptura* que utiliza el SDK de DJI para capturar imágenes aéreas con drones DJI y crear mapas y modelos 3D precisos. Uno de los usos más recientes se puede observar en el blog de la página oficial donde mencionan que se ha agilizado en un 70 % las inspecciones de edificaciones de gran altura utilizando su otra aplicación *PIX4DInspect* haciendo uso de drones para agilizar las inspecciones de fachadas de un edificio de varias plantas en Río de Janeiro, Brasil (Pix4D, 2023).

La tecnología de los drones también ha sido aprovechada por grandes empresas, como en el caso de Microsoft, la cual ha estado trabajando en un proyecto denominado *Azure FarmBeats*, la cual busca avanzar en la tecnología de agricultura de precisión analizando la captura de datos por parte de los sensores multiespectrales de los drones, logrando captar datos como calor, luz, humedad, mapeos de humedad en el suelo de las parcelas, etc. Ofreciendo así la capacidad a los agricultores para poder identificar enfermedades de cultivo, plagas, u otros problemas que puedan afectar a los cultivos (RiyazPishori, gourdsay, vijakum et al., 2023).



Figura 3: Esquema de funcionamiento de tecnología Azure FarmBeats.

Nota. Adaptada de RiyazPishori, gourdsay, vijakum et al. (2023).

B. Ecosistema Robotat

En (Perafán, 2022:16-19) se describe la integración de una red WiFi de comunicación para varios agentes y el sistema de captura de movimiento OptiTrack, denominado Robotat, el cual funcionaba con el protocolo de comunicación MQTT, el cual en combinación fue logrando la interacción de agentes no robóticos con el ecosistema por medio de una antena inteligente. Posteriormente, fue creada una librería en C para el microcontrolador ESP32 el cual fue utilizado para obtener y enviar datos dentro del ecosistema, específicamente con el equipo OptiTrack de igual manera utilizando el protocolo MQTT. También fue realizado un código en Python el cual correría en la computadora designada para el ecosistema, donde dicho programa tomaba los valores de las poses de los agentes provistos por el OptiTrack y los publicaba en un broker y el microcontrolador.

El ecosistema Robotat consiste en una plataforma de acero y plycem recubierta con melamina, capaz de soportar cargas puntales de hasta dos toneladas, un sistema de captura de movimiento de la marca OptiTrack, su propia red local WiFi, la cual funciona como servidor de comunicación entre el sistema OptiTrack, el software de Optitrack (Motive) y los robots (Barrera, 2022).



Figura 4: *Ecosistema Robotat (Perafán, 2022:29).*

Nota. Adaptada de RiyazPishori, gourdsay, vijakum et al. (2023).

El propósito de este proyecto es crear una aplicación para la plataforma móvil de Android. Esta aplicación permitirá la integración del dron DJI Air 2s en el ecosistema Robotat. Utilizará el sistema de captura de movimiento OptiTrack con el fin de analizar la posibilidad de realizar un sistema de movimiento o seguimiento automatizado. Esto se logrará al poder obtener los datos de vuelo del dron y los datos de movimiento de OptiTrack en tiempo real.

El SDK del fabricante DJI permite acceder a los datos de la cámara y la mayoría de los sensores, tales como el módulo GPS, brújula, velocidad de vuelo, altitud, etc, permitiendo saber su ubicación aproximada. Conociendo estos datos, es posible establecer una conexión entre la aplicación de control del dron y el ecosistema Robotat para realizar una traducción de posiciones que, luego, podrían ser enviadas por algún protocolo de comunicación a la aplicación de control del dron para que este realice un movimiento automatizado. Esto con la finalidad de controlar al dron como si fuese un agente más en el ecosistema Robotat.

A. Objetivo general

Desarrollar una plataforma de control para el dron *DJI Air 2S* utilizando el SDK del fabricante DJI e integrando el ecosistema Robotat con enfoque en aplicaciones de visión por computadora.

B. Objetivos específicos

- Desarrollar una aplicación en la plataforma Android usando el SDK del dron DJI Air 2S para obtener la información de los sensores de posición, orientación, altitud y cámara.
- Desarrollar un algoritmo de control para el funcionamiento automático del dron utilizando la información obtenida de la aplicación.
- Integración del dron con el ecosistema Robotat utilizando el sistema de captura de movimiento OptiTrack y comunicación inalámbrica.

Este trabajo de graduación abarca el desarrollo de una aplicación en la plataforma Android, centrada en el uso del software development kit (SDK, por sus siglas en inglés) del dron DJI Air 2S para lograr rutinas de autonomía básica del dron e integración con el ecosistema Robotat. El alcance del proyecto incluye la configuración de un entorno de desarrollo en una máquina virtual de Windows 10 utilizando *Android Studio* versión Dolphin | 2021.3.1 Patch 1. También incluye la implementación de funcionalidades para la adquisición y procesamiento de datos de los sensores del dron.

La creación de la aplicación involucra el diseño y desarrollo de interfaces y movimientos básicos. Estos incluyen elevar y descender el dron a una altura específica, joysticks para el movimiento manual del dron desde la aplicación y el despliegue de información obtenida de los sensores, específicamente la IMU. Para esto se utilizó la versión 4.16.4 del SDK del fabricante DJI para dispositivos Android, utilizando un celular con versión 11 de Android y 4 GB de RAM. Este proceso se vio sumamente limitado por la documentación. A pesar de que el SDK fue actualizado, la documentación no lo fue. Esto presentó un reto al intentar utilizar métodos obsoletos o modificados, haciendo difícil la integración sin saber por completo si funcionaría. Además, fue necesario contar con un dispositivo físico, ya que el SDK tiene validaciones de seguridad que obligan a tener un celular conectado al mando para poder acceder a sus funcionalidades. Debido a esto, fue imposible utilizar el simulador de celular integrado en *Android Studio*.

Para la realización de la aplicación se optó por trabajar en una plataforma de desarrollo móvil debido a que el único SDK disponible compatible con el dron DJI Air 2S es el mobile SDK, que acepta tanto Android como iOS. Entre ambas plataformas se decidió trabajar en Android debido a la familiaridad con el lenguaje de programación (Java) y el ambiente de desarrollo (*Android Studio*). La finalidad de este trabajo no consiste en dominar un nuevo lenguaje de programación o familiarizarse con un nuevo ambiente de desarrollo móvil, sino en implementar el dron con el ecosistema Robotat para controlarlo como un agente más.

El proyecto se enfoca en la integración de los datos adquiridos con la aplicación, una computadora y el ecosistema Robotat, empleando técnicas de comunicación inalámbrica (TCP). Esto se limita al envío de comandos para el movimiento autónomo del dron. La comunicación establecida mantiene la aplicación como servidor y una computadora como cliente, tanto del servidor de la aplicación como del ecosistema, haciendo una comunicación unilateral de la computadora a la aplicación.

Las funcionalidades tanto de autonomía como de la aplicación en general fueron realizadas específicamente para el dron DJI Air 2S debido a la versión del SDK lanzada para este dron y su disponibilidad en el Departamento de Ingeniería Electrónica, Mecatrónica y Biomédica.

La automatización de movimiento del dron se limitó a un algoritmo de movimientos lineales básicos (movimientos en un solo eje a la vez). Esta parte del proyecto se vio limitada por el espacio donde se trabajó, debido al cuidado necesario para no dañar el equipo o a las personas alrededor. Además, ciertos bloqueos del controlador de vuelo del dron no permiten actividades como "misiones", impidiendo configurar un valor fijo para movimientos en los ejes y obligando a moverse paso a paso. El desarrollo completo de funcionalidades autónomas complejas se contempla como una línea futura de investigación y desarrollo.

A. SDK del fabricante DJI

El SDK del fabricante DJI es una herramienta de desarrollo que permite a los desarrolladores crear nuevas aplicaciones y soluciones para drones DJI. Desde su lanzamiento, el SDK ha evolucionado para incluir diferentes componentes como el Mobile SDK, el UX SDK y el Payload SDK, entre otros. Estos componentes ofrecen una variedad de funciones y herramientas que permiten a los desarrolladores crear aplicaciones para diferentes plataformas como iOS, Android y Windows.

El SDK de DJI ofrece a los desarrolladores una amplia gama de funcionalidades y capacidades. Entre estas se incluyen el control de drones, transmisión de video en vivo, seguimiento de objetos, detección de obstáculos y la posibilidad de manejar componentes específicos como el control remoto, la batería y el enlace inalámbrico.

Además, este SDK permite acceder a datos en tiempo real de los sensores del dron a una velocidad de hasta 10 Hz, lo que incluye la posición del GPS, brújula, barómetro, velocidad de vuelo y altitud. Esta capacidad de obtener información precisa y en tiempo real es esencial para aplicaciones como navegación autónoma, toma de decisiones basada en datos de sensores y monitoreo de condiciones ambientales durante el vuelo.

DJI proporciona amplia documentación y ejemplos de código para ayudar a los desarrolladores a aprovechar al máximo estas funciones, fomentando la creación de aplicaciones innovadoras en el campo de los drones (DJIdeveloper, 0)(developer, 0).

B. Android Studio

Es el Entorno de Desarrollo Integrado (IDE, por sus siglas en ingles) oficial para desarrollar apps para Android. Basado en *IntelliJ IDEA*, sin embargo, este ofrece muchas más herramientas y utilidades que IntelliJ IDEA, por ejemplo:

- Un sistema de compilación flexible basado en *gradle*.
- Un emulador rápido y cargado de funciones.
- Un entorno unificado donde se puede desarrollar para todos los dispositivos Android.
- Ediciones en vivo para actualizar elementos componibles en emuladores y dispositivos físicos, en tiempo real (Google, 0).

B.1. Estructura de proyectos en Android Studio

Cada aplicación creada en Android Studio se divide en carpetas de proyectos, cada proyecto tiene sus propios módulos con sus funciones y archivos de recursos como se muestra en la figura 5.

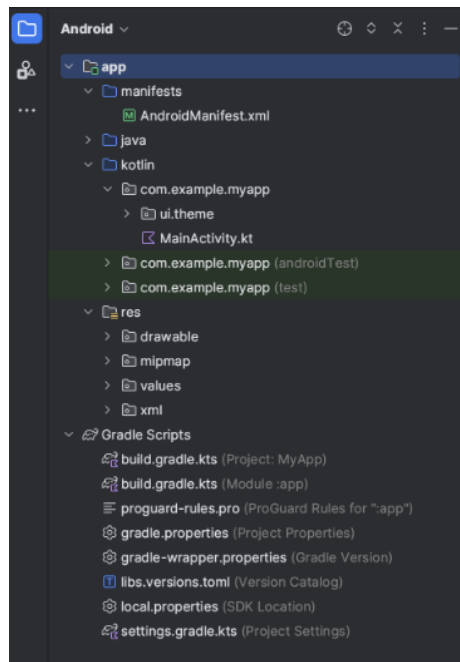


Figura 5: Vista de proyecto Android Studio.

Nota. Adaptada de Google (0).

La carpeta *manifests* contiene la información esencial para el desarrollo de la aplicación y el uso de las herramientas de creación de Android. En las carpetas *java* y *kotlin* se encuentran los archivos que contienen el código fuente de la aplicación, tales como su funcionamiento, cada una de las vistas, etc. En *res* se encuentran todos los recursos de la aplicación como imágenes, vistas de User Interface (UI, por sus siglas en inglés), mapas de bits, etc (Google, 0).

B.2. Sistemas de compilación

Android Studio utiliza *gradle* como sistema de compilación. Este sistema es una herramienta de código abierto. A gran escala, *gradle* realiza ciertas asunciones acerca de lo que se está tratando de compilar o bien qué se desea compilar.

El diseño del compilador *gradle* se basa en distintos fundamentos tales como alto rendimiento, fundación JVM, convenciones, extensibilidad, soporte en IDE y conocimiento. El compilador *gradle* se ejecuta en JVM (máquina virtual de Java), haciendo ventajosa su utilización para los usuarios con conocimientos de Java, ya que la estructura de construcción permite utilizar las API estándar de Java, facilitando la utilización de *gradle* en diferentes plataformas.

Al mencionar convenciones como uno de los fundamentos, se refiere a que *gradle* facilita la creación de tipos de proyectos comunes mediante ensamblajes. Los complementos establecen estándares para mantener los *scripts* de compilación al mínimo. Estos acuerdos no limitan a que pueda realizar ciertas acciones tales como editar configuraciones, agregar funciones y hacer otros cambios en sus dispositivos.

Otro fundamento esencial que proporciona *gradle* es el soporte en IDE, ya que muchos entornos de desarrollo, incluyendo *Android Studio*, brindan interacciones con el compilador *gradle* dándole la oportunidad de generar soluciones ante errores (Gradle, 2018).

B.3. Terminología de *gradle*

Gradle utiliza una terminología simple para funcionar, entre estas se encuentran las siguientes: proyectos, tareas y complementos. Los proyectos son construidos por *gradle* y contienen un *script* de compilación que es un archivo en la raíz del directorio del proyecto, generalmente llamado *build.gradle* o *build.gradle.kts*. En este, están definidas las tareas, dependencias y complementos o *plugins* que se usarán en el proyecto.

Las tareas contienen la lógica que debe correr el compilador, tales como compilar código, ejecutar pruebas o usar *software*. En la mayoría de los casos, utilizará tareas existentes. *Gradle* proporciona tareas que implementan muchas necesidades comunes del sistema de compilación, como la tarea Java integrada *Test* que puede ejecutar pruebas.

Los complementos proporcionan una forma rápida y eficaz de reutilizar código, conceptos y configuraciones en distintos proyectos, lo que ayuda a reducir la repetición en la estructura del documento. La planificación adecuada de cómo construir con complementos puede mejorar en gran medida la facilidad de uso y el rendimiento (Gradle, 2018).

C. Sistema de captura de movimiento OptiTrack

La captura de movimiento es un proceso para registrar el movimiento, orientación y posición de personas u objetos, y digitalizarlos para posteriormente ser transferidos a un *software*, teniendo como fin elaborar modelos 3D donde se refleje la posición o movimientos. Estos sistemas son ampliamente utilizados para propósitos como terapia deportiva, agricultura y atención médica, hasta películas y juegos. La captura de movimiento es una disciplina que esta en constante avance y se presta para realizar distintos experimentos (MoSys, 2020).



Figura 6: Sistema de captura de movimiento (Pennington, 2018).

Nota. Adaptada de Pennington (2018).

El sistema de captura que se encuentra en el ecosistema **Robotat** consiste en un juego de 6 cámaras capaces de captar el movimiento y orientación de los robots que tiene el Departamento de Ingeniería Electrónica, Mecatrónica y Biomédica de la Universidad del Valle de Guatemala. Para el correcto funcionamiento de estas cámaras hay que tomar diferentes puntos en cuenta como:

1. La colocación de las cámaras.
2. Cantidad de luz en el ambiente.
3. Calibración de las cámaras (Perafán, 2022).



Figura 7: Cámara Prime^x41 (OptiTrack, 0)

Nota. Adaptada de OptiTrack (0).

Las cámaras Prime^x41 de OptiTrack tienen una resolución de 4.1 megapíxeles, precisión 3D de ± 0.10 mm, y una tasa de cuadros nativa de 180 fotogramas por segundo (FPS, por sus siglas en inglés) que puede llegar a más de 250 FPS. Además, tienen un alcance impresionante de 100 pies para marcadores pasivos y más de 150 pies con la tecnología *Signature Pulse Active*. También, cuentan con procesamiento de imagen a bordo, lentes de baja distorsión diseñadas internamente, y diversas funcionalidades que facilitan su configuración y sincronización con otros dispositivos (OptiTrack, 0).

Dichas cámaras ofrecen un diseño intuitivo, facilitando su instalación y operación y pueden rastrear marcadores pasivos y activos con una alta precisión y proporcionan una amplia área de seguimiento volumétrico.

D. Drones

Los drones, también conocidos como vehículos no tripulados, son una tecnología emergente con gran variedad en su diseño. Dependiendo del número de hélices, pueden ser denominados cuadricópteros, hexacópteros u octacópteros si poseen 4, 6 u 8 hélices, respectivamente. Su funcionamiento es similar al de un avión o helicóptero, puesto que el giro de sus hélices, impulsadas por los motores, posibilita el vuelo. El control de estos drones se realiza a través de un mando a distancia, mediante el cual se pueden manejar las direcciones de movimiento y rotación utilizando palancas específicas (Expertos en Ciencia y Tecnología, 2018). Las aplicaciones de los drones pueden ser varias, tales como:

- Búsqueda y rescate de personas.
- Monitoreo de cultivos.
- Geología y planimetría.
- Seguridad y vigilancia (IAEROCOL, 2021).

Los drones utilizan tres ángulos de navegación principales: *roll* figura 8c, *pitch* figura 8b y *yaw* figura 8a. El *roll* permite rotar el dron sobre su eje longitudinal, el *pitch* sobre su eje transversal, y el ángulo de *yaw* sobre su eje vertical, permitiendo así controlar la orientación y dirección del dron durante el vuelo (Cienfuegos, 2023). Además, los drones incorporan sistemas de control y estabilización que utilizan giroscopios y acelerómetros para mantener el equilibrio durante el vuelo. Estos sistemas ajustan automáticamente la velocidad de las hélices para mantener la estabilidad, incluso en condiciones de viento (Cienfuegos, 2023).

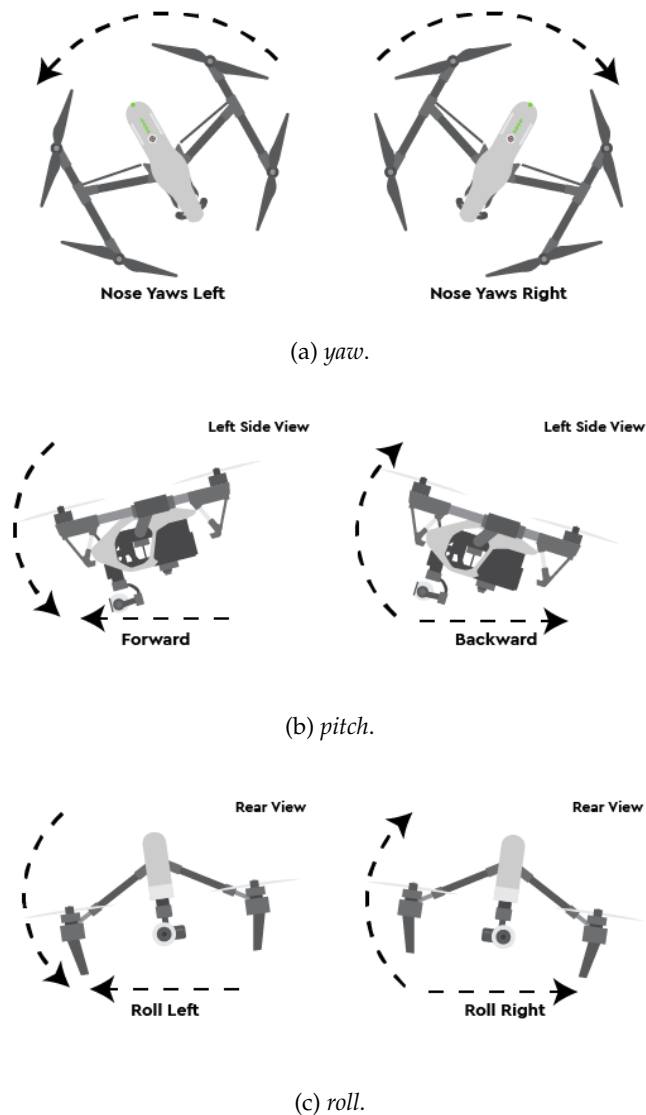


Figura 8: *Movimiento autónomo (Gajeski, 2017).*

Nota. Adaptada de Gajeski (2017).

La comunicación entre el dron y el mando se realiza generalmente a través de señales de radiofrecuencia. Los sistemas avanzados pueden incluir comunicación bidireccional que permite recibir datos del dron en tiempo real, como la transmisión de video. También existen sistemas que permiten la comunicación entre drones para operaciones en flota.

E. Dron DJI Air 2S

Este dron es la evolución del dron Mavic Air 2 del fabricante DJI y consiste en un dron de 4 hélices que permite ser controlador por un mando a distancia. La conectividad de este dron funciona con WiFi para la transferencia de video e imagen a un dispositivo móvil y radiofrecuencia con el mando a distancia.



Figura 9: Cuadricóptero DJI Air 2S (DJI, 2023).

Nota. Adaptada de DJI (2023).

Las características de este dron son las siguientes:

- **Peso:** 595 g.
- **Dimensiones:**
 - **Plegado:** 180x97x77 mm (largo x ancho x alto).
 - **Desplegado:** 183x253x77 mm (largo x ancho x alto) .
- **Longitud diagonal:** 302 mm.
- **Velocidad máxima de ascenso:** 6 m/s.
- **Velocidad máxima de descenso:** 6 m/s.
- **Tiempo máximo de vuelo (sin viento):** 31 minutos.
- **Distancia máxima de vuelo (sin viento):** 18.5 km.
- **Resistencia máxima a la velocidad del viento:** 10.7 m/s.
- **Velocidad máxima de vuelo modo S:** 19 m/s.
- **Velocidad máxima de vuelo modo N:** 15 m/s.
- **Velocidad máxima de vuelo modo C:** 5m/s.
- **Ángulo de inclinación máximo:**

- 35° (modo S).
 - Frente: 30°, Atrás: 20°, Izquierda: 35°, Derecha: 35° (Modo N).
- **Frecuencia de operación:**
 - 2.4 GHz.
 - 5.8 GHz.
- **Velocidad angular máxima:**
 - 250°/s (Modo S).
 - 90°/s (Modo N).
 - 60°/s (Modo C).
- **GNSS:** GPS+GLONASS+GALILEO.
 - **Brújula:** Brújula única.
 - **IMU:** IMU individual.
 - **Almacenamiento interno:** 8 GB.
 - **Cámara:**
 - **Sensor:** CMOS de 1": 20 MP; tamaño de pixel: 2.4 μ m.
 - **Lente:**
 - **Apertura:** f/2.8.
 - **FOV:** 88°.
 - **Equivalente al formato de 35 mm:** 22 mm.
 - **Rango de disparo:** 0.6 m a ∞ .
- **Cardán:**
 - **Estabilización:** 3 ejes(inclinación, balanceo, panorámica).
 - **Rango mecánico:**
 - **Pitch:** Inclinación: -135° a 45°.
 - **Roll:** -45° a 45°.
 - **Yaw:** -100° a 100°.
 - **Rango de vibración angular:** $\pm 0.01^\circ$.
- **Sistema de detección:**
 - Adelante.
 - Hacia atrás.
 - Hacia abajo.
 - Hacia arriba (DJI, 0).

F. Protocolo de comunicación TCP/IP

El Protocolo de Control de Transmisión TCP es un protocolo orientado a la conexión que se establece entre un cliente y un servidor antes de que se puedan enviar datos. TCP es utilizado extensivamente por muchas aplicaciones de internet, incluyendo la World Wide Web (WWW, por sus siglas en inglés), correo electrónico, Protocolo de Transferencia de Archivos (FTP, por sus siglas en inglés), Secure Shell (SSH, por sus siglas en inglés), compartir archivos peer-to-peer, y *streaming* de medios. Este protocolo es optimizado para la entrega precisa más que para la entrega oportuna, y puede incurrir en retrasos relativamente largos mientras espera mensajes fuera de orden o retransmisiones de mensajes perdidos (Eddy, 2022).

TCP es un protocolo orientado a la conexión, lo que significa que establece una conexión entre los puntos de comunicación antes de transmitir los datos. Se diseña para enviar paquetes a través de internet y asegurar la entrega exitosa de datos y mensajes sobre las redes. Descompone la información en paquetes antes de enviarlos al destino y utiliza números de secuencia y mensajes de recibo para proporcionar información de entrega al nodo emisor sobre los paquetes transmitidos a un nodo destinatario. TCP es parte de la suite de protocolos de internet, comúnmente referida como TCP/IP, donde IP se refiere al Protocolo de Internet (Eddy, 2022).

G. Formato JSON

JavaScript Object Notation JSON por sus siglas en inglés, es un formato ligero para el intercambio de datos. Es fácil de leer y escribir para las personas, y fácil de analizar y generar para las máquinas. Los datos que se pueden enviar o recibir en formato JSON incluyen números, cadenas, objetos (una colección no ordenada de pares clave-valor), arreglos, booleanos y valores nulos. En el contexto de TCP, JSON es un tipo de datos que se puede transmitir como un flujo de caracteres entre diferentes sistemas que se comunican a través de una red utilizando TCP. La transmisión de datos en formato JSON es común en aplicaciones y servicios web, especialmente cuando se utilizan APIs (Interfaces de Programación de Aplicaciones) para facilitar la comunicación entre diferentes sistemas o servicios (Network, 2023). La estructura de los formatos JSON se puede observar en la figura 10.

```
{
  "squadName": "Super hero squad",
  "homeTown": "Metro City",
  "formed": 2016,
  "secretBase": "Super tower",
  "active": true,
  "members": [
    {
      "name": "Molecule Man",
      "age": 29,
      "secretIdentity": "Dan Jukes",
      "powers": ["Radiation resistance", "Turning tiny", "Radiation blast"]
    },
    {
      "name": "Madame Uppercut",
      "age": 39,
      "secretIdentity": "Jane Wilson",
      "powers": [
        "Million tonne punch",
        "Damage resistance",
        "Superhuman reflexes"
      ]
    },
    {
      "name": "Eternal Flame",
      "age": 1000000,
      "secretIdentity": "Unknown",
      "powers": [
        "Immortality",
        "Heat Immunity",
        "Inferno",
        "Teleportation",
        "Interdimensional travel"
      ]
    }
  ]
}
```

Figura 10: Estructura del formato JSON (Network, 2023).

Nota. Adaptada de Network (2023).

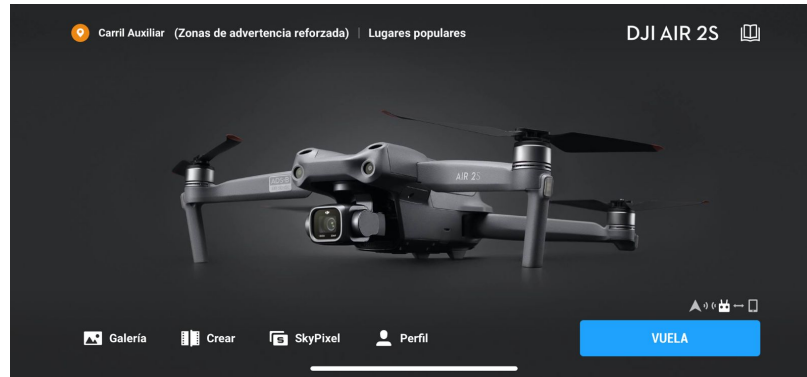
Desarrollo de una aplicación Android para obtener datos de sensores y cámara del dron DJI Air 2S.

En este capítulo se detalla el proceso y metodología para la configuración del entorno de desarrollo de Android Studio, el desarrollo de la aplicación para la plataforma de Android, la adquisición de los datos de los sensores del dron y la realización de rutinas de movimiento autónomo.

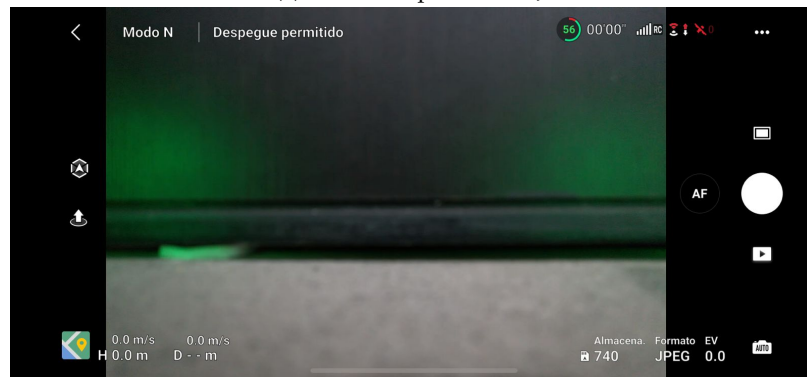
A. Pruebas iniciales del dron

Para familiarizarse con el funcionamiento del dron, se inició el proceso explorando las características y controles proporcionados a través de la aplicación original de DJI, instalada en un dispositivo iPhone 11. La aplicación, disponible en la App Store, sirvió como una herramienta esencial para la operación y gestión del dron, permitiendo una interfaz entre el operador y el sistema de *hardware* del dron como se muestra en la figura 11.

Tras la instalación, se procedió a explorar las diversas funcionalidades ofrecidas por la aplicación, tales como el monitoreo en tiempo real, ajustes de configuración del dron, y acceso a datos telemétricos. Esta fase inicial de familiarización permitió obtener una comprensión sólida del comportamiento y capacidades del dron, además de, las opciones de configuración y modos de operación.



(a) Inicio de aplicación DJI



(b) Menú de aplicación DJI

Figura 11: *Vistas de la aplicación DJI.*

Nota. Elaboración propia.

Posteriormente a la exploración inicial con la aplicación de DJI, se dio inicio al proceso de investigación y prueba del SDK proporcionado por DJI. Se enfrentaron desafíos de configuración inicial, siendo el primer obstáculo la incompatibilidad con la versión de *Android Studio Giraffe | 2022.3.1*. Sin embargo, tras varios intentos, se logró la integración del SDK con la versión *Android Studio Dolphin | 2021.3.1 Patch 1*.

B. Creación de entorno específico para el desarrollo de aplicaciones con el SDK del fabricante DJI

El siguiente objetivo fue establecer un entorno virtual que permitiera trabajar con la versión ya integrada del SDK en Android Studio. Inicialmente, se intentó con una versión de *Ubuntu 23.04*, pero tras enfrentar problemas de integración, se optó por crear un entorno virtual utilizando *Windows 10 Home*, logrando finalmente una integración exitosa.

Para configurar un entorno de desarrollo de aplicaciones utilizando el SDK de DJI. Se utilizaron las herramientas, configuraciones y validaciones necesarias para asegurar un entorno de desarrollo funcional.

C. Requisitos de la máquina virtual

Para la instalación de las herramientas de desarrollo, se utilizó un sistema con las siguientes especificaciones:

- **Sistema operativo:** Windows 10 Home.
- **Memoria RAM:** 10 GB.
- **Espacio en disco:** 60 GB.

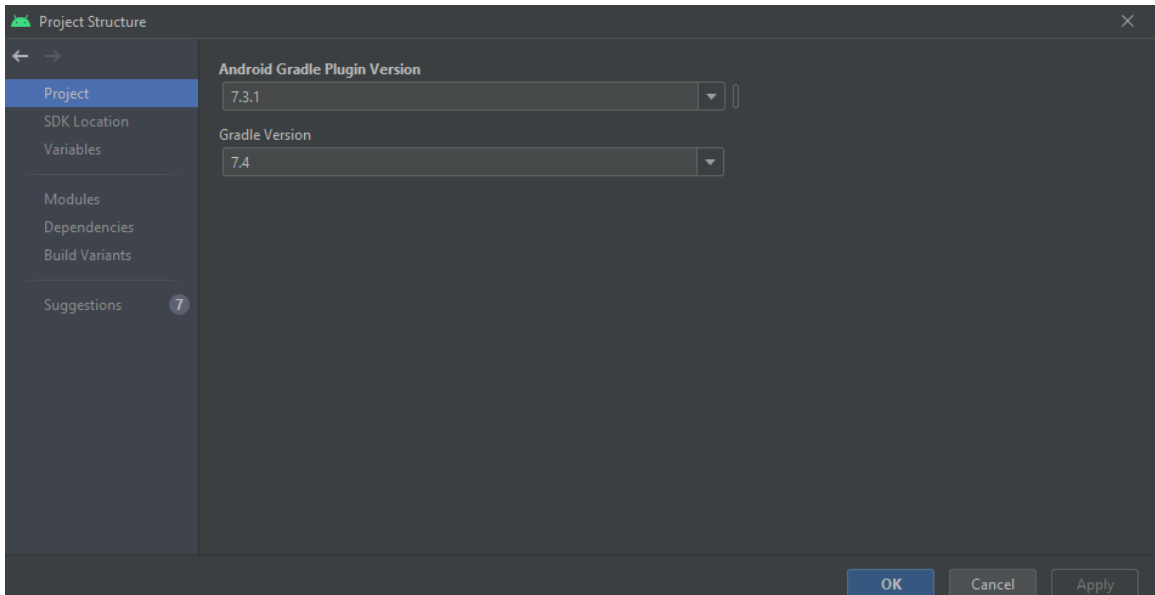
D. Herramienta de desarrollo

Se optó por utilizar *Android Studio* Dolphin | 2021.3.1 Patch 1 como entorno de desarrollo integrado (IDE). Esta versión fue seleccionada debido a su compatibilidad con el SDK de DJI y las características adicionales que ofrece para el desarrollo de aplicaciones Android.

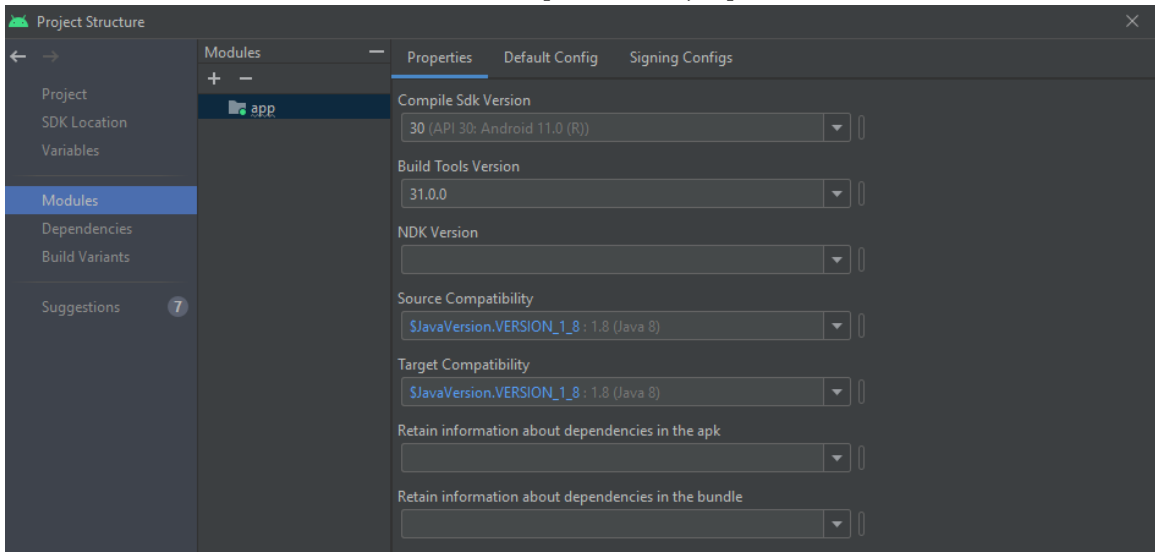
E. Configuración del IDE

Se procedió a configurar el SDK de Android en la versión 30. Además, se configuró la herramienta de compilación en la versión 31.0.0. De igual manera, se colocó como versión mínima de SDK 23 para asegurar el correcto funcionamiento en distintas versiones de Android. Para la configuración del proyecto, se utilizó una versión de build gradle de 7.3.1.

Como se puede observar en las Figuras 12b y 12a se presenta la ventana de configuración del proyecto, a la cual se puede acceder desde el menú de opciones *file*, luego, en *project structure* desde la cual se puede configurar el sistema de compilación y SDK de Android.



(a) Inicio de aplicación de ejemplo



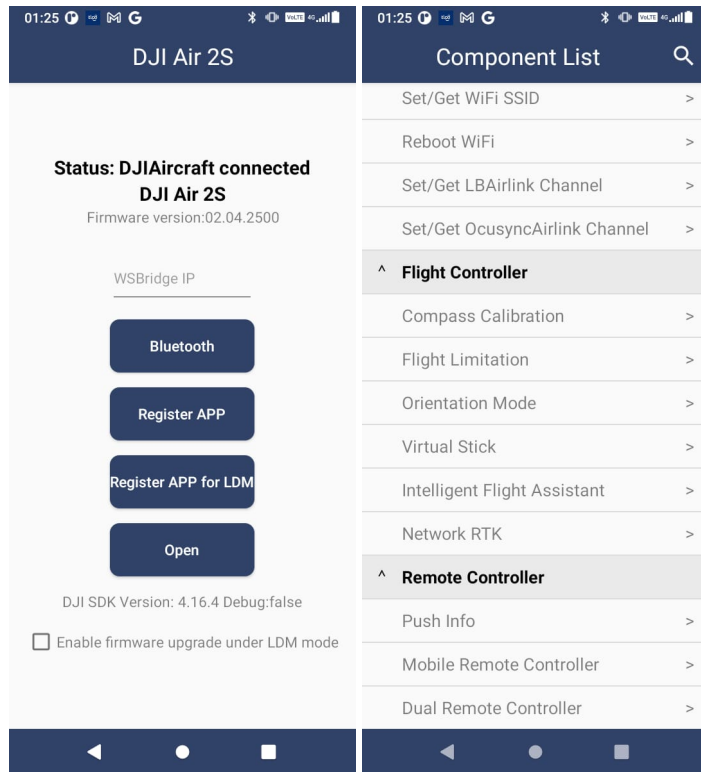
(b) Menú de aplicación de ejemplo

Figura 12: *Vistas de la aplicación de ejemplo*

Nota. Elaboración propia.

El SDK que proporciona el fabricante DJI provee una aplicación para Android que contiene ejemplos de los métodos a los que se pueden acceder. Una vez familiarizado con el funcionamiento general del dron se procedió a instalar dicha aplicación en un dispositivo Android. Ya instalada la aplicación, debido a que no existía documentación sobre su uso, se tomó el reto de investigar cómo funcionaba y la manera para controlar el dron.

Como primer paso, se mostró un menú donde, al momento de conectar el mando al celular y tener encendidos tanto el mando como el dron se muestra el dispositivo que se estará controlando, en este caso, es el dron DJI Air 2S (Figura 13a). Luego se debe activar el *API key* para que se activen las opciones de menú como se puede observar en la Figura 13b.



(a) Inicio de aplicación de ejemplo (b) Menú de aplicación de ejemplo

Figura 13: *Vistas de la configuración del SDK*

Nota. Elaboración propia.

Con el SDK ya integrado, se procedió a explorar la aplicación de ejemplos proporcionada por DJI para Android. Esta aplicación resultó ser una valiosa herramienta de aprendizaje, a pesar de la falta de documentación detallada sobre su uso. La aplicación presentó un menú inicial que, tras conectar el mando al dispositivo móvil y encender tanto el mando como el dron, mostró el dispositivo controlado, en este caso, el dron DJI Air 2S. Era necesario activar la *API key* para habilitar las opciones del menú.

Dentro del menú, se encontró una opción que dirigía a una interfaz de control virtual del dron como se observa en la Figura 14, la cual contenía *virtual sticks* para el movimiento, un botón de elevación, un botón de simulador, entre otros. Al intentar interactuar con estas opciones, se observó que el dron no respondía a los comandos.

Para obtener un mayor entendimiento sobre el funcionamiento de la aplicación, se procedió a analizar los códigos de Java proporcionados en la aplicación de ejemplo. Esta exploración, junto con la revisión de la documentación disponible, permitió adquirir un conocimiento más profundo sobre el flujo de funcionamiento y los métodos utilizados. Aunque la documentación no estaba completamente actualizada con respecto a las versiones recientes del SDK, los foros de desarrolladores de DJI resultaron ser una fuente invaluable de información para descubrir métodos y funcionalidades nuevas que fueron cruciales para el avance del proyecto.

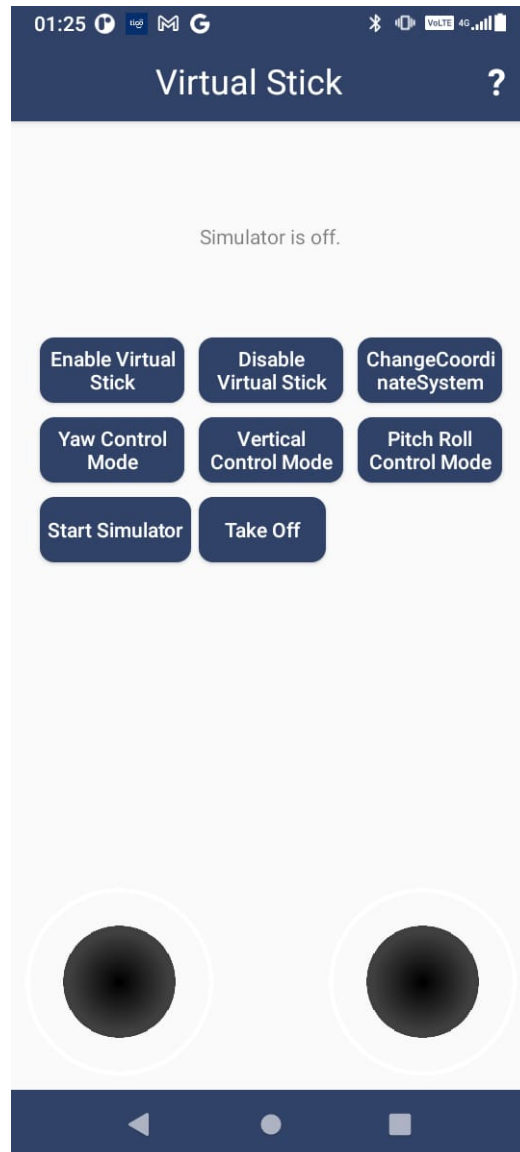


Figura 14: *Funcionamiento virtual*

Nota. Elaboración propia.

F. Configuración del SDK del fabricante DJI

Se utilizó la versión 4.16.4 del SDK del fabricante DJI, para el cual se creó una *API key* con el nombre del paquete de la aplicación (*com.dji.importSDKDemo*) para poder realizar la validación y uso de las herramientas del SDK.

F.1. Configuración de permisos para la aplicación

Se solicitaron diversos permisos del dispositivo Android para poder lograr la comunicación tanto con el control del dron como con un servidor externo para poder lograr la comunicación TCP. El dispositivo Android que se utilizó fue un *SKY devices* con 4 GB de RAM, versión 11 de Android y procesador *Quad Core 2.0 GHz T310*.

F.2. Funcionalidades utilizadas

A pesar que el SDK proporciona una amplia variedad de funcionalidades no todas fueron necesarias para poder lograr el objetivo, por lo que se procedió a utilizar las siguientes funcionalidades:

- Control de vuelo
- Simulador
- *Joystick* virtual
- *UI* dinámica
- Modo de control
- Eventos y *callbacks*
- Verificación de módulos
- Manejo de accesorios

G. Proceso de obtención de datos

Para la obtención de datos de los sensores del dron primero se le muestra al usuario una vista de inicio donde deberá registrar el paquete con los servidores de DJI, esto mediante el botón "Activación" que se puede observar en la Figura 15a, una vez registrado exitosamente el paquete con el *API key* se le permite al usuario ingresar a la vista de menú donde existen diversas funcionalidades, esto mediante el botón Menú en la Figura 15b, a partir de este punto el dron inicia el envío de información de los sensores por medio del control a la aplicación.

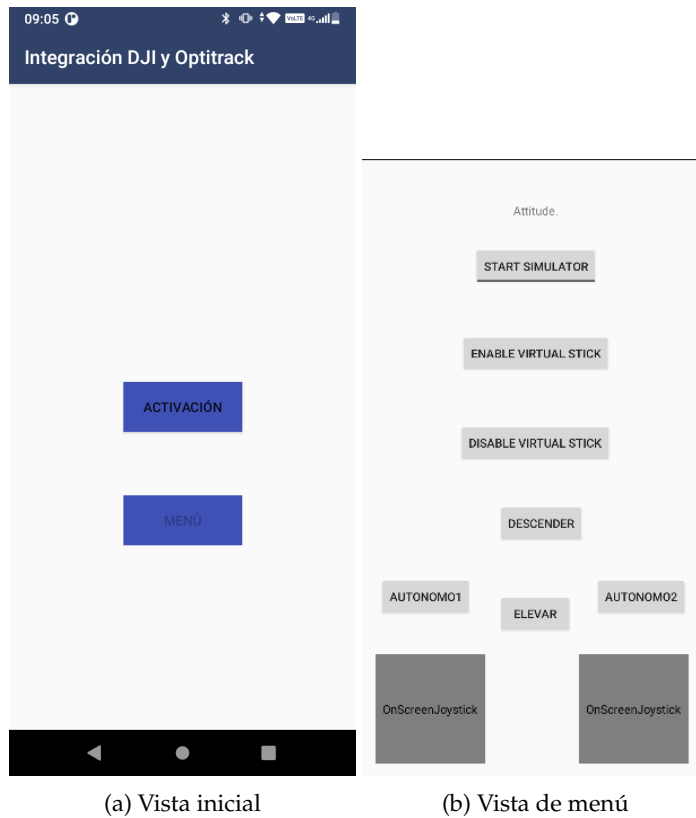


Figura 15: *Vistas de la aplicación*

Nota. Elaboración propia.

```

import dji.sdk.flightcontroller.FlightController;

//Algoritmo de Obtención de Actitud del Dron (pitch, yaw, roll)
private FlightController flightController = null;
flightController = MApplication.getAircraftInstance().getFlightController();
flightState = flightController.getState();
Attitude attitude = flightState.getAttitude();

//Algoritmo de Obtención de Altitud del Dron
altitud = flightState.getAircraftLocation();
float altitud2= altitud.getAltitude();

```

Figura 16: *Flujo de obtención de datos*

Nota. Elaboración propia.

- **Obtención de información del controlador del dron:**
 1. **Importar la librería del SDK:** importar la librería necesaria para interactuar con el dron.
 2. **Inicialización de variables:** declarar una variable *flightController* de tipo *FlightController* e inicializarla como *null*.
 3. **Enlace al controlador:** la variable *flightController* actuará como un enlace al controlador de vuelo del dron.

- **Conexión y estado del dron:**
 1. **Obtener instancia del dron:** usar el método *getAircraftInstance()* de la clase *MApplication* para obtener una instancia del dron conectado.
 2. **Acceso al controlador de vuelo:** utilizar el método *getFlightController()* para acceder al controlador de vuelo del dron y almacenarlo en *flightController*.
 3. **Obtener estado del dron:** usar el método *getState()* del objeto *flightController* para obtener el estado actual del dron.

- **Información detallada:**
 1. **Extraer *attitude* del dron:** utilizar el método *getAttitude()* del objeto *flightState* para extraer las orientaciones angulares del dron, como el *pitch*, *yaw* y *roll*.

- **Obtención de la altitud:**
 1. **Obtener ubicación geográfica:** usar el método *getAircraftLocation()* del objeto *flightState* para obtener la ubicación actual del dron, que incluye latitud, longitud y altitud.
 2. **Extraer altitud:** utilizar el método *getAltitude()* del objeto *altitude* para extraer la altitud del dron en metros.
 3. **Almacenar altitud:** almacenar el valor de la altitud en una variable de tipo float llamada *altitud2*.

H. Movimiento autónomo del dron

Para lograr la autonomía del dron fue necesario la configuración y envío de datos al controlador de vuelo del dron, para lo cual se tuvo que configurar diversos parámetros del controlador de vuelo tales como: *VerticalControlMode* en modo velocidad, *YawControlMode* en modo velocidad angular y *RollPitchControlMode* en modo velocidad, esto para que al momento de que se le enviaran datos al controlador fueran tomados como velocidades y no como ángulos o valores específicos de altura.

Para realizar el movimiento autónomo del dron se implementaron 2 botones como se muestra en la Figura 15b, el botón "Autónomo2" eleva/desciende el dron a 2 metros desde su punto de despegue, posteriormente se le mandó un ángulo de 45 grados para el *yaw* haciendo que gire sobre su eje *z* y luego se colocó un ángulo de *pitch* durante un tiempo aproximado de 100 ms para poder hacerlo volar hacia adelante una distancia aproximada de 1 metro. El botón "Autonomo1" realiza un algoritmo más simple el cual simplemente eleva/desciende el dron a 1 metro desde su punto de despegue.

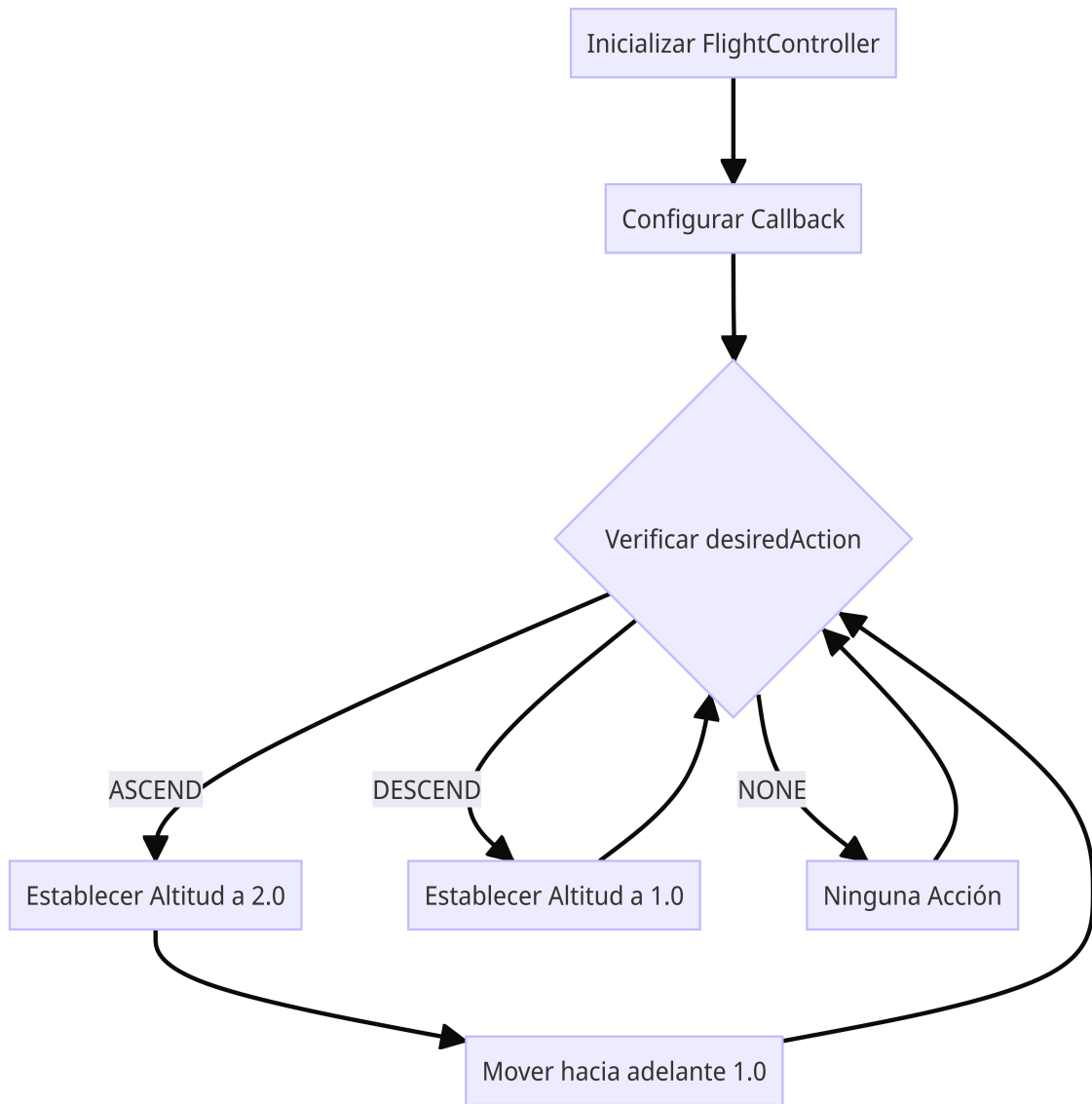


Figura 17: Flujo de movimiento autónomo del dron

Nota. Elaboración propia.

I. Resultados

Cuando el botón de "Autónomo2" es presionado se coloca *desiredAction* en *ASCEND* para poder realizar la rutina que se programó. Si se presiona el botón "Autónomo1" se coloca el *desiredAction* en *DESCEND* para poder realizar la rutina que se le programó.

Cuando se presiona el botón "elevar" el dron se eleva a una posición de 1.5 metros con respecto a su posición de despegue como se muestra en la Figura 18.



Figura 18: *Altura inicial*

Nota. Elaboración propia.

Al momento de que se presionó el botón de "Autónomo2" se eleva a una altura de 2 metros con respecto a la posición de despegue del dron (Figura 19a), posteriormente el dron giró 45 grados con respecto a su eje z (Figura 19b) y luego se mueve un aproximado de 1 metro hacia adelante (Figura 19c) como está programada la rutina del botón.



(a) Altura final

(b) Posición x inicial

(c) Posición x final

Figura 19: *Movimiento autónomo*

Nota. Elaboración propia.

Para poder obtener una visión más clara del movimiento autónomo del dron se decidió colocar un *logger* que guardara los datos de la *IMU* para posteriormente con ayuda de Matlab graficarlas obteniendo así el seguimiento de movimiento del dron como se muestra en la Figura 20.

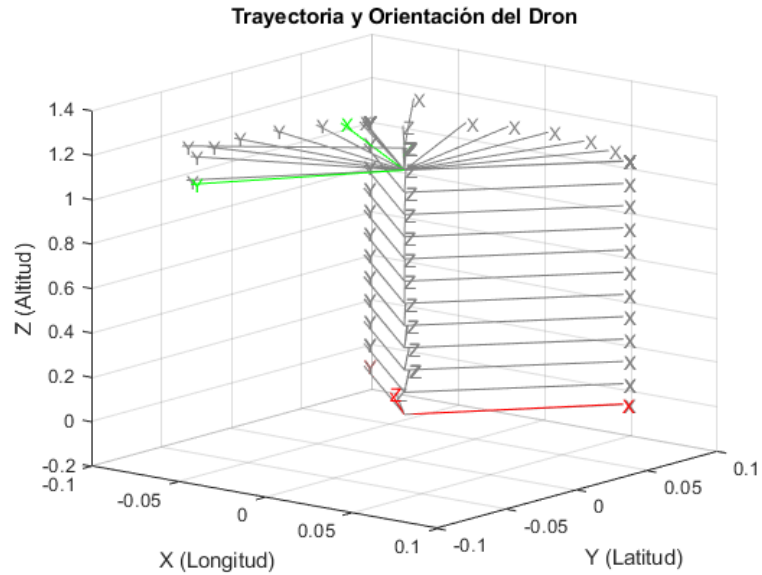


Figura 20: *Movimientos con autonomía (rotación)*

Nota. Elaboración propia.

Como se puede observar en la Figura 20 en rojo esta la posición inicial del dron junto con su altura y orientación y en verde la posición final de igual manera con su respectiva altura y orientación. La elevación final llegó a 1.2m de altura lo cual es lo que sucede al presionar el botón de Elevar, una vez alcanzada esta altura se presionó el botón de Autonomía2 el cual estaba configurado para hacer girar el dron modificando el ángulo yaw el cual estaba configurado en velocidad angular luego de un movimiento aproximado de 90° se detuvo al dron presionando el botón Autonomía1 para detener al dron ya que este botón tenía programada una rutina que colocaba en 0 la velocidad angular de yaw .

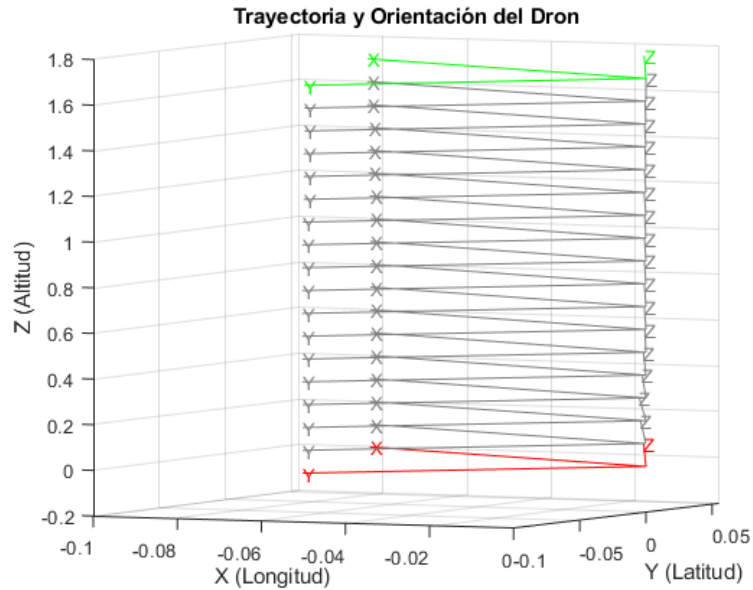


Figura 21: *Movimientos con autonomía (altura)*

Nota. Elaboración propia.

Para la siguiente prueba se realizó un cambio en la rutina de autonomía para esto se colocó un *Throttle* de 0.1 que equivale a una elevación de 0.1m/s para el botón "Autonomía2" y un *Throttle* de 0 para el botón "Autonomía1".

En la figura 21 se puede observar el comportamiento de autonomía al realizar un cambio en la velocidad de elevación, en rojo se puede observar la posición inicial del dron y en verde la posición final, inicialmente el dron se elevó a una altura de 1.1m al presionar el botón de Elevar. Una vez alcanzada esta elevación se presionó el botón de "Autonomía2" para poder elevar el dron de manera autónoma hasta una altura considerable con la intención de no colocar el dron en riesgo alcanzando una altura máxima 1.7m. Una vez alcanzada dicha altura se presionó el botón Autonomía1 para detener al dron.

La obtención de datos de los sensores de posición, orientación y altitud del dron DJI Air 2S se logró exitosamente mediante la aplicación desarrollada en la plataforma Android. Los datos adquiridos, como la altitud y la orientación angular fueron cruciales para entender y controlar el comportamiento del dron en tiempo real. La estructura y la metodología empleada para acceder a estos datos se diseñó de manera que proporcionaran una interfaz intuitiva y efectiva para el usuario, permitiendo una fácil activación y visualización de los datos en la aplicación.

Al comparar las dos aplicaciones, se destaca que la aplicación desarrollada despliega los datos de la *IMU* en tiempo real, lo cual proporciona una mejor experiencia de uso para el usuario. Aunque el SDK carece de documentación detallada, la exploración de los códigos en *Kotlin* y los recursos disponibles en los foros de desarrolladores de DJI permitieron descubrir y utilizar métodos efectivos para acceder a los datos de los sensores del dron.

En lo que respecta a las rutinas autónomas, los resultados mostraron que el dron pudo ejecutar movimientos programados como ascender, girar sobre su eje z y moverse hacia adelante a una distancia predeterminada. La implementación de los botones "Autónomo1" y "Autónomo2" en la aplicación proporcionó una interfaz sencilla para activar estas rutinas. Aunque la ejecución fue exitosa, el proceso de programación y prueba requirió una comprensión profunda del funcionamiento del dron y la aplicación del SDK.

Un desafío notable fue la necesidad de una inicialización adecuada, como la activación de la *API key* y la conexión correcta entre el mando y el dron, para garantizar la comunicación efectiva y el control del dron. Además, se observó la importancia de la secuencia de activación de los controladores de vuelo. También se descubrió que para el correcto funcionamiento era esencial presionar, primero, el botón de despegue *take off* o bien elevar con el control el dron y, posteriormente, activar los *virtual sticks*.

Aunque se pudieron obtener datos de los sensores y realizar movimientos autónomos, sería beneficioso, en futuras investigaciones, explorar y mejorar la interfaz de usuario, la documentación del código y las características de seguridad. También sería relevante investigar métodos para optimizar la precisión y la rapidez de la obtención de datos, así como la respuesta del dron a las rutinas autónomas programadas, para expandir las capacidades y aplicaciones de esta tecnología en el campo de los drones autónomos.

Cuando se verificó qué datos se pueden obtener desde la cámara mediante la documentación proveída por DJI, se encontró que los métodos que ofrece el SDK únicamente permiten el acceso a las imágenes captadas en tiempo real por la cámara, por lo que no puede extraerse información relevante para el control autónomo del dron.

Integración del dron con Robotat usando OptiTrack y comunicación inalámbrica

A. Pruebas iniciales de comunicación

Se optó por utilizar el protocolo TCP para garantizar una comunicación entre el servidor Robotat y la aplicación. Se configuró la aplicación para poder establecer una conexión directa entre la aplicación y el servidor Robotat. Una vez establecida la comunicación se inició el envío de los datos de los sensores obtenidos en la aplicación en formato JSON como se muestra en la Figura 22, los datos recibidos por el servidor fueron el *pitch*, *yaw*, *roll* y la altitud del dron. Esto se realizó de manera preliminar por medio de un servidor de python utilizando la librería *socket* para lograr un ambiente de pruebas, esto con el fin de simular el servidor del ecosistema Robotat.

```
{ 'PITCH': 1.0, 'YAW': 40, 'ROLL': 0.0, 'ALTURA': 2.0 }
{ 'PITCH': 1.1, 'YAW': 40, 'ROLL': 0.0, 'ALTURA': 2.0 }
{ 'PITCH': 1.2, 'YAW': 40, 'ROLL': 0.0, 'ALTURA': 2.0 }
{ 'PITCH': 1.3, 'YAW': 40, 'ROLL': 0.0, 'ALTURA': 2.0 }
{ 'PITCH': 1.4, 'YAW': 40, 'ROLL': 0.0, 'ALTURA': 2.0 }
{ 'PITCH': 1.5, 'YAW': 40, 'ROLL': 0.0, 'ALTURA': 2.0 }
{ 'PITCH': 1.6, 'YAW': 40, 'ROLL': 0.0, 'ALTURA': 2.0 }
{ 'PITCH': 1.7, 'YAW': 40, 'ROLL': 0.0, 'ALTURA': 2.0 }
```

Figura 22: Recepción de datos

Nota. Elaboración propia.

En la implementación, se utilizó una clase *TcpCommunicationTask* que extiende la clase *AsyncTask* de Android para manejar la comunicación TCP en un hilo secundario de comunicación. En el constructor de esta clase se inicializaron cuatro parámetros: *pitch*, *yaw*, *roll* y altura. Estos parámetros se emplean para construir un objeto JSON que contienen los campos "pitch", "yaw", "roll" y "altura".

Dentro del método *doInBackground*, se inicializó la aplicación como servidor para lograr la comunicación TCP y la lectura y escritura por medio de esta. Para leer los datos recibidos desde el servidor, se usó *BufferedReader* e *InputStreamReader*. Si en algún momento se producía una excepción, se capturaba y se imprimía su seguimiento de pila para fines de diagnóstico. Mientras que para enviar datos al servidor se empleó *PrintWriter*.

Una vez establecida la conexión, se creó un objeto JSON utilizando los cuatro parámetros previamente inicializados y se envió al servidor como una cadena de texto. Se esperó una respuesta del servidor, que posteriormente se almacenó en una variable llamada *response*.

El método *onPostExecute* se ejecutó tras la finalización del método *doInBackground()*. Este método se ejecutó en el hilo principal y recibió como entrada la respuesta del servidor obtenida en *doInBackground()*. Se presentó un mensaje emergente con la respuesta del servidor a través del método *showToast()*.

B. Integración completa y resultados

Como resultado final se configuró la aplicación para poder establecerla como un servidor y se realizó un algoritmo en Matlab para poder usarse como cliente entre la aplicación y el ecosistema Robotat. Una vez se estableció la comunicación entre los 3 puntos (aplicación<-computadora<-robotat) se procedió a realizar la prueba de funcionamiento autónomo del dron corriendo las 2 funciones implementadas que fueron movimiento en altura y rotación en *yaw*, esto debido a que por seguridad del equipo, el entorno y personas no se decidió hacer la prueba de movimiento de avance modificando los ángulos de *pitch* y *roll*, sin embargo, los algoritmos si fueron implementados tanto en Matlab como en la aplicación (Figura 23) para futuras pruebas con un entorno más seguro.

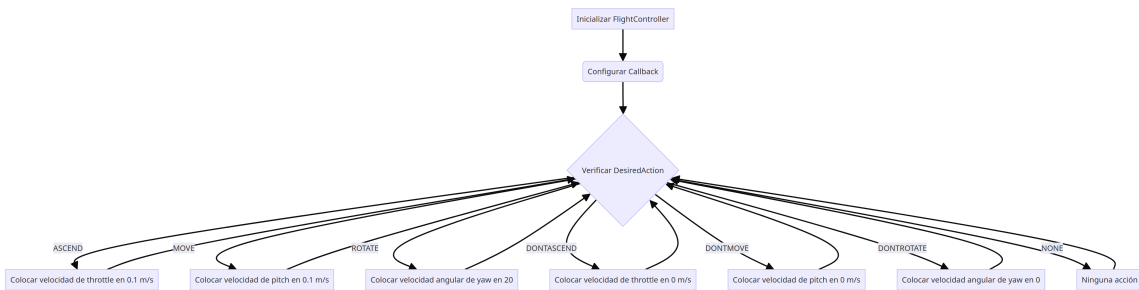


Figura 23: Métodos de autonomía del dron

Nota. Elaboración propia.

Para poder realizar la comunicación entre el cliente de Matlab y el servidor de la aplicación se utilizó la librería TCP/IP que ofrece el *software* de Matlab creando así una conexión hacia la ip del servidor en el puerto 50,000 puesto que tras investigar se encontró que este puerto no se utilizaba en ninguna aplicación o hilo secundario lo cual garantizaría una comunicación estable. En las primeras pruebas realizadas con otros puertos como 3333 o 7777 ocurrieron diversos problemas en la comunicación lo cual retardaba o hacia que fallara el envío de mensajes entre cliente y servidor.

Para poder realizar la autonomía del dron una vez implementado en el ecosistema *Robotat* se realizó de manera diferente a las rutinas programadas antes de la implementación del lado de la aplicación, pues en el método *onPostExecute* se colocó un dato *hardcoded* de *yaw* o *throttle* dependiendo la prueba que se estuviera realizando.

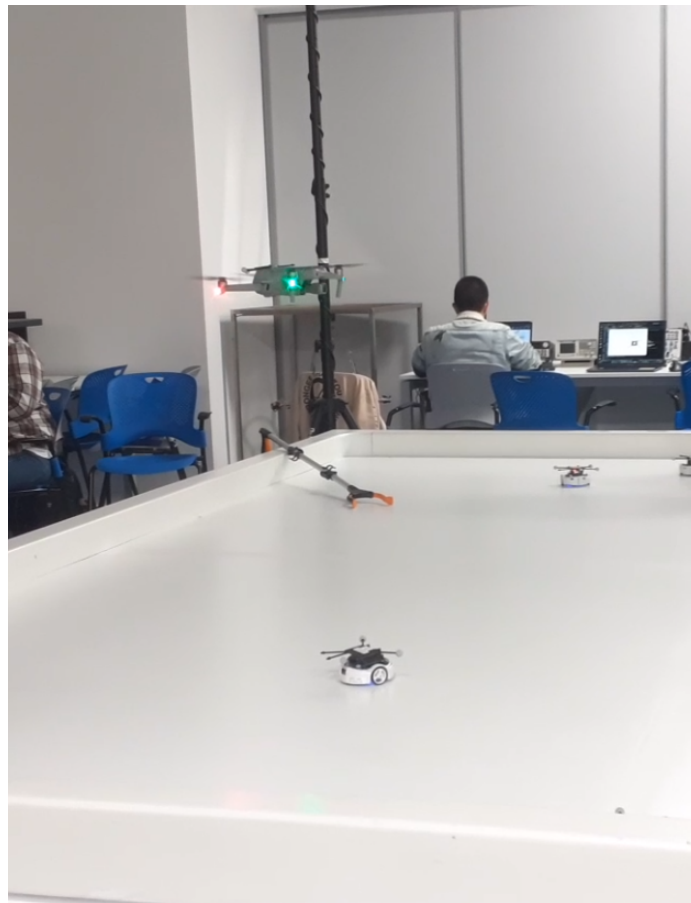


Figura 24: Implementación con *Robotat*

Nota. Elaboración propia.

En la figura 24 se puede observar la implementación del dron con el ecosistema *Robotat*. Durante la prueba se le mandó un comando *elevate* a la aplicación para que el dron despegara y se colocara a una altura de 1.1m. Posteriormente se le mandó un comando de *Rotate* para realizar la misma sesión de autonomía de rotación que se realizó antes de la implementación con el ecosistema.

Durante el proceso de autonomía implementado con el ecosistema Robotat surgieron varios problemas pues inicialmente el dron realizaba los movimientos por pasos a pesar que tenía configurado el control de *yaw* como velocidad angular y el de altura como velocidad, por lo que, se realizó una rutina en matlab para poder realizar el movimiento del dron de manera continua, sin embargo, esto hacía que el movimiento se viera poco natural, por lo que se presumió que el controlador de vuelo del dron tenía alguna validación de seguridad para evitar accidentes de choque con el dron, por lo tanto, la solución encontrada fue que ya no se colocó *hardcoded* el valor de *yaw* y *throttle* sino que se procedió a realizar una configuración a la variable *desiredAction* la misma que utilizaba en la autonomía inicial.

Una vez logrado que el movimiento del dron se viera más natural se procedió a realizar nuevamente las pruebas con el dron y el ecosistema. Las cuales constaron de pruebas en rotación variando la velocidad angular de *yaw* y la velocidad de elevación variando el *throttle*. Los comandos utilizados para la variación en rotación son *elevate* para aumentar la velocidad de elevación y *stop_elevate* para colocar en 0 la velocidad de elevación, también se utilizaron los comandos de *rotate* para modificar la velocidad angular de *yaw* y *stop_rotate* para colocar en 0 dicha velocidad y detener al dron.

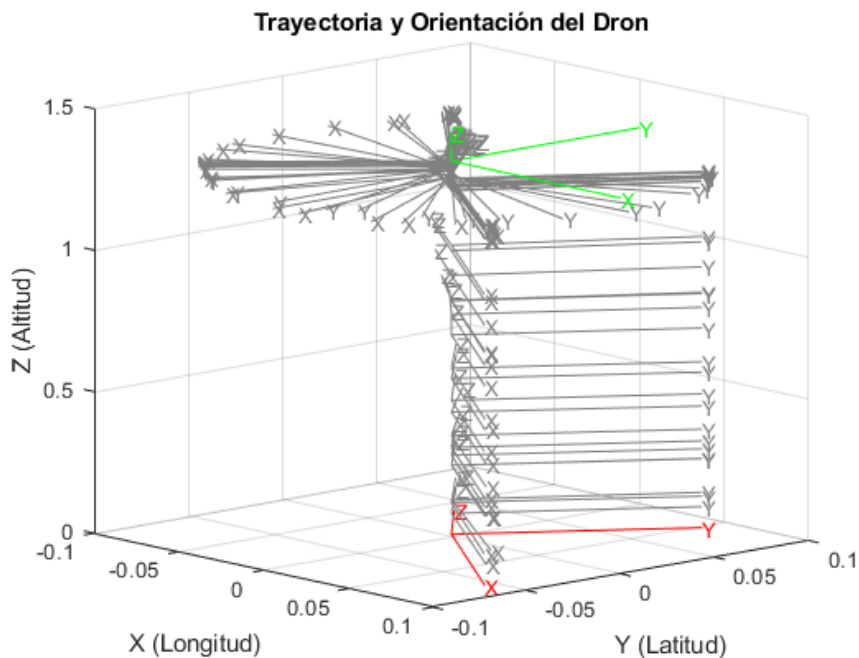


Figura 25: Autonomía con Robotat (Rotación)

Nota. Elaboración propia.

En la figura 26 se puede observar el movimiento captado por el OptiTrack donde en color rojo se puede observar la posición inicial del dron con su respectiva altura y orientación y en verde la posición final de igual manera con su altura y orientación. A diferencia de la figura 20 los ejes tienen un leve desfase esto debido a la posición del marcador colocado en el dron no estaba alineado con los ejes globales del ecosistema, sin embargo, se puede apreciar de igual manera la rotación del mismo. Además que se procedió a realizar

la rutina de elevación del dron enviándole el comando *elevate* donde de igual manera se puede apreciar un crecimiento en la elevación del dron al enviarle el comando *elevate* y posteriormente el comando *stop_elevate* para detenerlo. Para lograr esta altura se elevó el dron y se descendió a una altura de 1m y la altura alcanzada durante esta prueba fue de 1.2m con un crecimiento de 0.1m/s.

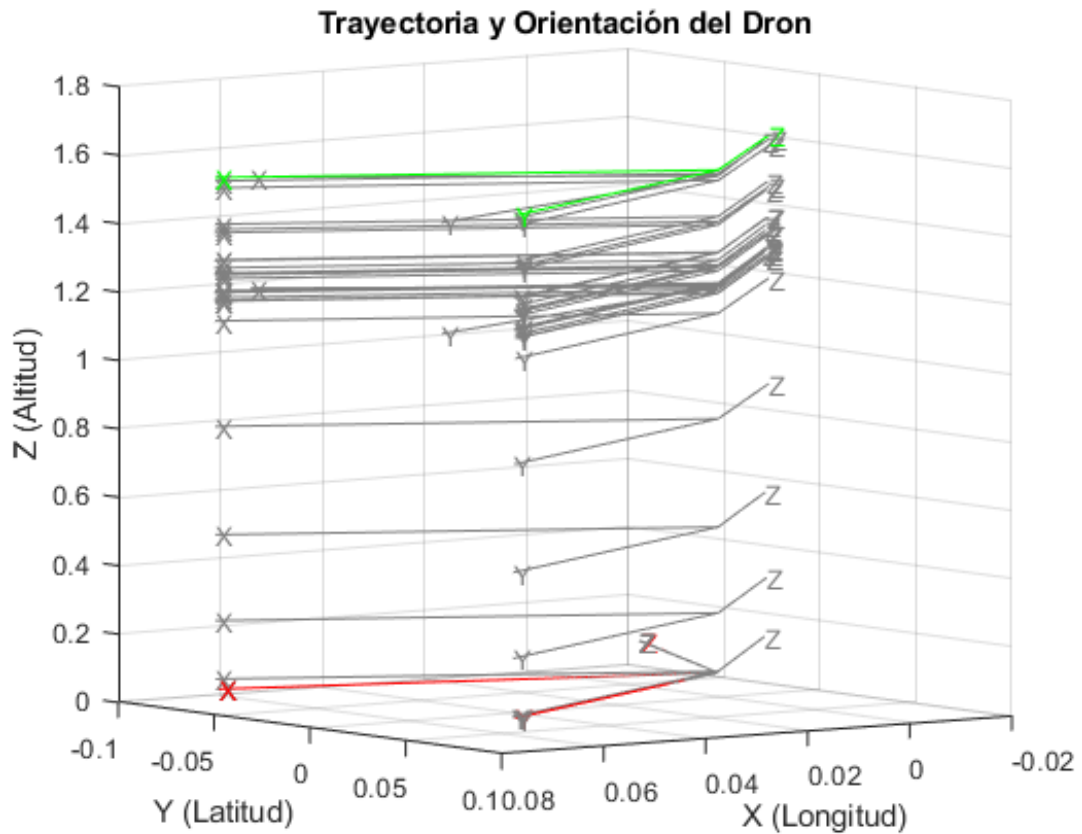


Figura 26: *Autonomía con Robotat (Altura)*

Nota. Elaboración propia.

Como se puede observar en la figura 26 ahora la altura alcanzada fue de 1.5m de igual forma con un crecimiento de 0.1m/s, sin embargo, en esta ocasión fue detenido mediante un script de matlab que constantemente obtenía la posición del marcador colocado en el dron y mientras no se cumpliera la validación de que la altura fuera menor a 1.2m no detuviera el dron este seguía aumentando su altura, esta altura de 1.5m se debió a un *delay* que se encontró entre las comunicaciones entre los servidores y el cliente, que tarda un promedio de 2 segundos en detener al dron una vez enviado el comando.

- Se logró con éxito el desarrollo de una aplicación funcional en Android que permite la adquisición y procesamiento de datos de los sensores del dron DJI Air 2S.
- Se estableció una comunicación entre la aplicación y el ecosistema Robotat utilizando, como cliente común, una computadora corriendo el *software* Matlab y aplicando el protocolo de comunicación TCP/IP, lo que permitió la integración de datos en tiempo real.
- Se desarrolló un algoritmo preliminar para el funcionamiento autónomo del dron, utilizando la información obtenida de la aplicación y la información obtenida del ecosistema Robotat.
- Se logró implementar una rutina de movimiento autónomo del dron mediante la integración con el ecosistema Robotat utilizando comunicación TCP/IP.

Recomendaciones

- Es fundamental continuar el desarrollo y la optimización de algoritmos de control y la comunicación para mejorar la autonomía del dron.
- Utilizar el dron con las funcionalidades de autonomía en ambientes grandes donde no hayan obstáculos para evitar colisiones.
- No utilizar el dron en terrazas de gran altura y donde no haya restricción de vuelo para utilizar libremente la aplicación desarrollada.
- No utilizar el dron con la aplicación desarrollada en ambientes oscuros.
- Tener especial cuidado en los valores de ángulos y velocidades que se colocan en el proceso de autonomía.

En esta sección se presentan diversos elementos complementarios al contenido principal del documento. A continuación, se incluye el enlace al repositorio de código del proyecto y una lista detallada de los permisos solicitados por la aplicación para su correcto funcionamiento.

A. Repositorio de código

A continuación, se presenta el enlace para acceder al repositorio de código donde se encuentra el proyecto de integración del dron DJI Air 2S con el ecosistema Robotat utilizando el SDK de DJI:

<https://github.com/sag18640/Integracion-del-dron-DJI-Air-2S-con-el-ecosistema-Robotat-empleando-el-SDK-de-DJI.git>

B. Permisos solicitados por la aplicación

La aplicación requiere ciertos permisos para funcionar correctamente. A continuación, se detalla la lista de permisos solicitados:

- android.permission.BLUETOOTH
- android.permission.BLUETOOTH_ADMIN
- android.permission.VIBRATE
- android.permission.INTERNET

- android.permission.ACCESS_WIFI_STATE
- android.permission.ACCESS_COARSE_LOCATION
- android.permission.ACCESS_NETWORK_STATE
- android.permission.ACCESS_FINE_LOCATION
- android.permission.CHANGE_WIFI_STATE
- android.permission.RECORD_AUDIO
- android.permission.WRITE_EXTERNAL_STORAGE
- android.permission.READ_EXTERNAL_STORAGE
- android.permission.READ_PHONE_STATE
- android.permission.FOREGROUND_SERVICE
- android.hardware.usb.accessory

-
-
- [1] P. Barrera, *El Robotat, el hábitat donde interactúan los robots en el CIT*, Recuperado el 20 de mayo de 2023, 2022. dirección: <https://noticias.uvg.edu.gt/el-robotat-el-habitat-donde-interactuan-los-robots-en-el-cit/>.
 - [2] C. Cienfuegos, *¿Cómo funciona la estabilización giroscópica en un dron?* Recuperado en octubre de 2023, 2023. dirección: <https://www.dronesbaratosya.com/como-funciona-la-estabilizacion-giroscopica-en-un-drone/>.
 - [3] C. Cienfuegos, *¿Qué son los Roll, Pitch y Yaw?* Recuperado en octubre de 2023, 2023. dirección: <https://www.dronesbaratosya.com/que-son-los-roll-pitch-y-yaw/>.
 - [4] D. developer, *SDK DJI*, <https://developer.dji.com/document/4cd08995-3952-4db6-ab6e-bc3a754da153>, Recuperado el 17 de abril de 2023, 0.
 - [5] DJI, *DJI Air 2S*, <https://www.dji.com/air-2s/>, Recuperado el 13 de abril de 2023, 2023.
 - [6] DJI, *Specs*, Recuperado el 20 de mayo de 2023, 0. dirección: <https://www.dji.com/air-2s/specs>.
 - [7] DJIdeveloper, *DJI Developer*, <https://developer.dji.com/>, Recuperado el 7 de abril de 2023, 0.
 - [8] W. M. Eddy, *Transmission Control Protocol (TCP)*, W. M. Eddy, ed., RFC 9293, ago. de 2022. DOI: 10.17487/RFC9293.
 - [9] E. de Expertos en Ciencia y Tecnología, *¿Qué es un dron y cómo funciona?* Recuperado el 20 de mayo de 2023, 2018. dirección: <https://www.universidadviu.com/int/actualidad/nuestros-expertos/que-es-un-dron-y-como-funciona>.
 - [10] A. Gajeski, *10 TERMS AND ABBREVIATIONS THAT EVERY DRONE ENTHUSIAST SHOULD KNOW*, Recuperado el 20 de mayo de 2023, 2017. dirección: <https://botlink.com/blog/2017/12/18/10-terms-and-abbreviations-that-every-drone-enthusiast-should-know>.
 - [11] Google, *Introducción a Android Studio*, Recuperado el 20 de mayo de 2023, 0. dirección: <https://developer.android.com/studio/intro?hl=es-419>.

- [12] Gradle, *What is Gradle?* Recuperado el 20 de mayo de 2023, 2018. dirección: https://docs.gradle.org/current/userguide/what_is_gradle.html.
- [13] IAEROCOL, *7 usos de los drones en la actualidad*, Recuperado el 20 de mayo de 2023, 2021. dirección: <https://iaerocol.co/blog/uso-de-los-drones-en-la-actualidad/>.
- [14] M. D. Network, *Introducción a JSON*, Último acceso: octubre de 2023, 2023. dirección: <https://developer.mozilla.org/es/docs/Learn/JavaScript/Objects/JSON>.
- [15] OptiTrack, *Primex 41 Specs*, Recuperado el 20 de mayo de 2023, 0. dirección: <https://optitrack.com/cameras/primex-41/specs.html>.
- [16] A. Pennington, *What Do Motion Capture Actors Actually Do?* Recuperado el 20 de mayo de 2023, 2018. dirección: <https://www.backstage.com/uk/magazine/article/trends-and-intelligence-motion-capture-performance-66001/>.
- [17] C. Perafán, *Robotat: un ecosistema robótico de captura de movimiento y comunicación inalámbrica*, Tesis de licenciatura, Universidad del Valle de Guatemala, 2022.
- [18] Pix4D, *PIX4D*, <https://www.pix4d.com/blog/digital-twin-high-rise-building-pix4dinspect/>, Recuperado el 17 de abril de 2023, 2023.
- [19] RiyazPishori, gourdsay, vijakum et al., *Overview of Azure FarmBeats*, 2023. dirección: <https://learn.microsoft.com/en-us/azure/industry/agriculture/overview-azure-farmbeats>.
- [20] Skycatch, *Skycatch*, <https://skycatch.com/about>, Recuperado el 13 de abril de 2023, 2023.
- [21] Mo-Sys, *What is motion capture and how does it work?* Recuperado el 20 de mayo de 2023, 2020. dirección: <https://www.mo-sys.com/what-is-motion-capture-and-how-does-it-work/>.