UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE TECNOLOGÍAS DE LA INFORMACIÓN, TELECOMUNICACIONES E INDUSTRIAL

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y COMUNICACIONES

Tema:

SISTEMA AUTÓNOMO DE PULVERIZACIÓN PARA FUMIGACIÓN DE PLANTACIONES DE FRUTILLA ASISTIDO POR UN DRONE

Trabajo de Graduación. Modalidad: Proyecto de Investigación, presentado previo la obtención del título de Ingeniero en Electrónica y Comunicaciones.

SUBLINEA DE INVESTIGACIÓN: Sistemas Embebidos.

AUTOR: Diego Fernando Mayanquer Gavilanez.

TUTOR: Ing. Mg. Santiago Manzano Villafuerte.

Ambato – Ecuador

Mayo 2019

APROBACIÓN DEL TUTOR

En mi calidad de tutor del Trabajo de Investigación sobre el tema: "SISTEMA AUTÓNOMO DE PULVERIZACIÓN PARA FUMIGACIÓN DE PLANTACIONES DE FRUTILLA ASISTIDO POR UN DRONE", del señor Diego Fernando Mayanquer Gavilanez, estudiante de la carrera de Ingeniería en Electrónica y Comunicaciones, de la Facultad de Tecnologías de la Información, Telecomunicaciones e Industrial, de la Universidad Técnica de Ambato, considero que el informe investigativo reúne los requisitos suficientes para que continúe con los trámites y consiguiente aprobación de conformidad con el numeral 7.2 de los Lineamientos Generales para la aplicación de Instructivos de las Modalidades de Titulación de las Facultades de la Universidad Técnica de Ambato.

Ambato mayo, 2019

EL TUTOR

Ing. Mg. Santiago Manzano.

AUTORÍA

El presente Proyecto de Investigación titulado: "SISTEMA AUTÓNOMO DE PULVERIZACIÓN PARA FUMIGACIÓN DE PLANTACIONES DE FRUTILLA ASISTIDO POR UN DRONE", es absolutamente original, auténtico y personal, en tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato mayo, 2019

Diego Fernando Mayanquer Gavilanez.

CC: 1803728714

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que haga uso de este Trabajo de Titulación como un documento disponible para la lectura, consulta y procesos de investigación.

Cedo los derechos de mi Trabajo de Titulación, con fines de difusión pública, además autorizo su reproducción dentro de las regulaciones de la Universidad.

Ambato mayo, 2019

Diego Fernando Mayanquer Gavilanez.

CC: 1803728714

APROBACIÓN DE LA COMISIÓN CALIFICADORA

La Comisión Calificadora del presente trabajo conformada por los señores docentes Ing. Patricio Encalada e Ing. Marco Jurado revisó y aprobó el Informe Final del Proyecto de Investigación titulado "SISTEMA AUTÓNOMO DE PULVERIZACIÓN PARA FUMIGACIÓN DE PLANTACIONES DE FRUTILLA ASISTIDO POR UN DRONE", presentado por el señor Diego Fernando Mayanquer Gavilanez de acuerdo al numeral 9.1 de los Lineamientos Generales para la aplicación de Instructivos de las Modalidades de Titulación de las Facultades de la Universidad Técnica de Ambato.

Ing. Elsa Pilar Urrutia Urrutia, Mg.
PRESIDENTA DEL TRIBUNAL

Ing. Patricio Encalada, Mg.
DOCENTE CALIFICADOR

Ing. Marco Jurado Lozada, Mg.
DOCENTE CALIFICADOR

DEDICATORIA:

A mi Madre,

Por ser esa persona incondicional quien siempre está a mi lado ayudándome en cada uno de mis anhelos, por toda su paciencia, atención, apoyo y sobretodo su amor sin límites que me brinda día a día, siendo mi más grande motivación de ser cada vez mejor.

A mi Padre,

Por cada una de sus lecciones llenas de valores y educación las cuales me llenan de carácter y perseverancia en el paso de mi vida; por su sacrificio invaluable en cada alba para que esta meta se haya hecho realidad.

A mi hermano,

Por todos y cada uno de los momentos compartidos, por poder encontrar una motivación siempre en él.

Fernando Mayanquer.

AGRADECIMIENTO:

Principalmente a Dios, por llenarme de sabiduría e inteligencia, por ayudarme a cumplir mis sueños con su incondicional amparo.

A mi núcleo familiar, por ser el soporte motivacional y económico en el trascurso diario de esta deseada meta.

A mis profesores, por su arduo trabajo de hacer llegar su conocimiento a lo largo de mi formación profesional; por su amistad y cordialidad brindada dentro y fuera de la Alma Mater.

A mis amigos y amigas, por cada anécdota, cada inolvidable momento compartido.

Fernando Mayanquer.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

En el presente proyecto de investigación, se implementó un sistema de pulverización para la fumigación autónoma de plantaciones de frutilla asistido por un drone; el cual tiene como objetivo dar un correcto cubrimiento al follaje de estas plantaciones en el momento de realizar la aspersión del producto fitosanitario, esto se resume en una disminución de tiempo y recursos económicos, además con la ayuda de la tecnología de drones se trata de impedir los posibles envenenamientos causantes de cáncer en personas que se encuentran expuestas a productos de fumigación, puesto que será el operador quien dé instrucciones de mando al aeronave a una distancia lejana y arbitraria para así contraer el contacto directo de respiración de los químicos existentes en los productos de fumigación.

El drone que asiste al sistema de pulverización en este prototipo vuela con autonomía de aproximadamente 20 minutos con la disposición de una aplicación de punto de ruta, creada dentro de una Interfaz de software libre Android, en ésta se muestra una vista de mapa con la ubicación en tiempo real de la aeronave, además es aquí donde se parametriza la velocidad y la altura de vuelo, el modo de giro del UAV de acuerdo al rumbo de cada punto de ruta, los puntos de vuelo por donde el drone hace su recorrido y pulveriza el material y la acción a realizar de la aeronave después de terminar su recorrido.

El sistema de pulverización acoplado al drone cuenta con la implementación de una bomba de agua (motor DC), una batería de polímero de litio de 7.4 V y 300mAh, dos boquillas de aspersión uniforme, dos tanques contenedores de 250 cm^3 de capacidad y sus respectivos conductos; una vez activado la bomba de agua, el líquido fitosanitario hace el ingreso por ésta y es expulsado de forma tangencial al exterior debido a la trayectoria circular y el movimiento de los álabes que la bomba genera; finalizando su trayectoria en las boquillas pulverizadoras de abanico plano que emanan el producto fitosanitario en aspersión uniforme para un correcto cubrimiento del follaje del fruto.

Palabras Clave: Drone, Sistema de pulverización, Android, Puntos de ruta, Plantaciones de frutilla.

SUMMARY

In the present research project, a spray system was implemented for the autonomous fumigation of strawberry plantations assisted by a drone; which aims to give a correct coverage to the foliage of these plantations at the time of spraying the phytosanitary product, this is summarized in a decrease of time and economic resources, also with the help of drone technology is to prevent the possible poisonings that cause cancer in people who are exposed to fumigation products, since it will be the operator who will give the aircraft control instructions at a far and arbitrary distance in order to contract the direct breathing contact of the chemicals in the products of fumigation.

The drone that assists the spray system in this prototype flies with autonomy of approximately 20 minutes with the disposition of a point of route application, created within an Android free software interface, in this it is shown a map view with the location It is also here where the speed and flight height are parameterized, the UAV rotation mode according to the direction of each waypoint, the flight points where the drone makes its way and pulverizes the material and the action to be taken of the aircraft after finishing its journey.

The spray system coupled to the drone has the implementation of a water pump (DC motor), a lithium polymer battery of 7.4 V and 300mAh, two uniform spray nozzles, two container tanks of 250 cm3 capacity and their respective conduits; once activated the water pump, the phytosanitary liquid makes the entrance for this one and is expelled of tangential form to the exterior due to the circular trajectory and the movement of the blades that the pump generates; finalizing its trajectory in the spray nozzles of flat fan that emanate the phytosanitary product in uniform spray for a correct covering of the foliage of the fruit.

Keywords: Drone, Spray system, Android, Waypoints, Strawberry plants.

INTRODUCCIÓN

En los últimos días los vehículos aéreos no tripulados (UAV), se han transformado en una de las tecnologías más atractivas, dando cabida al uso de estos en una amplia gama de profesiones, como el periodismo, las técnicas de ayuda en rescate de personas con difícil accesibilidad, la fotografía, la agronomía, entre otras.

Si se enfoca en el campo de la agricultura, los drones están cumpliendo un rol cada vez más fundamental para las tareas de procesos fitosanitarios, prometiendo cambiar para siempre el futuro de las explotaciones agrícolas. Por consiguiente, en este proyecto se ha decidido utilizar esta tecnología para desarrollar un prototipo basado en un sistema de pulverización para la fumigación de plantaciones de frutilla, con la finalidad de lograr una correcta cobertura en el follaje de estas plantaciones y a su vez tratar de evitar los posibles envenenamientos de las personas que están expuestas directamente con el producto fitosanitario.

Las tareas realizadas por los drones cumplen una función importante dentro de la seguridad y facilidad de los operadores, por lo que este proyecto aporta también un mejoramiento en la fumigación, principalmente en el cubrimiento adecuado de producto fumigador en las plantaciones. A continuación, se describe los capítulos que estructuran este proyecto:

En el capítulo I se describe el problema, donde se hace mención de las enfermedades más comunes a las que están expuestas las plantaciones de frutilla, conjuntamente a esto se estudia los efectos adversos que se presentan en la salud de personas que se encuentran en contacto directo con materiales de fumigación; además, en este apartado se añade la justificación la cual muestra el por qué se busca realizar este proyecto, y por último los objetivos con los cuales se cumple a cabalidad para el desarrollo de esta investigación.

El capítulo II, constituye el desarrollo del Marco Teórico, en el cual se muestra antecedentes investigativos de sistemas de fumigación acoplados en vehículos aéreos no tripulados, así mismo se describe semblantes teóricos para comprender los sistemas de pulverización, como la cantidad de volumen para la aplicación de líquido fitosanitario, la cantidad de caudal que emergen las boquillas, etc. de la misma forma se analiza las diferentes patologías causantes de problemas que radican en la mala producción en las plantaciones de frutilla, a que parte de la plantación afecta y las posibles alternativas de manejo con productos fitosanitarios.

El Capítulo III detalla la Metodología empleada en este proyecto, donde se presenta las diferentes técnicas de investigación para su desarrollo, así como la recolección de información suficiente y el análisis de datos necesarios; además de las actividades que se cumplieron para el desarrollo de este proyecto.

El Capítulo IV muestra el desarrollo de la propuesta en donde se analiza los requerimientos tanto de software como de hardware para el desarrollo e implementación del sistema de pulverización para plantaciones de frutilla, conjuntamente se muestra el análisis de la altura, velocidad de vuelo, recorrido por los puntos de ruta y condiciones ambientales para llevar a cabo la misión y por último se hace un estudio comparativo de la cobertura que tiene el uso de bombas de espalda y el uso de este prototipo.

Finalmente, el Capítulo V describe las Conclusiones y Recomendaciones que se genera como resultado del desarrollo de esta investigación.

CAPÍTULO I
EL PROBLEMA

1.1 Tema.

"Sistema autónomo de pulverización para fumigación de plantaciones de frutilla asistido por un drone."

1.2 Planteamiento del problema

Sin duda alguna, la agricultura comercial en donde se maneja grandes hectáreas de cultivo es considerada como una actividad peligrosa para la salud del ser humano puesto que, el uso de sustancias químicas para el tratamiento de fumigación puede generar posibles envenenamientos en personas que se encuentran expuestas a estas.

El glifosato es uno de los activos químicos más utilizado en principios de pulverización y el herbicida más utilizado alrededor del mundo, el cual ha sido incorporado a la lista de sustancias 2A según la Agencia Internacional para la Investigación sobre el Cáncer IARC, esta lista asocia sustancias con un 80% de probabilidad de causar cáncer en humanos. [1]

El glifosato causa daño en el ADN y en los cromosomas de las células humanas; en Buenos Aires en la población de Monte Maíz se realizó una investigación por parte de la Universidad de Córdoba en el año 2015, en donde se descubrió que la tasa bruta de incidencia de cáncer, tanto de piel como pulmonar, es de 707 por cada 100.000 habitantes; uno de los agentes principales que causaría esta patología mortal sería el contacto directo que se tiene con el glifosato, un herbicida usado en las plantaciones de soja transgénica en esta ciudad. [2]

Según el libro “La otra Guerra” el cual estudia la situación de los plaguicidas en el Ecuador, realizado en Quito por Alexander Naranjo en 2017 menciona que, para el año 2012 se presentaron 2126 casos de intoxicaciones por pesticidas en el Ecuador, mientras que en 2013 se obtuvieron 1877 casos, siendo Manabí la provincia donde se encuentra más incidencia de esta contrariedad con 306 y 274 casos para cada año respectivamente. Si bien se tiene una tendencia decreciente, el número de casos presentes es alarmante, además a esto, las personas intoxicadas solo acuden a un centro de salud si a su juicio se trata de un caso grave, por el contrario, si los síntomas son leves como dolores de cabeza, se los trata en el hogar. [3]

En la tesis “Uso de plaguicidas en el cultivo de papa” realizada por el ingeniero agrónomo Nelson Villacrés en el año 2014, determinó que las poblaciones de Guangaló e Hipolongo del cantón Quero en la provincia de Tungurahua son las más afectadas por el mal empleo de los químicos utilizados en procesos de fumigación; las conclusiones de dicha investigación muestran que, el 32% de agricultores presenta dolores de cabeza, el 19% mareo y náuseas y el 9% irritación de la piel, conjuntamente el mismo estudio aclaró que el 87% de agricultores no conoce el impacto que produce los químicos en las vías respiratorias y el sistema digestivo; y su relación con el cáncer. [4]

Por otra parte, las plantaciones de frutilla son afectadas por diversos microorganismos habitantes en el suelo, mismos que afectan de forma notable el sistema radicular y cuello de estos cultivos, cuya enfermedad es conocida comúnmente como el corazón rojizo (Phytophthora fragariae) dificultando la absorción de agua y nutrientes; además se presentan diversos síntomas en la copa de las plantas asociados con la amarillez y necrosis en los tejidos, teniendo como enemigo a un sinnúmero de enfermedades como la pudrición gris (Botrytis cinerea) o Rizoctiosis (Rhizoctia solani). Por consiguiente, se obtiene frutos de menor tamaño, menor producción y disminución de la vida útil de las plantas e incluso la muerte de estas cuando el manejo correctivo se realiza demasiado tarde. [5]

1.3 Delimitación

Delimitación de contenidos

Área académica: Física y Electrónica.

Línea de investigación: Sistemas de control.

Sublínea de investigación: Sistemas Embebidos.

Delimitación espacial.

El presente proyecto se desarrolló en la Facultad de Tecnologías de la Información, Telecomunicaciones e Industrial de la Universidad Técnica de Ambato.

Delimitación temporal

El proyecto se realizó en el periodo marzo 2018 – mayo 2019, de acuerdo con lo establecido en el reglamento de graduación para obtener el Título de Tercer Nivel de la Universidad Técnica de Ambato.

1.4 Justificación

Los vehículos aéreos no tripulados, se han transformado en una de las tecnologías más atractivas y publicitadas, un claro ejemplo de esto es que la empresa Yamaha de Japón cuenta con 2500 drones sobrevolando los campos agrícolas de este país, mejorando así la calidad de vida de los agricultores de este país. [6]

Día a día, decenas de cultivos de frutilla son fumigados por agricultores que utilizan el método tradicional de bombas de espalda para mejorar la calidad de su producto, exponiéndose así de forma directa a los activos químicos que se utiliza para este proceso fitosanitario, lo que a corto o a largo plazo se traduce en la aparición de enfermedades en estas personas; es por lo que nace la necesidad de desarrollar un sistema autónomo de pulverización para la fumigación de plantaciones de frutilla asistido por un drone, para de esta manera dar un nuevo enfoque en el proceso de fumigación de cultivos en el sector agropecuario.

Por consiguiente, los agricultores enfocados en este fruto serán los principales beneficiarios, puesto que la importancia de este proyecto es la reducción de

envenenamientos y apariciones de enfermedades graves como el cáncer, ya que será el drone el encargado de pulverizar el material fitosanitario sobre los cultivos de frutilla de forma autónoma, además se reduce en gran manera los tiempos, costos y recursos que demanda el proceso de fumigación.

En conclusión, la investigación es factible ya que el drone y los materiales electrónicos necesarios para el sistema de tratamiento fitosanitario, como pulverizadores y bombas segregadoras de pesticidas, no tienen una dificultad elevada de adquisición, además se cuenta con el presupuesto que demandará la investigación.

1.5 Objetivos

1.5.1 Objetivo General

Implementar un sistema autónomo de pulverización para fumigación de plantaciones de frutilla asistido por un drone.

1.5.2 Objetivos Específicos

- Analizar el tratamiento de fumigación de plantaciones de frutilla.
- Analizar las características técnicas del dron para el sistema autónomo de pulverización.
- Diseñar la interfaz remota para el control del sistema autónomo de pulverización.

CAPÍTULO II

MARCO TEÓRICO

2.1 Antecedentes investigativos.

La investigación bibliográfica se realizó con la ayuda de artículos científicos de diferentes repositorios digitales de Ingeniería tanto nacionales como internacionales, se encontró proyectos afines, los cuales han servido de soporte para este trabajo de investigación.

D. Yallappa, en su artículo científico "Development and evaluation of drone mounted sprayer for pesticide applications to crops" dictado en la Conferencia Global de Tecnología Humanitaria (GHTC) realizada por la IEEE en San Jose, CA, Estados Unidos, en el año 2017, hace mención del desarrollo de un prototipo de pulverizador montado en un dron de control manual, mismo que cuenta con motores BLDC, cámara FTP, sensores y placas de circuitos montados en el bastidor de aire. Este prototipo fue diseñado para el estudio y evaluación de la velocidad de descarga, tamaño y densidad de gotas, anchura de barrido y uniformidad de pulverización, donde uno de los resultados que se presenta es que a 150 cm de altura se obtiene un 99% de uniformidad de esta manera Yallapa concluye que la altura del pulverizador es proporcional a la uniformidad de pulverización. [7]

Swati D Kale, en su artículo realizado en Pune, Maharashtra, India "Agriculture Drone for Spraying Fertilizer and Pesticides" publicado en la Revista Internacional de Investigación Avanzada en Ciencias de la Computación e Ingeniería de Software en el año 2015, lleva a cabo la descripción de una arquitectura basada en vehículos aéreos

no tripulados (UAVs), empleados para la implementación de un circuito de control mismo que, realiza el proceso de aplicación de producto químico mediante el control de la información obtenida por la red de sensores inalámbricos dispersos en el campo a fumigar, este proyecto utiliza un microcontrolador ATmega328, ESC para el control de los motores BLDC, sensores acelerómetro y giroscópico; mientras que el área de software maneja Arduino, EAGLE (Easily Applicable Graphical Layout Editor) y Multi Wii. [8]

Fuminori Matsuura en su artículo "Research of Crop-sprayer for Dotted Farmland using Airflow Induced by UAV" publicado en la Conferencia Anual de la Sociedad de Ingenieros de Control e Instrumentos de Japón (SICE) en el año 2017, realiza una investigación acerca del efecto de la velocidad del viento inducido por los rotores de los UAVs y su incidencia en la penetración de los productos químicos en los cultivos, con el fin de averiguar la configuración óptima para la pulverización de estos, en donde se utiliza motores con diferentes hélices trabajando a distinta velocidad y simuladores para su experimentación; luego de esta investigación se dice que la posición de la boquilla y la velocidad del viento inducido por el rotor no afecta a la relación del área de pulverización del vehículo aéreo no tripulado. [9]

Mientras tanto en el proyecto "Monitorización de Cultivos Utilizando Drones", realizado por Oliver Núñez y Tania Figueroa en el Centro Educativo Cruz Azul, Oaxaca México en 2015, se pretende realizar reconocimientos aéreos sobre superficies de cultivos a través de la tecnología de drones, en donde se pueda detectar con cierta facilidad plagas nocivas al plantío obteniendo fotografías desde el aire para de esta manera seguir una tendencia más general del uso de sensores y la robótica e incorporar así, el uso de datos a la agricultura de precisión; en otras palabras este proyecto establece un antecedente de los elementos requeridos para el uso de los drones en los procesos de seguimiento, optimización, producción y mejora de las cosechas en este país. [10]

En Tamaulipas México, Daniel Soto Guerrero, en el año 2012, realizó el trabajo "Interacción hombre-robot con vehículos aéreos no tripulados basada en visión" en donde se propone el desarrollo de una interfaz entre un vehículo aéreo no tripulado (UAV por sus siglas en ingles) y una persona. La interfaz realiza el rastreo de una

persona y el reconocimiento de sus órdenes gestuales, por medio de visión por computadora. Generalmente, se emplea una estación de trabajo fija en tierra para implementar el procesamiento digital de imágenes, limitando la aplicación del vehículo al rango de comunicación entre ambas partes. Para evitar el uso de una estación fija en tierra, el control del dron, y lograr una mayor autonomía del uso de la interfaz, se propone que el procesamiento computacional de las imágenes se realice en un dispositivo móvil portado por el usuario. [11]

Según la revista ICT Update en su edición 82 "Drones para la agricultura" publicada en el año 2016 en Estados Unidos, se realiza proyectos utilizando la tecnología de los drones para de esta manera proporcionar a los agricultores un método rentable para la planificación de la agricultura, acelerando la misma y a su vez haciendo más eficiente la construcción de los sistemas de riego de las plantaciones de arroz. [12]

2.2 Fundamentación teórica.

2.2.1 Fumigación y manejo sanitario

La fumigación trata del procedimiento destinado a la protección vegetal con el uso de productos fitosanitarios, principalmente de origen químico para combatir y controlar plagas y/o transmisores de enfermedades bacterianas o virales de los cultivos. [13]

Los productos fitosanitarios, según la OMS se define como la sustancia o mezcla de sustancias destinadas a prevenir el daño vegetal, destruyendo de forma directa insectos, ácaros, moluscos, roedores, hongos, malas hierbas, bacterias y otras formas de vida animal o vegetal perjudiciales para la salud pública y también para la agricultura. Inclúyase en este ítem los plaguicidas, insecticidas, herbicidas, fungicidas, defoliantes, desecantes y las sustancias reguladoras del crecimiento vegetal o fitorreguladores. [14]

Es importante mencionar que para el manejo sanitario de las plantaciones de frutilla existen acciones que se deben cumplir de manera obligatoria como, el uso alternado de productos de diferentes grupos químicos para el control de enfermedades y plagas, o a su vez el monitoreo del pH del agua que va a ser manejada para los tratamientos fitosanitarios. [15]

Dependiendo del producto a utilizar, entre las principales ventajas de la aplicación de fumigantes para el cultivo de frutilla se puede mencionar. [16]

- Controlan un amplio espectro de problemas sanitarios: enfermedades radiculares, insectos del suelo, nematodos y algunas malezas relacionadas a la germinación y las semillas.
- Favorecen al establecimiento del cultivo
- No tiene efecto residual.

2.2.2 La frutilla.

La frutilla es una planta de la familia de las rosáceas, con tallos rastreros, nudosos y con estolones, hojas pecioladas, vellosas y de color blanquecino por el envés, divididas en tres segmentos aovados y con dientes gruesos en el margen; flores pedunculadas de un color blanco o amarillento, solitarias o en corimbos poco nutridos, y fruto casi redondo, algo apuntado, rojo y comestible. [17]

2.2.3 Enfermedades de la frutilla

La frutilla es afectada muchas de las veces por diferentes enfermedades patológicas que atacan la longevidad de la planta, el rendimiento y la calidad del fruto, estas enfermedades se reconocen por los síntomas que causan.

Para llegar a tener una mayor rentabilidad en las plantaciones de frutilla es necesario la oportuna identificación y determinación de las patologías que se puedan presentar en estas plantaciones, con la finalidad de mantener la calidad y sanidad de los cultivos de frutilla.

Una correcta evaluación permitirá un control eficiente, es por lo que es de suma importancia examinar en agente causante de las distintas enfermedades.

Ocasionalmente se pueden detectar otros patógenos, pero su importancia es mucho menor en comparación a las descritas en la tabla 1, y por lo general no se necesita de un manejo específico o se controlan indirectamente cuando se manejan las enfermedades de importancia.

En la tabla 1, se muestran las principales enfermedades que afectan a las plantaciones de frutilla.

Tabla 1. – Patologías de la frutilla, indicando la zona en donde afectan [18]

Nombre Común	Nombre Científico	Zona Afectada
Más Frecuentes		
Corazón rojizo	*Phytophthora fragariae*	Raíces y daño general de la planta.
Oídio, peste ceniza	*Sphaerotheca macularis*	Hojas, brotes y frutos
Tizón de la hoja	*Phomopsis obscurans*	Hojas
Pudrición gris	*Botrytis cenerea*	Flores, frutos y hojas
Viruela	*Ramularia tulasnei*	Hojas, tallos y frutos
Verticilosis	*Verticillium dahliae*	Raíces y daño general de la planta.
Pudrición de la corona	*Phytophthora cactorum*	Raíces y cuello de las plantas.
Rizoctoniasis	*Rhizoctonia solani*	Raíces y planta en general.
Ocasionalmente		
Fusariosis	Fusarium oxysporum	Raíces y decaimiento de la planta.
Pudrición blanca.	Sclerotinia sclerotiorum	Frutos y follaje.
Mancha negra de la hoja	Colletotrichum gloeosporoides	Hojas.
Mancha necrótica de la hoja	Coniella fragariae	Hojas
Quemadura de la hoja	Diplocarpon earlianum	Hojas.
Mancha de la hoja	Gnomonia comari, Hainesia lythri	Hojas
Pudrición carbonosa	Macrophmina phaseolina	Raíces.
Virosis	Straberry latent ringspot (SLRSV)	Hojas, brotes, flores y frutos.
Nematosis	Pratylenchus, Xiphinema, Meloidogyne, Criconemoides.	Raíces y decaimiento general de la planta.

1. Corazón rojizo *(Phytophthora fragariae)*

Descripción: El corazón rojizo es un patógeno que se disemina a través de esporas flageladas, llamadas zoosporas; la producción de las zoosporas se debe al exceso de riego, lluvia o mal drenaje del suelo, el inóculo puede provenir de plantas enfermas, agua de riego o tierra contaminada incluso en implementos agrícolas.

Es una patología frecuente en plantaciones mal manejadas, que sufren daños por insectos en las raíces.

Síntomas: La figura 1 muestra cómo afecta el corazón rojizo a las raíces de la planta, teniendo como principal síntoma un color rojizo oscuro y un desprendimiento fácil de las mismas; esto hace que la absorción del agua y nutrientes se dificulte lo que se traduce en daños aéreos como la marchitez y necrosis de las hojas, marchitez del follaje, perdida de flores y frutos e incluso se corre el riesgo que la planta se seque por completo.

Fig. 1: Corazón rojizo, provocado por Phytophthora fragariae [18]

Recomendaciones de manejo.

Una de las recomendaciones de manejo para el control del corazón rojizo es que no se debe plantar en suelos con problemas de drenaje, sectores bajos y susceptibles a inundarse pero la mejor medida de control es plantar en camellones altos para mejorar el drenaje en la zona del cuello y aireación de las raíces; también se recomienda iniciar el cultivo con la inoculación con *Trichoderma* para otorgar mayor protección contra enfermedades de la raíz o en el caso de presentarse la enfermedad, se debe eliminar las plantas con síntomas y aplicar fungicidas

granulares o líquidos, previa revisión de la vigencia en el registro de productos autorizados para el cultivo, destacando: metalaxil, mefenoxam o fosetil aluminio. Las aplicaciones de fosfitos también ayudan a prevenir la enfermedad o disminuir los daños, siempre y cuando estas aplicaciones sean regulares y antes de que aparezcan los síntomas. [18]

2. Oídio *(Sphaerotheca macularis f. sp. fragariae)*

Descripción: El Oídio es una de las enfermedades más comunes, populares y fáciles de examinar en las plantaciones de frutilla. Este patógeno restringe severamente la fotosíntesis de la planta, causando distorsión en las hojas, además genera micelio que solo crece sobre la superficie de las plantas, sin invadir su interior; esta enfermedad no mata la planta, pero si consume sus nutrientes y reduce el crecimiento, rendimiento y calidad de ésta.

Síntomas: El perjuicio que causa el oídio se restringe a las células de la epidermis, traduciéndose a necrosis, distorsión de las hojas y deformaciones en los frutos, como se muestra en la figura 2. Además, aparecen manchas circulares y difusas de apariencia polvorienta, como depósitos de polvillo blanquecino sobre el área de los tejidos aéreos, cabe mencionar que cualquier parte aérea de la planta puede ser afectada pero normalmente esta enfermedad se centra en las hojas, peciolos y frutos.

Si esta enfermedad llega a las flores significará una menor producción de polen, en los frutos verdes causa deformaciones en los mismos y a signos avanzados, impedirá el crecimiento y secará el follaje.

Fig. 2: Oídio, dada por Sphaerotheca macularis. [18]

Recomendaciones de manejo.

Para controlar la enfermedad del oídio se utiliza comúnmente azufre elemental; el manejo se debe dar de forma preventiva y es efectivo siempre y cuando se aplique antes de la aparición de síntomas. Los fungicidas sistémicos son más eficientes que el azufre, pero tienen un mayor costo, estos pueden ser: benomyl, carbendazim, cyprodinil y fludioxanil.

Otras alternativas de control son las aplicaciones foliares de sales de fosfato, detergentes y aceites finos, mismos que pueden ayudar a disminuir las aplicaciones de fungicidas químicos y adaptarse a producción orgánica. [18]

3. Pudrición gris *(Botrytis cinerea)*

Descripción: La pudrición gris es la principal enfermedad de la planta de frutilla. Puede atacar a cualquier zona, siendo las flores y frutos los más susceptibles, manifestándose como una pudrición blanda del fruto, con presencia de micelio y conidias de color plomizo.

Síntomas: Los principales síntomas de esta enfermedad viene acompañada de ablandamiento y secreción de jugo del fruto. Cuando la enfermedad se presenta en un estado medio la pudrición blanda va también presenta ligeros cambios de color en el fruto infectado, como se puede identificar en la figura 3.
En las hojas y flores se observan masas de micelio y conidias plomizas sobre los tejidos, además en los tallos se observan lesiones plomizas que forman anillos concéntricos.

Fig. 3: Frutos con pudrición gris. Estado avanzado. [18]

Recomendaciones de manejo.

Para el manejo de la enfermedad de pudrición gris se debe sembrar cada planta respetando las distancias de las hileras para lograr una suficiente aireación y favorecer al secado del follaje; es importante también abastecer de algunas dosis de nitrógeno y calcio para prevenir el aparecimiento del hongo, además se debe realizar un control químico al momento de la floración, después de las lluvias y temperaturas mayores a los 15 ºC, rotando ingredientes activos registrados y autorizados para no generar resistencia.

Entre los productos excelente acción contra el Botrytis están azoxystrobin, boscalid, cyprodinil, pyraclostrobin y mezclas de sistémico y contacto como boscalid + pyraclostrobin.

El control biológico es otra opción, con productos sobre la base de *Bacillus subtilis o Trichoderma*, pero se deben anticipar a la aparición de síntomas y no aplicarlo en exceso porque resulta fitotóxico para la plantación. [18]

4. Viruela *(Ramularia tulasnei)*

Descripción: La viruela es una de las patologías principales del follaje en la frutilla y es muy fácil de reconocer por las pústulas que se presentan en las hojas de las plantaciones. La incidencia de la viruela depende de la susceptibilidad de las variedades y las condiciones climáticas; la viruela requiere de temperaturas en torno a los 20°C a 25°C para desarrollarse y la lluvia actúa como agente diseminador del inóculo; además, a mayor cantidad de humedad mayor será la incidencia de la enfermedad.

Síntomas: Los principales indicios que presenta la viruela son las pústulas de bordes bien definidos y de color púrpura, con el centro color café claro o plomizo, como se muestra en la figura 4.

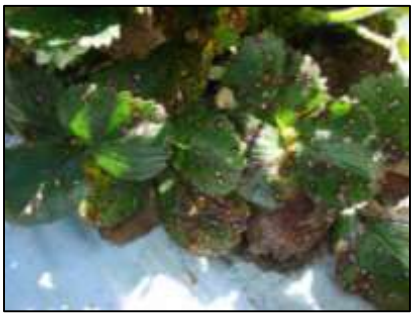


Fig. 4: Necrosis, características de Ramularia tulasnei. [18]

Recomendaciones de manejo.

Las plantaciones deben ser bien ventiladas, favoreciendo la orientación de las hileras en el sentido en que predominan los vientos, esto favorecerá al secado de las hojas después de una lluvia y le dan menos posibilidades de que el hongo pueda germinar, esta enfermedad se puede ser tratada con el manejo que se le da a la pudrición gris, siendo efectiva la poda y eliminación de las hojas enfermas.

Algunos de los fungicidas que se utilizan para el control de esta patología son: clorotalonil, iprodione, azoxystrobin, cyprodinil y fludioxanil. [18]

5. Rizoctoniasis *(Rhizoctonia solani)*

Descripción: La enfermedad de Rizoctoniasis se da por el hongo *Rhizoctonia solani*, este hongo es habitante normal del suelo, el cual afecta numerosas especies de hortícolas y cultivos anuales, pudiendo producir pérdidas importantes como ocurre en el caso de la remolacha, papas y leguminosas.

En el caso de la frutilla aparece sola, formando un conjunto de hongos radiculares.

Síntomas: Los síntomas que se presentan en esta patología son: presencia de clorosis, disminución del crecimiento, aborto de flores, fruta que demora en madurar, bajo calibre o frutos que se secan en la planta. En la figura 5 se muestra

la parte radical con necrosis parcial de raíces primarias, las que adquieren una coloración negra y deshidratada.

Fig. 5: Rizoctoniasis causada por Rhizoctonia solani. [18]

Recomendaciones de manejo.

La enfermedad de Rizoctoniasis se debe prevenir, ya que una vez que se presenta resulta muy difícil o imposible sanar a las plantas. Una recomendación preventiva es evitar suelos con exceso de humedad y altos contenidos de nitrógeno, ya que favorecen su desarrollo.

Otra medida de prevenir es evitar realizar el cultivo después de haber cosechado papas o leguminosas en dicho terreno y utilizar camellones altos para una buena aireación de las plantas; en caso de presentarse la enfermedad se debe eliminar las plantas para evitar la diseminación. [18]

2.2.4 Tratamiento Fitosanitario de plantaciones de frutilla.

La producción de frutas y hortalizas apunta a la reducción del uso de agroquímicos, especialmente por el efecto negativo que genera en la salud humana (residuos en productos comestibles) y por contaminación al medio ambiente.

Para el caso de los cultivos de frutilla, la aplicación de plaguicidas por medio de pulverizadores se ve en desmedro, puesto que la tecnología empleada es ineficiente con el uso de bombas de espalda para dicha acción.

Los pulverizadores hidráulicos de mochila (bombas de espalda) son recomendados para aplicaciones de herbicidas, pero en frutales pierden su eficacia puesto que no logran cubrir la zona central del follaje. [18]

Para el tratamiento de fumigación con herbicidas en plantaciones de frutilla se aconseja el uso de boquillas de abanico plano con aspersión uniforme, debido a que estas son las apropiadas para la aspersión de material en bandas; éstas generan un abanico plano uniforme en la aplicación lo que se traduce a una pulverización de gotas uniformes.

Para realizar una correcta fumigación es necesario realizar ciertos cálculos que se mencionan y detallan a continuación.

Velocidad de avance del operador.

Para determinar la velocidad del UAV se marca una distancia mínima de 20 metros y se procede a medir el tiempo en segundos que tardará el drone en recorrer esta distancia, esta velocidad se expresa en kilómetros por hora (m/s) y se calcula con la ecuación 1. [19]

$$v\left(\frac{m}{s}\right) = \frac{d(m)}{t(s)}$$

$Ecuación\ (1)$

Donde:

v: Velocidad en m/s
d: Distancia medida en metros.
t: Tiempo en segundos que tardará el dron.

Medición del ancho de trabajo de pulverización.

La medición del ancho de trabajo dependerá de la extensión de pulverización que logre una boquilla a un metro de altura (valor medido de forma manual), la altura a la que vuele el avión, la distancia entre las boquillas y el número de boquillas; el ancho de trabajo se calcula mediante la ecuación 2.

$$at = (ab * h) + [db(nb - 1)]$$

$Ecuación\ (2)$

Donde:

at: ancho de trabajo del sistema de pulverización.
ab: ancho de asperje de cada boquilla a un metro de altura; este dato se toma a través de medición manual.
h: altura del UAV.
db: distancia entre boquillas pulverizadoras.
nb: número de boquillas pulverizadoras.

Además, el ancho de concentración se calcula con la ecuación 3, la cual tiene gran estreches con la ecuación 2.

$$ac = at - 2\ db$$

$Ecuación\ (3)$

Donde:

ac: ancho de concentración.
at: Ancho de trabajo del sistema de pulverización (1.15 metros)
db: Distancia entre las boquillas (0.15 metros)

Medición del caudal de la boquilla.

Para su medición del caudal de las boquillas se debe aplicar la boquilla dentro de un jarrón graduado durante un minuto, el volumen recibido será el caudal de boquilla y se lo expresa en litros sobre minuto (L/min).

Medición del volumen de material fitosanitario para la aplicación de fumigación

El volumen de material fitosanitario para la aplicación mide el número de litros que se necesita por hectárea para la pulverización.
Con los datos obtenidos anteriormente se procede a determinar el volumen de material fitosanitario para la aplicación, mediante la ecuación 4.

$$Q = \frac{q * 600}{a * v}$$

$Ecuación\ (4)$

Donde:

Q: Volumen de aplicación en litros por hectárea (L/ha)

q: Caudal de la boquilla en litros por minuto (L/min)

a: Ancho de trabajo de la/s boquilla/s en metros (m)

v: Velocidad del operador en kilómetros por hora (Km/h)

2.2.5 Sistema de pulverización.

Pulverizar se refiere a disipar una sustancia líquida en partículas muy pequeñas, es decir, pulverizar significa la destrucción de algo.

La idea de difuminar una sustancia en el agro se utiliza haciendo hincapié a la acción desarrollada por los aviones las cuales pueden acarrear tripulación o no, sin pasar por alto las máquinas que fumigan los campos de manera manual como las bombas de espalda ocupadas como método tradicional por los agricultores. De esta manera la acción de difuminar una sustancia, involucra esparcir algún tipo de material fitosanitario (fungicida, herbicida, plaguicida, insecticida) a través de su pulverización sobre las plantaciones que necesiten tratamiento.

Debido a la acción del barrido que un pulverizador puede generar, estos se pueden clasificar en nebulizadores, vaporizadores, espolvoreadores o atomizadores [20]

Bombas de agua.

Una Bomba de Agua, es un dispositivo que se utiliza para bombear agua de un lugar a otro, sin importar el fluido y a su vez incrementar la energía cinética de su caudal. Hay Bombas de Agua que mueven: Aguas Sucias, Aguas Limpias, fluidos como vino, leche, entre otros.

Se llama Bomba de Agua a cualquiera de esos dispositivos que realizan actividades como:

- Desocupar Piscinas (Aguas Limpias)
- Desocupar Pozos Sépticos (Aguas Sucias)
- Regar Cultivos
- Abastecer de Agua un sitio sin acceso a ella,
- Transportar agua de un lugar a otro, con más o menos presión, etc.

Sin embargo, las bombas de agua de alimentación de 12 Volts en corriente directa son una opción ideal para presurizar agua en un sistema cerrado y son muy importantes en donde se desee ocupar una alimentación limitada, además del tamaño y peso reducido que estas brindan, como muestra la figura 6. [21]

Fig. 6: Bombas de agua.

La mayoría de bombas de agua accionadas por un voltaje DC están constituidas por un motor eléctrico el cual actúa como dispositivo de accionamiento excitado por dicho voltaje, este motor está acoplado a un elemento rotativo llamado rodete, el cual está compuesto por álabes que giran y de esta manera transmiten parte de la energía en forma tangencial al líquido que ingresa a través de la bomba de agua. Cada uno de los álabes que conforman la bomba tienen una pequeña curvatura, esta distorsión constituye una guía para el fluido que lo atraviesa, de tal forma que su energía se maximiza a medida que el líquido hace su recorrido por los álabes, haciendo de la cantidad de fluido directamente proporcional a la velocidad que alcance el motor de este dispositivo [21]

Boquilla

Una boquilla de pulverización es un elemento utilizado para transformar un flujo de líquido en pequeñas partículas de acuerdo a la forma de aspersión que se desee.

Cabe mencionar que se debe tener muy en cuenta el tipo de boquilla que se puede utilizar independientemente del barrido y la aplicación para las que se utilice, tomando en cuenta aspectos como el tamaño de las gotas o la uniformidad de la distribución; existen boquillas de diferentes cualidades las mismas que se nombran a continuación:

- De Abanico: Esta boquilla pulverizadora realiza la pulverización de partículas con una integración de gotas hacia el centro en relación con su contorno. Es por esto que para el uso de estas boquillas se necesitará un caudal suficiente para que el barrido sea el correcto y logre alcanzar su grado de uniformidad
- De cono o turbulencia: Este tipo de boquillas pulverizadoras de material, crean una mayor perturbación lo que se traduce a una pulverización de gotas más finas, esto permite dar una buena cobertura al follaje del cultivo; debido a que su turbulencia forme un barrido en forma circular lo que permite que las gotas lleguen aún por el dorso de las hojas de las plantaciones.
- De espejo; a comparación de las boquillas anteriores, estas originan partículas de mucho más volumen manejando presiones menores a las que se necesitan en boquillas de abanico o de cono.
- De aspersión uniforme; estas boquillas son ideales si se necesita una aplicación en bandas o camellones, su barrido forma un paralelepípedo con una buena concentración sobre los cultivos debido al volumen de las gotas que estas emanan. [22]

Baterías de Polímero de Litio

Las baterías de polímero de litio son baterías de última generación que están formadas de polímero de iones de litio, estas brindan una óptima relación entre capacidad, peso, volumen, voltaje y corriente.

Estas están formadas por elementos de 3.7 Volts los cuales también pueden ser llamadas celdas o células; pueden llegar a alcanzar un conjunto de 8 elementos o más, aumentando así la capacidad de voltaje y corriente de descarga a la carga. [23]

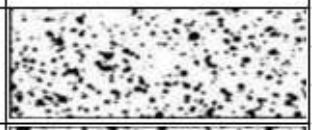
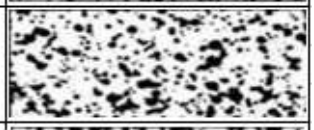
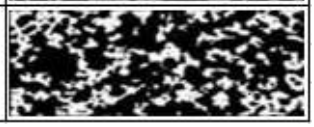
Conductos de agua

Generalmente son dispositivos en especie de tubos de longitud larga y flexible, principalmente hechos a base de goma; estos son utilizados para conducir por su interior un líquido de un lugar a otro, siendo sus extremos, una entrada y una salida independientemente.

Comprobación de la pulverización

Para la comprobación de la aplicación de producto fitosanitario, se utiliza papel hidrosensible, con el fin de verificar la calidad de cubrimiento; estos son de color blanco y se tiñen de azul al contacto con el agua, un papel con buen cubrimiento es aquel que queda con fondo blanco y muchas gotas pequeñas de color azul, es decir con un cubrimiento del 50%, en la tabla 2 se muestra el patrón de referencia para estimar la calidad de la aplicación [19]

Tabla 2. – Patrón de referencia para estimar la calidad de la aplicación [19]

PATRON DE REFERENCIA PARA ESTIMAR LA CALIDAD DE LA APLICACIÓN		
Gotas cm2	Porcentaje de cobertura	Papel hidrosencible
85	10%	
70	20%	
60	30%	
55	40%	
40	50%	

2.2.6 El Drone

Un drone es un aparato volador no tripulado, es decir, hace referencia a que el manejo del mismo se hace a distancia o que puede ser controlado de forma remota a través de radio control. [24]

El funcionamiento de un drone puede parecer muy sencillo, sin embargo, el procedimiento consiste realmente en el seguimiento de un sinnúmero de instrucciones y lectura de sensores en cortos tiempos para que su vuelo se realice de manera óptima.

Los drones cuentan con un sistema multi-hélice ubicado en su interior el que hace que este dispositivo sea autónomo con un margen de error minimizado.

Los drones que poseen un gran número de motores en el interior tienen la capacidad de ganar más control sobre su elevación y, por lo tanto, pueden transportar carga durante el vuelo. Estas hélices consiguen su energía de una fuente dedicada y la mayor parte de estos dispositivos contienen baterías extraíbles, por lo que puede permanecer en el aire y funcionar por bastante tiempo. [25]

Hoy en día los drones se abren campo en una gran gama de actividades como el periodismo, el rescate, la agricultura la cinematografía y otras más, con la gran ventaja de poder volar muy cerca de sus objetivos.

2.2.7 Wi-Fi

Wifi es una tecnología de comunicación inalámbrica que permite conectar equipos electrónicos, como computadoras, tabletas, smartphones, etc., mediante el uso de radiofrecuencias para la trasmisión de la información; wi-fi trabaja a una frecuencia de 2.4 GHz con un alcance de 100 metros o a 5.8 GHz con un alcance de 10 metros, siempre y cuando se tenga línea de visión entre los dispositivos conectados, sin embargo el uso antenas amplificadoras hace que el alcance de comunicación puede extenderse o minimizarse. [26]

2.2.8 Android Studio

Android Studio es el entorno de desarrollo integrado (IDE) oficial para la creación de aplicaciones en Android, el cual está sustentado en IntelliJ IDEA un poderoso editor de código que facilita la creación de aplicaciones gracias a sus numerosas herramientas, este entorno de programación pone a disposición las siguientes funciones las cuales amplían su versatilidad durante la compilación de aplicaciones en Android.

- Un sistema de compilación basado en Gradle flexible.
- Un emulador rápido con varias funciones.
- Un entorno unificado en el cual se puede crear aplicaciones para todos los dispositivos Android, sim importar las características de hardware o software.
- Instant Run para emplear cambios mientras la aplicación se ejecuta sin la necesidad de compilar un nuevo APK.
- Compatibilidad con C++ y NDK [27]

La ventana principal de Android Studio tiene varias áreas lógicas, las cuales se detallan a continuación en la figura 7.

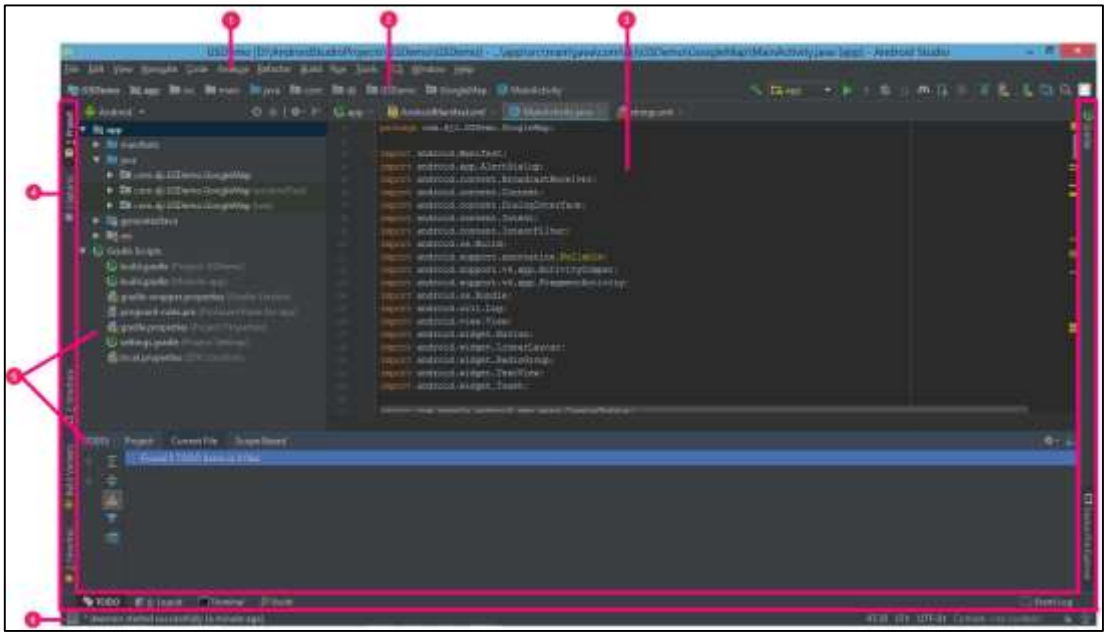


Fig. 7: Interfaz de usuario de Android Studio. [27]

1 La barra de herramientas permite ejecutar una gran variedad de operaciones, como la ejecución de la aplicación y el inicio de herramientas de Android.

2 La barra de navegación permite examinar cada proyecto y abrir archivos para editarlos, adicionalmente provee de una vista más íntegra de la organización visible en la ventana Project.

3 La ventana del editor es el espacio el cual permite crear y editar código. Independientemente de cuál sea el archivo actual, el editor ofrece la ventaja de cambiar ya sea al entorno de programación (código) o al entorno de diseño (gráfico)

4 La barra de la ventana de herramientas está ubicada alrededor de la parte externa de la ventana del IDE la cual contiene los botones que permiten maximizar o minimizar cada una de las ventanas de herramientas.

5 Las ventanas de herramientas dan la facilidad de ingresar a tareas específicas, como las búsquedas, la administración de proyectos o los controles de versión.

6 La barra de estado, como su nombre lo indica, muestra el estado de la aplicación, de igual manera también muestra cualquier advertencia o mensaje que se pueda generar si el código de progresión se escribe de manera incorrecta.

Android Studio muestra los archivos de cada aplicación en la vista de proyectos del IDE, véase la figura 8. Esta ventana se constituye en módulos para de esta manera poder suministrar un acceso vertiginoso a los archivos de origen de cada proyecto.

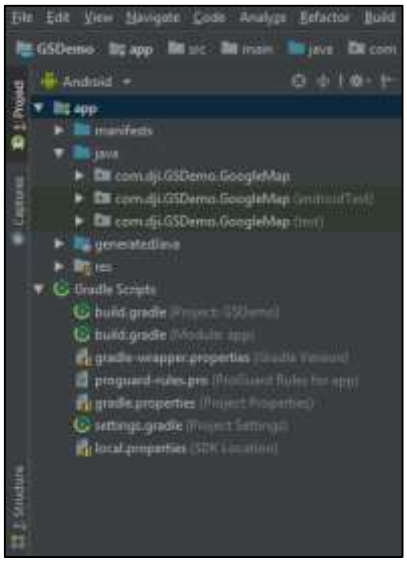


Fig. 8: Archivos del proyecto en Android Studio, desarrollado por el investigador.

Cada una de las carpetas y archivos de compilación contiene las siguientes carpetas:

- **Manifests:** este guarda en su interior el archivo AndroidManifest.xml, en el cual se puede realizar configuraciones básicas del proyecto.
- **java:** contiene los archivos de código fuente de Java, además de esto también contiene el archivo de código de prueba llamado JUnit.
- **res:** Contiene todos los recursos, es decir las cadenas de IU, los diseños XML, además contiene también cada una de las imágenes de mapa de bits. [27]

CAPÍTULO III

METODOLOGÍA

3.1 Modalidad de la Investigación

El proyecto de investigación fue de modalidad aplicada, porque su elaboración se utilizó conocimientos adquiridos durante el proceso de formación superior mientras se aprobaba la malla de estudios de la Carrera de Ingeniería en Electrónica y Comunicaciones de la Facultad de Tecnologías de la Información, Telecomunicaciones e Industrial, aplicando conocimientos en desarrollo e implementación de circuitos electrónicos, así como en programación.

El proyecto fue de carácter bibliográfico, porque éste se basó en fuentes documentales de libros, revistas, artículos, y base de datos científicas disponibles en sitios oficiales y repositorios digitales de universidades y organizaciones nacionales e internacionales, de donde se obtuvo información necesaria para fundamentar el trabajo en base a teorías, resultados experimentales, herramientas de simulación y técnicas de diseño e implementación.

Este proyecto tuvo una investigación de campo, puesto que se procedió a la manipulación de variables externas propias del campo de frutilla, para ser estudiadas y analizadas rigurosamente, con el fin de describir un acontecimiento en particular y poder proceder de forma exacta al tratamiento fitosanitario adecuado mediante sustancias de fumigación.

El proyecto se consideró como investigación experimental, ya que para el desarrollo del trabajo se requirió de pruebas de funcionamiento permitiendo obtener información verídica al momento de exponer los resultados y conclusiones de este.

3.2 Recolección de Información

Para la recolección de información fue necesario el uso de los repositorios digitales y base de datos nacionales como internacionales, que permitan acceder a las fuentes bibliográficas, tutoriales, hojas de especificaciones de los diferentes componentes para de esta manera ampliar la visión del proyecto, además de realizar las pruebas necesarias del prototipo con el fin de observar un adecuado funcionamiento y comparar resultados.

3.3 Procesamiento y Análisis de datos

Para el procesamiento y análisis de datos se tomó en cuenta los siguientes puntos:

- Organización y revisión de la información recolectada
- Análisis de la información con el fin de establecer la mejor alternativa que proporcione una solución al problema.
- Planteamiento de la propuesta de solución.

3.4 Desarrollo del Proyecto

Para el desarrollo del proyecto se tomaron en cuenta las siguientes actividades.

- Analizar los procesos de fumigación en plantaciones de frutilla.
- Reconocer el agente causal asociado a las distintas patologías y malezas que afectan al cultivo, así como el mejor producto fitosanitario para cada una de estas.
- Indagar los drones existentes en el mercado y analizar sus especificaciones técnicas, para definir el apropiado que cumpla los requerimientos para el acople del sistema de pulverización.

- Seleccionar el tanque de fluido, el bastidor de soporte y el tren de aterrizaje que será acoplado en el vehículo aéreo no tripulado.
- Determinar la correcta boquilla para el tratamiento de fumigación que cubra toda la zona central del follaje.
- Realizar el sistema de pulverización, asistido por una bomba de agua DC que efectúe la presión de aire necesaria en la/s boquilla/s.
- Diseñar la interfaz en un dispositivo móvil de sistema operativo y software libre Android, que servirá para la parametrización de datos de los puntos de recorrido y giro localizados con GPS.
- Acoplar el sistema de pulverización a la plataforma cuadcopter.
- Ejecutar pruebas de funcionamiento, verificando la autonomía del drone y el trabajo deseado por el sistema de pulverización.
- Realizar el análisis de resultados y obtención de conclusiones.
- Elaborar el informe final.

CAPÍTULO IV

DESARROLLO DE LA PROPUESTA

4.1 Introducción

En todo el mundo existe un sinnúmero de agricultores que utilizan bombas de espalda como instrumento de cuidado y fumigación en las plantaciones de frutilla, esto a corto o largo plazo desemboca en enfermedades por posible envenenamiento debido a la toxicidad y el contacto con productos fitosanitarios.

El glifosato es una de las sustancias más utilizadas como herbicida para el tratamiento de cultivos, el cual ha sido puesto en la lista de sustancias que pueden generar cáncer en los seres humanos según la Agencia Internacional para la Investigación sobre el Cáncer. [1]

Es así como para el año 2012 se presentaron 2126 casos de intoxicaciones por pesticidas en el Ecuador, siendo Manabí la provincia con mayor índice de intoxicación con 306 casos, conjuntamente en el cantón Quero de la provincia de Tungurahua se registró que un 32% de agricultores muestran síntomas como el dolor de cabeza luego de realizar el proceso de fumigación a través del uso de bombas de espalda en sus cultivos. [3] [4]

De la misma manera la calidad de pulverización que ofrece las bombas de espalda no es la apropiada, puesto que de acuerdo al manual de manejo agronómico de la frutilla escrito por Carmen Morales, menciona que la aspersión debe cubrir solo el 50% del follaje de las plantaciones a razón de 40 gotas por cm^2, además se debe

utilizar boquillas de abanico plano con aspersión uniforme debido que son las apropiadas para la fumigación de bandas o camellones; de esta manera se logrará una eficiencia mayor en los cultivos de frutilla [19]

Es por lo que se desarrolla un sistema autónomo de pulverización para fumigación de plantaciones de frutilla asistido por un drone, el cual liberará al operador del contacto directo con el producto fitosanitario al dar instrucciones de mando al vehículo aéreo no tripulado a través de la aplicación de misión punto de ruta creada en un dispositivo Android para establecer parámetros como altura y velocidad de vuelo; conjuntamente el sistema de pulverización rociará el material fitosanitario generando un cubrimiento de las plantaciones de un 50% aproximadamente.

4.2 Requerimientos del sistema

El sistema autónomo de pulverización para la fumigación de plantaciones de frutilla asistido por un drone se desarrollará con enfoque exclusivo a estos frutos y tiene como finalidad emanar partículas de producto fitosanitario dependiendo de la enfermedad a tratar, a través de la pulverización, a una altura de rango entre dos a tres metros dependiendo de la altura de las hileras; en donde estas partículas no se vean afectadas por el viento presente que genera el drone y se tenga un cubrimiento adecuado en el follaje de las plantaciones; todo esto sin pasar por alto un levantamiento máximo de 350 gr incluido el tren de aterrizaje, los tanques contenedores de producto fumigador, la batería de polímero de litio para accionamiento de la bomba de agua DC y todo el sistema de pulverización en sí. Lo que se traduce a un levantamiento máximo de 200 cm^3, es decir 100 cm^3 en cada tanque de abastecimiento, para de esta manera no afectar al centro de gravedad del UAV.

Para que este prototipo funcione de manera adecuada se menciona las siguientes características a seguir

- Comunicación del drone por medio de WiFi (5.8 GHz) proporcionado por el control de mando de éste, generando un puente de comunicación entre la interfaz y el dispositivo volador.
- Parametrización de la aeronave a través de la interfaz y la aplicación de la misión de punto de ruta desarrollada en Android.
- Accionamiento del sistema de pulverización e inicio de la misión a través del dispositivo móvil para el vuelo del drone y la aspersión autónoma del producto fitosanitario seleccionado de acuerdo a la patología a tratar.

La figura 9 muestra el diagrama de bloques del sistema

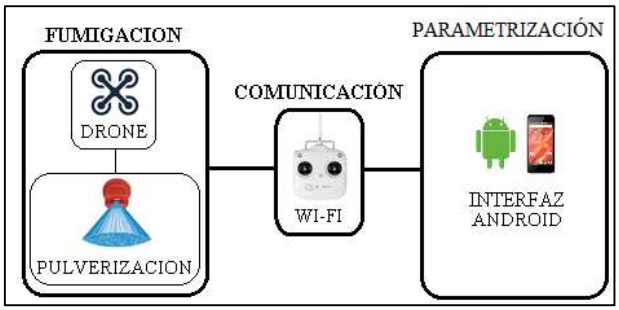


Fig. 9: Diagrama de bloques del sistema autónomo de pulverización para fumigación de plantaciones de frutilla asistido por un drone, desarrollado por el investigador.

4.3 Desarrollo

Se procede a la explicación de las actividades realizadas para el desarrollo de este proyecto, respetando cada uno de los bloques y sub-bloques presentados en la figura 9.

4.3.1 Fumigación

El bloque de fumigación es la primera fase del desarrollo de este proyecto y se divide en dos sub-bloques en donde el primero hace hincapié en la selección del drone para este prototipo y el segundo en el procedimiento de la construcción del sistema de pulverización.

A) Drone

Características Técnicas del Drone

El análisis de las características técnicas del drone para el desarrollo de este proyecto, se calcula teniendo en cuenta el peso total del drone y el peso externo que demandará el sistema de pulverización, para ello se pone a consideración la capacidad de carga útil que reuniría el tren de aterrizaje, los tanques de fluido, la batería, los conductos y los pulverizadores.

En la tabla 3 se hace el análisis técnico comparativo de las características de algunos de los drones, que pueden satisfacer el desarrollo de este prototipo.

Tabla 3. – Análisis Técnico comparativo de las características de los drones [28] [29] [30] [31]

ANALISIS TECNICO COMPARATIVO DE LAS CARACTERISTICAS DE LOS DRONES			
DRONES / CARACTERÍSTICAS	**DJI Phantom 3 estándar**	**Mavic 2 Pro**	**Parrot Bebop 2**
	AERONAVE		
ASPECTO			
PESO	1216 g	907 g	525 g
VELOCIDAD MAXIMA	16 m/s modo ATTI	7 m/s	16 m/s
VELOCIDAD MAXIMA DE ASCENSO	5 m/s	4 m/s	6 m/s
DISTANCIA MAXIMA DE VUELO	750 m con línea de vista	1000 m sin corrientes de viento y con línea de vista	300 m con línea de vista
ALTURA MAXIMA DE VUELO	120 m con linea de vista	110 m con línea de vista	150 m con línea de vista
TIEMPO DE VUELO	25 minutos	22 minutos	25 minutos
GPS	SI	SI	SI
	BATERIA DE VUELO		
CAPACIDAD	4480 mAh	3850 mAh	2700 mAh
VOLTAJE	15,2 v	15,4 v	13,05 v
TIPO	LiPo 4S	LiPo 4S	Li-ion
ENERGIA	68 wh	60 wh	45 wh
	CONTROL REMOTO		
FRECUENCIA	5,8 GHz	2,4 - 5,8 GHz	2,4 GHz
CORRIENTE	2600 mAh LiPo	1800 mAh	1500 mAh
VOLTAJE	3,7 v	3,8 v	5 v

Si bien los drones previstos en la tabla 3 no son drones industriales, se selecciona el DJI Phantom 3 Estándar, puesto que, la energía emitida por la batería de vuelo dará más potencia a los motores y por ende la facilidad de levantar más peso.

Según A. García, en la prueba de levantamiento de peso adicional realizada con un DJI Phantom 3 Estándar, éste drone es capaz de levantar un peso externo máximo de 350 gr; sin embargo, esto se traduce a desgaste más rápido de batería y por ende menos tiempo de autonomía de vuelo. [32]

Conjuntamente a esto luego de analizar la tabla 3, se selecciona el drone DJI Phantom 3 Estándar debido a las siguientes razones adicionales:

- Mayor integridad y robustez.
- Mayor distancia máxima de vuelo.
- Mayor tiempo en autonomía de vuelo debido a la capacidad de corriente en la batería de vuelo.
- Mayor capacidad de voltaje en la batería de vuelo inteligente.
- Frecuencia de comunicación Wi-Fi de 5.8 GHz, menor interferencia de señales.
- Compatibilidad de programación en Android Studio.

DJI Phantom 3 Standar.

DJI es una empresa china, líder en tecnología de vehículos aéreos no tripulados, para fotografía aérea y videografía, conquistando así campos como la robótica, las energías limpias y las telecomunicaciones.

A continuación, en la tabla 4 se muestra los detalles a fondo de cada una de las especificaciones del drone DJI Phantom 3 Standar.

Tabla 4. – Especificaciones del Drone DJI Phantom 3 Standar. [28]

ESPECIFICACIONES PHANTOM 3 ESTÁNDAR			
AERONAVE		**CONTROL REMOTO**	
Peso (Batería y hélices incluidas)	1216 gr	Frecuencia de operación	5.725 - 5.825 GHz
Tamaño diagonal (hélices excluidad	350 mm	Distancia Máxima	LOS: 750 m NLOS: 500 m
Velocidad máxima de ascenso	5 m/s	Rango de temperatura de funcionamiento	0 ° a 40 ° C
Velocidad máxima de descenso	3 m/s	Batería	v2600 mAh LiPo 18650
Máxima velocidad	16 m / s (modo ATTI)	Potencia del transmisor	LOS: 19 dBm NLOS: 14 dBm
Ángulo de inclinación máxima	35 °	Corriente/voltaje de funcionamiento	v600 mA@3.7V
Velocidad angular máxima	150 ° / s	**BATERIA DE VUELO**	
Techo de uso máximo sobre el nivel del mar	6000 m	Capacidad	4480 mAh
Tiempo máximo de vuelo	Aprox. 25 minutos	Voltaje	15,2 V
Rango de temperatura de funcionamiento	0 ° a 40 ° C	Tipo de Batería	LiPo 4S
Sistemas de posicionamiento por satélite	GPS	Energía	68 wh
Rango de precisión de la libración	Vertical: ± 0.5 m Horizontal: ± 1.5 m	Peso Neto	365 gr
		Temperatura de carga	5 ° a 40 ° C

El Phantom 3 cuenta con un asistente automático de vuelo, el GPS incorporado registra el punto de despegue del vehículo aéreo no tripulado y lo recuerda mientras vuela. Además, si alguna vez se pierde la señal de control, el drone regresa al usuario de manera instantánea. [33]

Esta versión de DJI cuenta con las opciones ya descritas, mismas que para aprovecharlas se necesita de un operador que las controle, si bien este proyecto es de carácter autónomo haremos hincapié en misiones de este sentido.

DJI tiene varias funciones que hacen uso de la programación del SDK móvil, el cual está disponible tanto para Android como iOS, cuyas funciones son de carácter autónomo y se describen a continuación:

- Aplicación de gestión de medios
- Aplicación MapView y Waypoint en GaodeMap
- Aplicación MapView y Waypoint en GoogleMap
- Aplicación TapFly y ActiveTrack
- Aplicación del sistema GEO

SDK móvil de DJI

El DJI Mobile SDK es un kit de desarrollo de software diseñado para que los usuarios tengan acceso a la capacidad de los productos de cámara de mano y de avión de DJI. El SDK simplifica el proceso de desarrollo de la aplicación al cuidar la funcionalidad de nivel inferior, como la estabilización de vuelo, la gestión de la batería, la transmisión de señales y la comunicación.

El objetivo del SDK Móvil es proporcionar a cualquier desarrollador con experiencia en iOS o Android el conocimiento y la comprensión necesarios para crear aplicaciones que cambian las perspectivas utilizando la tecnología de DJI. [34]

El SDK incluye:

- Una biblioteca / marco que se puede importar a una aplicación de Android o iOS que le da acceso al producto DJI
- Un simulador de aviones y una herramienta de visualización.
- Depurador y registrador remoto para iOS
- Código de muestra y tutoriales

Aplicación MapView y Waypoint

Esta aplicación hace uso del SDK móvil de DJI para de esta manera crear una vista de mapa simple, modificar las anotaciones de vista de mapa, y mostrar la aeronave en dicho mapa de Google Maps usando datos del GPS, además esta aplicación usa la vista del mapa para mostrar cada uno de los puntos de ruta y mostrar la ruta de vuelo de la aeronave en tiempo real cuando se está ejecutando la misión; conjuntamente dicha misión permite configurar los ajustes de punto de ruta, creando y modificando dichos puntos, para que el vehículo aéreo no tripulado se movilice hasta ellos y a su vez el sistema de pulverización pueda realizar el trabajo de fumigación. [35]

Además, se describe a continuación el controlador de vuelo y los sensores que actúan dentro de la aeronave DJI Phantom 3 Estándar.

Controlador de vuelo

El controlador de vuelo es una computadora a bordo que combina la información de control del piloto con la información del sensor para ajustar el empuje en cada hélice y volar la aeronave como se desee. [36]

El controlador de vuelo es responsable de:

- Control de vuelo incluyendo control de motor, despegue y aterrizaje, modos de vuelo manual y misiones.
- Información del estado del avión como actitud, altitud, velocidad.

- Subcomponentes del sensor, como brújulas, IMU y sistemas de posicionamiento.
- Sub componentes del avión como el tren de aterrizaje.

A continuación, se muestra una introducción a los sensores presentes en la aeronave.

Brújula

La brújula mide la dirección del campo magnético y se utiliza para determinar el rumbo de la aeronave con respecto al Norte. [36]

Barómetros.

El barómetro mide los cambios en la presión del aire para determinar los cambios en la altitud, de esta manera los cambios en la presión del aire son lineales con los cambios en la altitud. [37]

Posicionamiento.

La aeronave viene con un sistema de posicionamiento satelital para el consumidor, el cual utiliza constelaciones de satélites GPS y GLONASS [36]

IMU (Unidad de Medición Inercial)

La unidad de medición inercial es la combinación de un acelerómetro y un giroscopio que se encarga de medir los movimientos del UAV, estos movimientos dependen de la ubicación y orientación de los ejes que forman un sistema de coordenadas o marco referencial. [38]

El SDK móvil de DJI utiliza los sistemas de coordenadas de cuerpo y mundo.

Sistema de coordenadas de cuerpo

El sistema de coordenadas de cuerpo hace mención al propio avión en donde establecen tres ejes perpendiculares, instaurando el origen de los ejes en el centro de masa.

El eje X se dirige a través de la parte delantera de la aeronave, el eje Y a la derecha y el eje Z está a través de la parte inferior del UAV, como se muestra en la figura 10.

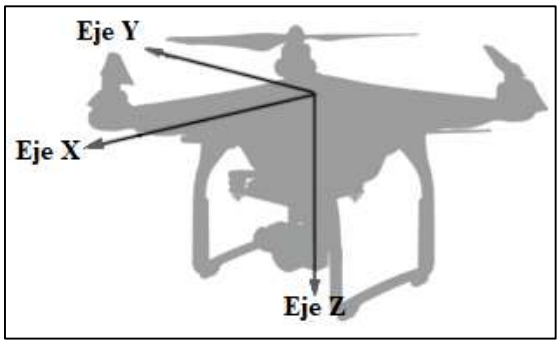


Fig. 10: Sistema de coordenadas de cuerpo. [38]

La rotación de la aeronave también se describe con estos mismos ejes utilizando la regla de coordenadas de la mano derecha para definir la dirección de rotación positiva. Al describir el movimiento de rotación, los ejes X, Y y Z pasan a llamarse Roll, Pitch and Yaw respectivamente, véase la figura 11. Estos movimientos son consistentes con el sistema de coordenadas de cuerpo del avión.

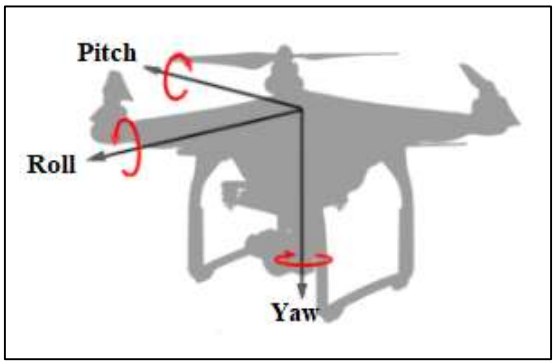


Fig. 11: Ejes de rotación roll, pitch y yaw. [38]

Sistema de coordenadas de mundo.

El sistema de coordenadas de mundo o coordenadas terrestres alinea los ejes positivo X, Y y Z con las direcciones de Norte, Este y Abajo, como se muestra en la figura 12; esta convención se llama North-East-Down o NED.

El origen de un sistema de coordenadas NED suele ser un punto en el marco mundial como la ubicación de despegue.

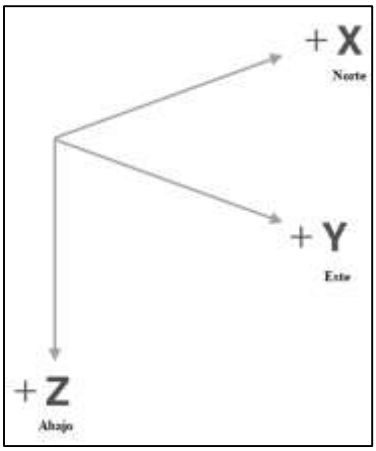


Fig. 12: Sistema de coordenadas de mundo. [38]

Actitud de vuelo

La actitud de vuelo se define por la rotación alrededor de los ejes de roll, pitch y yaw (balanceo, inclinación y giro) en el sistema de coordenadas de cuerpo.

Roll - Balanceo

En la figura 13, se muestra el UAV desde atrás, para entender el movimiento de roll, el cual mide la rotación de la aeronave sobre el eje longitudinal (X, roll). Al manipular este movimiento, la aeronave se moverá hacia la izquierda o hacia la derecha.

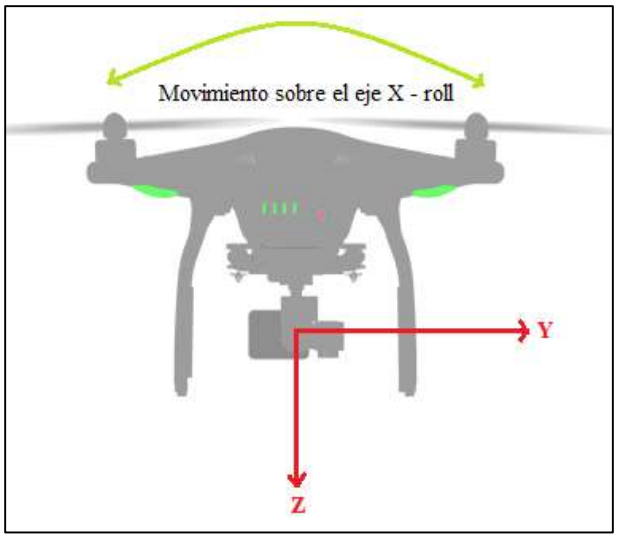


Fig. 13: Movimiento sobre el eje longitudinal X - roll [38]

Pitch - Inclinación

La figura 14 muestra la aeronave desde su costado para entender el movimiento de pitch, el cual mide la rotación del UAV alrededor del eje lateral (Y, Pitch). El pitch moverá la aeronave hacia adelante o hacia atrás.

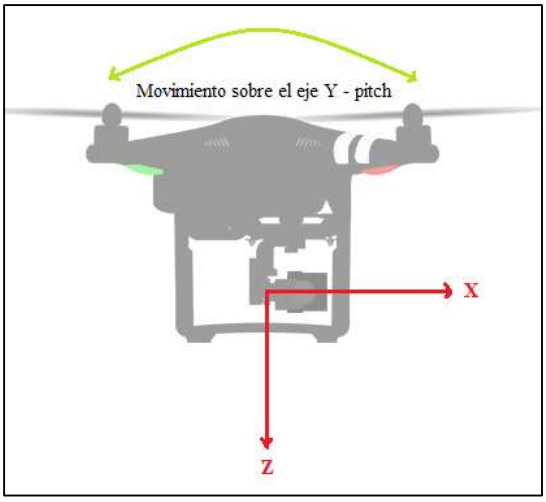


Fig. 14: Movimiento sobre el eje longitudinal Y - pitch [38]

Yaw – Giro.

En la figura 15 se puede observar el avión desde arriba. El movimiento de Yaw mide la rotación de la aeronave alrededor del eje vertical (Z, Yaw). El movimiento de giro mueve la aeronave sobre su eje cambiando el rumbo de vuelo.

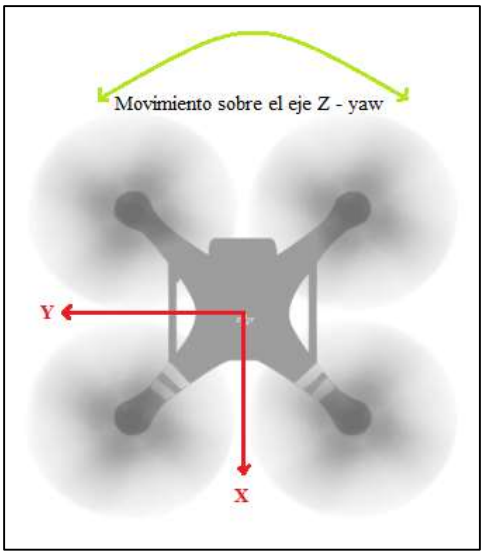


Fig. 15: Movimiento sobre el eje longitudinal Z - yaw [38]

Throttle - Acelerador

El movimiento de throttle realiza el ascenso o descenso del cuadricóptero en el eje z verticalmente, como se muestra en la figura 16. Este movimiento controla el empuje promedio de la aeronave desde su sistema de propulsión

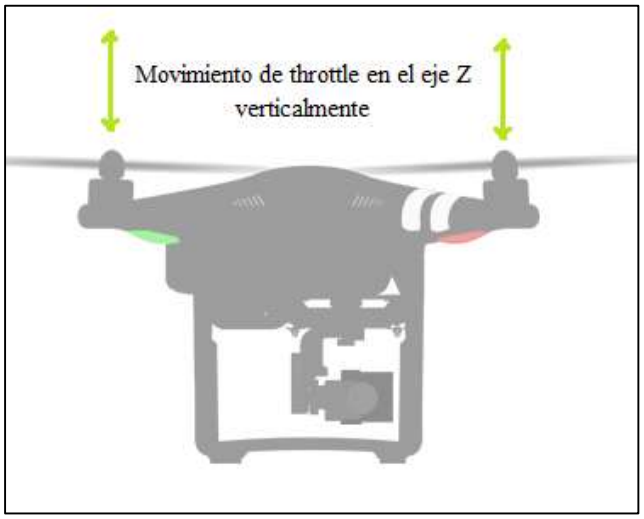


Fig. 16: Movimiento de throttle en el eje Z verticalmente. [38]

B) Sistema de pulverización

Un sistema de pulverización necesita de dispositivos como el tanque almacenador de líquido, un motor o bomba de agua, sus conductos y por supuesto las boquillas que darán la forma de barrido para la cobertura del cultivo.

Por consiguiente, se detalla a continuación cada uno de los materiales elegidos para el diseño y acoplamiento de este sub-bloque en la etapa de la fumigación:

Bomba de presión de agua

En el mercado existen varios tipos de bombas o motores que trabajan con excitación de corriente directa, a continuación en la tabla 5 se muestra la comparación de algunas de estas para definir la apropiada que será acoplada en el vehículo aéreo no tripulado.

Tabla 5. – Comparación de bombas de agua de alimentación DC. [39] [40] [41]

BOMBAS DE AGUA DE ALIMENTACION DC			
Bombas / Caracteristicas	Bomba de Diafragma R385-PLUS	Micro-Bomba Anself 12VDC	RS-360SH
ASPECTO			
DIMENSIONES (mm)	90 x 40 x 35	52 x 46 x 55	42 x 45 x 65
PESO	98 gr	65 gr	106 gr
CAUDAL DE SALIDA	1,5 - 2 lt/min	1,5 - 2 lt/min	1 - 2 lt/min
VOLTAJE DE OPERACIÓN	6 - 12 volts	6 - 12 volts	3 - 9 volts
CORRIENTE	0,25 - 0,75 Ampers	≤ 0,25 Ampers	≤ 0,2 Ampers
VOLTAJE NOMINAL	7,2 volts	9 Volts	7,2 volts

Si bien las dimensiones y el peso de la Micro-bomba Anself 12 VDC son menores en comparación a las demás; la corriente que esta bomba ocupa es limitado y el stock de este producto en tiendas es escaso.

Por lo que, se toma en consideración la bomba de diafragma R385-PLUS para el acople del sistema de pulverización, debido a que el voltaje nominal de trabajo que demanda es ideal para el uso de una batería de polímero de litio en su alimentación, la cual se describe más adelante, además es de fácil adquisición y bajo costo.

Conjuntamente la bomba de diafragma R385-PLUS ofrece un caudal de suficiente potencia para la pulverización de material fitosanitario en el follaje de las plantaciones de frutilla.

Tanque almacenador de líquido.

Los tanques almacenadores son elementos que están constituidos generalmente de plástico y existen en un sinnúmero de capacidades y formas.

Cabe mencionar que independientemente de la densidad, viscosidad y capacidad el tanque será diferente en cuanto a su espesor y calidad.

Ya que los plaguicidas posibles a ocupar no son de gran densidad se ocupa tanques almacenadores de agua común de plástico, adaptado al vehículo aéreo no tripulado, conjuntamente con la bomba y sus conductos hacia la/s boquilla/s para la pulverización del material.

Este proyecto contará con la implementación de dos tanques de abastecimiento con capacidad de 250 cm^3, figura 17, para de esta manera no afectar el centro de masas del drone y la misión se pueda realizar a cabalidad; además, es de suma importancia explicar que, cada tanque almacenador se llenará con 100 cm^3 (capacidad máxima) debido a la capacidad limitada de levantamiento del drone y el desarrollo de esta investigación con carácter de un prototipo.

Fig. 17: Tanque almacenador de capacidad de 250 cm^3.

Batería de Polímero de litio.

Las baterías de polímero de litio existen de varios modelos, capacidad, peso y dimensiones, sin embargo, debido a que la alimentación de la bomba de agua está en el rango de 6 a 12 volts y su voltaje de trabajo nominal es de 7.2 Volts se selecciona una batería de doble célula recargable, es decir de 7.4 volts y 0.3 amperes, véase la figura 18.

Fig.18: Batería de polímero de litio seleccionada.

Conductos de agua

Los conductos de agua por el cual se transportará el producto fitosanitario desde el taque de almacenamiento hasta la/s boquilla/s será mediante mangueras de goma de forma cilíndrica con ancho de 5 mm de diámetro, similares a las que son utilizadas en medicación de enfermos en casas de salud, figura 19.

Fig. 19: Conductos transportadores de material fitosanitario.

Boquillas pulverizadoras.

Las boquillas pulverizadoras de material fitosanitario escogidas para el tratamiento de frutillas se realizó mediante el estudio de la tabla 6.

Tabla 6. – Tipos de boquillas de pulverización. [22] [42]

TIPOS DE BOQUILLAS DE PULVERIZACIóN		
BOQUILLA	PATRóN DE ASPERSIÓN	CARACTERíSTICAS
Abanico		Pulverización con concentración hacia el centro en relación con su contorno.
Cono		Gotas Finas; completa cobertura de la plantación; barrido circular.
Espejo		Gotas gruesas: necesitan de un gran flujo de líquido para que su aspersión sea la adecuada.
Aspersión uniforme		Ideal para pulverización en bandas o camellones; provocan una aspersión uniforme sin traslape.

Luego de analizar las diferentes boquillas de pulverización disponibles para la aspersión de material fitosanitario, se selecciona la de aspersión uniforme debido a que esta es óptima en aplicaciones en bandas donde se cultivan las plantaciones de frutilla, véase la figura 20, conjuntamente se escogió este tipo de boquillas debido al barrido que emanan para así cumplir con la correcta calidad de pulverización en frutilla según el Manual de manejo agronómico de la frutilla escrito por C. Morales [19].

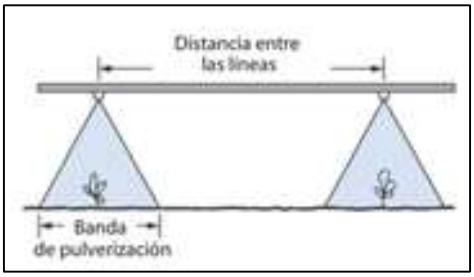


Fig. 20: Aplicación de pulverización en bandas [42]

Acoplamiento del sistema de pulverización al UAV

El acoplamiento del sistema de pulverización dependerá del análisis del correcto montaje de los elementos que conforman dicho sistema (bomba de agua DC, tanques, boquillas, conductos), para de esta manera no afectar el centro de gravedad del drone y poder tener una buena estabilidad en el momento del recorrido de forma autónoma por los puntos de ruta.

Es de mucha importancia mencionar que para que el drone no sufra inestabilidades se debe hacer coincidir el centro de empuje con el centro de gravedad. [43]
El centro de empuje es el punto en donde el empuje del drone es mayor y este depende exclusivamente de la posición de los motores; se calcula uniendo los ejes de los motores de manera diagonal, mírese la figura 21.

Fig. 21: Centro de empuje del UAV, desarrollado por el investigador.

Sin embargo, el centro de gravedad es aquel punto geométrico en donde el drone quedaría en equilibrio y este depende de la distribución de pesos en el UAV. En Física, el centro de masas puntuales se calcula de la siguiente manera, donde:

$$X_{CM} = \frac{\sum x_i\, m_i}{\sum m_i} \qquad\qquad Y_{CM} = \frac{\sum y_i\, m_i}{\sum m_i}$$

$$Ecuación\ (6)$$

Donde:

X_{CM}, Y_{CM}: Puntos de centro de masa en el plano.

x_i: Unidades de posicionamiento de cada masa i-ésima en el eje x, respecto al sistema de referencia supuesto.

y_i: Unidades de posicionamiento de cada masa i-ésima en el eje y, respecto al sistema de referencia supuesto.

m_i: Masa de cada elemento i-ésima, presente el en plano. [44]

De esta manera se realiza el análisis de la implementación de los elementos que conforman el sistema de pulverización en el UAV, véase la figura 22, haciendo referencia como punto de origen el centro de empuje, debido a que es donde debe coincidir el centro de gravedad para que la aeronave no sufra algún tipo de inestabilidad.

Los pesos de los tanques mostrados en la figura 22, se dan cuando se tiene 100 ml de líquido fitosanitario en cada tanque.

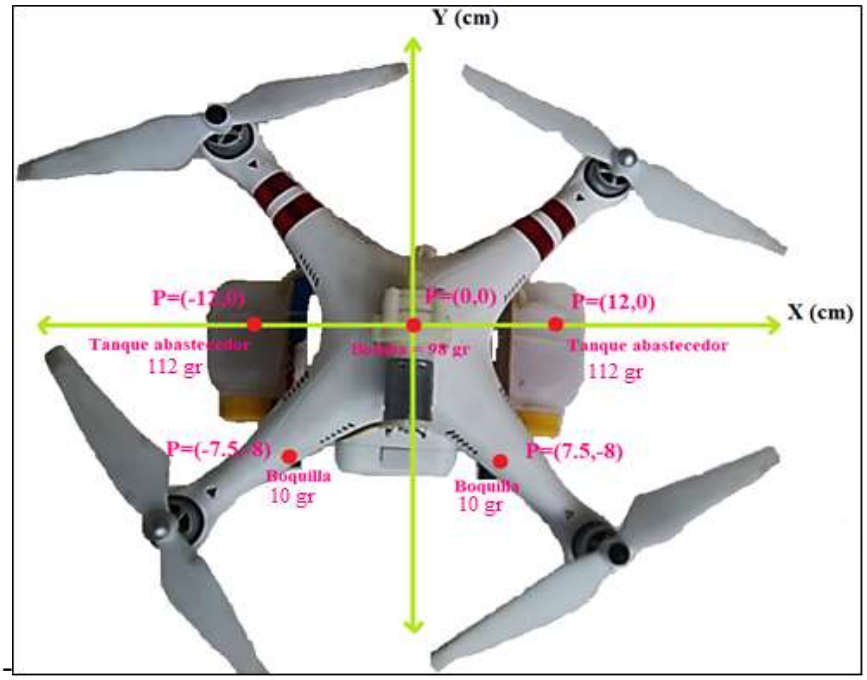


Fig.22: Análisis del centro de gravedad del UAV, desarrollado por el investigador basado en [44]

De esta manera, mediante el uso de la ecuación 6 se obtiene el centro de masas de acuerdo a los cálculos siguientes:

Cálculo de centro de masa en el eje "X"

$$X_{CM} = \frac{\sum x_i\, m_i}{\sum m_i}$$

$$X_{CM} = \frac{(-12\ cm * 112\ gr) + (-7.5\ cm * 10\ gr) + (0\ cm * 98gr) + (12\ cm * 112gr) + (7.5\ cm * 10\ gr)}{112gr + 10gr + 98\ gr + 10\ gr + 112\ gr}$$

$$X_{CM} = \frac{0\ cm\ gr}{342\ gr}$$

$$X_{CM} = 0\ cm$$

Cálculo de centro de masa en el eje “Y”

$$Y_{CM} = \frac{\sum y_i \, m_i}{\sum m_i}$$

$$X_{CM} = \frac{(0\ cm * 112\ gr) + (-8\ cm * 10\ gr) + (0\ cm * 98\ gr) + (-8\ cm * 10\ gr) + (0\ cm * 112\ gr)}{112\ gr + 10gr + 98\ gr + 10\ gr + 112\ gr}$$

$$Y_{CM} = \frac{(-80\ cm\ gr) + (-80\ cm\ gr)}{342\ gr}$$

$$Y_{CM} = \frac{-160\ cm\ gr}{342\ gr}$$

$$Y_{CM} = -0.4678\ cm$$

Centro de gravedad P = (0, - 0.46) cm

Si se ubica el punto de centro de gravedad en la aeronave, este tiene una leve variación de cuatro milímetros en Y^- en relación al centro de empuje, mírese la figura 23; el comportamiento de la altura, velocidad y modo de vuelo se de acuerdo a esta variación se lo realiza en el punto 4.4 de este documento.

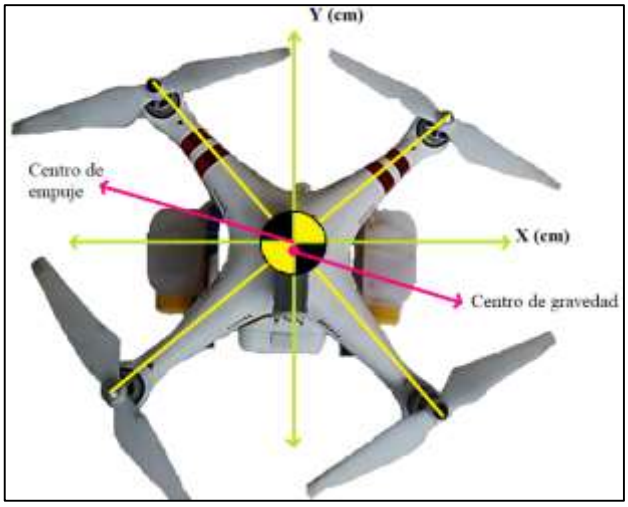


Fig. 23*:* Coincidencia del centro de empuje y el centro de gravedad, desarrollado por el investigador.

Es así como entonces se optó por acoplar dos maderas de 3 mm de espesor y 8 gr de peso, a cada costado del tren de aterrizaje, dos tanques contenedores de líquido con capacidad de 100 ml con un peso de 7 gr cada uno y dos boquillas pulverizadores en la parte posterior del tren de aterrizaje, separadas en 15 centímetros y con un peso de 8 gr cada una.

Además, la bomba de agua se acopló en el techo del drone, liberando así de cualquier interferencia electromagnética que pueda causar al sensor de brújula cuando su motor es accionado; de esta manera la Unidad de Medición Inercial no se ve afectada y la pulverización de material se puede realizar sin problemas a través de los puntos de ruta establecidos.

Todo el sistema de pulverización reúne un peso aproximado de 342 gr cuando los tanques se encuentran con 100 ml de líquido fumigador, y de 144 gr cuando están vacíos; la figura 24 muestra el sistema de pulverización acoplado en el vehículo aéreo no tripulado.

Fig. 24: Sistema de pulverización acoplado en el vehículo aéreo no tripulado.

4.3.2 Comunicación

La comunicación, siendo esta la segunda faceta, entre el dispositivo móvil y el vehículo aéreo no tripulado se realiza de forma transparente, puesto que al momento de encender UAV y el control remoto establecen un enlace de

comunicación Wifi a una frecuencia de 5.8 GHz y un alcance máximo de 750 metros con línea de vista entre estos dos dispositivos.

De la misma manera el dispositivo móvil se conecta al control de la aeronave de forma inalámbrica a través de Wifi a 2.4 GHz y un alcance máximo de 10 metros, de esta manera se cuenta con dos enlaces inalámbricos que conectan el controlador remoto, la aeronave y el dispositivo móvil.

En la figura 25 se da una ilustración de la conexión que se establece entre el dispositivo móvil, el control remoto y la aeronave.

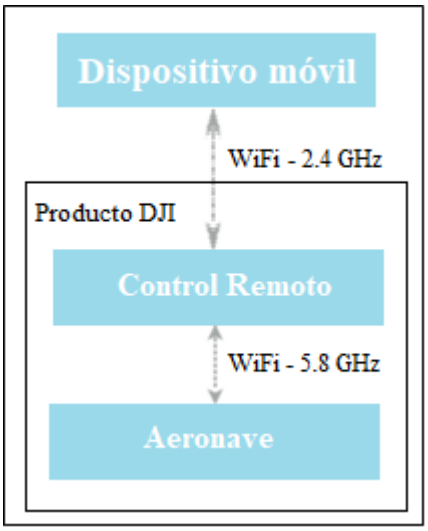


Fig.25: Conexión del dispositivo móvil, el control y la aeronave, desarrollado por el investigador basado en [33]

4.3.3 Parametrización de la misión de punto de ruta.

La parametrización es la tercer y más importante faceta en donde se hace mención de la construcción de una aplicación de punto de ruta en un dispositivo móvil de sistema operativo Android, para que de esta manera se pueda parametrizar datos relacionados al recorrido de puntos como:

- Altura de vuelo.
- Velocidad de vuelo.
- Modo de vuelo.
- Acción después de finalizar la misión.

Se detalla a continuación los factores que intervienen para el desarrollo de este último bloque de parametrización.

Descarga e instalación de Android Studio

Android Studio facilita herramientas más vertiginosas para la instauración de proyectos en todas las clases de equipos que cuenten con sistema operativo Android; en la tabla 7 se puede observar la disponibilidad de Android Studio para cada uno de los sistemas operativos.

Tabla 7. – Disponibilidad de Android Studio para sistemas operativos. [45]

Plataforma	Paquete de Android Studio	Tamaño
Windows (64 bits)	android-studio-ide-181.5014246-windows.exe Recomendado	923 MB
Windows (32 bits)	android-studio-ide-181.5014246-windows32.zip Sin instalador .exe	999 MB
MAC	android-studio-ide-181.5014246-mac.dmg	988 MB
Linux	vandroid-studio-ide-181.5014246-linux.zip	1006 MB

De la misma manera cada uno de los sistemas operativos hace mención de los requerimientos del sistema los cuales de detallan a continuación en la tabla 8.

Tabla 8. – Requerimientos de Sistema de Android Studio para sistemas operativos. [45]

<table>
<tr><th colspan="3">REQUERIMIENTOS DEL SISTEMA</th></tr>
<tr><th>WINDOWS</th><th>MAC</th><th>LINUX</th></tr>
<tr><td rowspan="2">Microsoft® Windows® 7/8/10 (32 bits ó 64 bits)</td><td rowspan="2">Mac® OS X® 10.10 (Yosemite) o superior, hasta 10.13 (macOS High Sierra)</td><td>GNOME o KDE de escritorio
Probado en Ubuntu® 14.04 LTS, Trusty Tahr</td></tr>
<tr><td rowspan="2">Distribución de 64 bits capaz de ejecutar aplicaciones de 32 bits</td></tr>
<tr><td rowspan="2">3 GB de memoria RAM como mínimo (se recomiendan 8 GB), más 1 GB para utilizar el emulador.</td><td rowspan="2">Mínimo 3 GB de memoria RAM (se recomienda 8 GB), más 1 GB para el emulador del sistema.</td></tr>
<tr><td>GNU C Library (glibc) 2.19 o versiones posteriores</td></tr>
<tr><td rowspan="2">Mínimo 2 GB de espacio en disco utilizable (se recomiendan 4 GB); 500 MB para el IDE + 1.5 GB para Android SDK y la imagen de sistema.</td><td rowspan="2">2 GB de espacio en disco disponible como mínimo (se recomiendan 4 GB); 500 MB para el IDE + 1.5 GB para Android SDK y la imagen de sistema.</td><td>3 GB de memoria RAM como mínimo (se recomiendan 8 GB), más 1 GB para el emulador de Android.</td></tr>
<tr><td rowspan="2">2 GB de espacio en disco disponible como mínimo (se recomiendan 4 GB); 500 MB para el IDE + 1.5 GB para Android SDK y la imagen de sistema.</td></tr>
<tr><td>Resolución de pantalla mínima de 1280 x 800</td><td>Resolución de pantalla mínima de 1280 x 800</td></tr>
<tr><td>Resolución de pantalla mínima de 1280 x 800</td></tr>
</table>

En la comparación de la tabla 8 de los requerimientos de cada sistema operativo se puede observar que, el tamaño de memoria RAM necesaria, el espacio en disco y la resolución de pantalla; demandan de exactamente lo mismo, sin embargo, para este proyecto se hace uso de Windows 8.1 (64 bits), puesto que en Ecuador el sistema operativo Windows es distribuido por marcas internacionales como: HP, Dell, Lenovo, Sony y Samsung, según la G. Durán gerente general de Microsoft en Ecuador. [46]

Para la Instalación de Android Studio, se siguen los siguientes pasos:

Ejecutar el archivo .exe, véase la figura 26, previamente descargado de la página oficial de Android Studio, sección descargas. [47]

Fig. 26: Instalación de Android Studio

Se procede a seguir con las indicaciones del asistente de configuración para instalar Android Studio y las herramientas de SDK necesarias.

En algunos sistemas de Windows, la secuencia de comandos de inicio no encuentra el destino de instalación del JDK. Si se presenta este problema, se tendrá que configurar una variable de entorno que indique la ubicación correcta.

Una vez Instalado correctamente este software aparecerá la interfaz, mostrada en la figura 27.

Fig. 27: Interfaz inicial de Android Studio.

Antes de empezar con el detalle de los pasos que se siguieron para la programación de la aplicación en Android se presenta el diagrama de flujo característico del comportamiento de la aplicación de puntos de ruta en la figura 28.

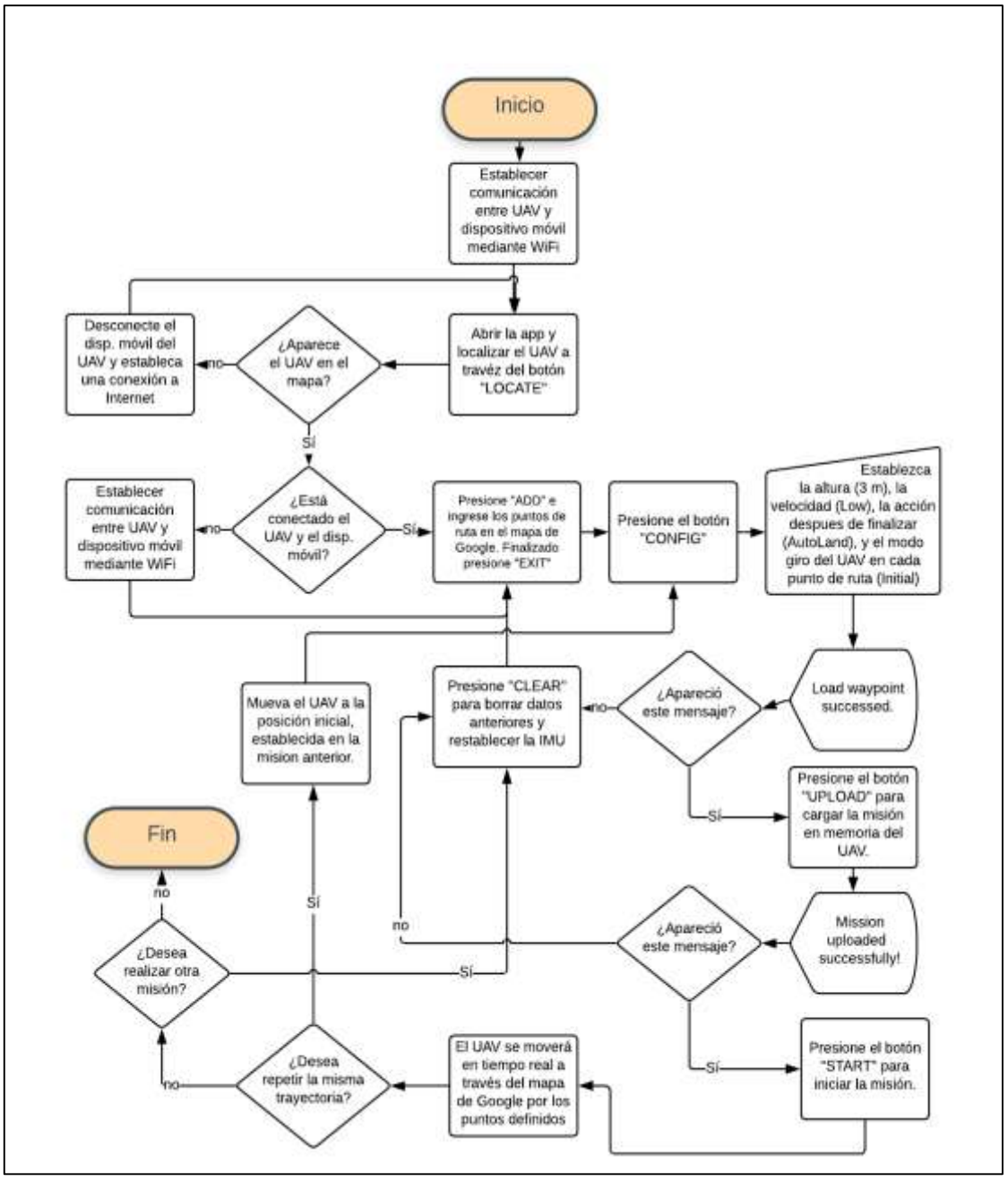


Fig. 28: Diagrama de flujo de la aplicación de punto de ruta, desarrollado por el investigador.

Creación de la aplicación de puntos de ruta.

La creación de la aplicación de puntos de ruta necesita de ciertos requisitos que se mencionan a continuación:

- Una tarjeta de crédito o un número de teléfono para la verificación del registro de desarrollador DJI (no se harán cargos).
- Un dispositivo móvil de sistema operativo Android.
- Nivel de API de Android 19 o superior
- Android Studio 1.5 o superior
- API 19: Android 4.4 (KitKat) para "Teléfono y tableta"
- Dispositivo compatible: En este caso estaremos usando un dispositivo SONY Xperia XA (F3113)
- Registro como desarrollador DJI

Creación del proyecto

En la barra de herramientas de Android Studio se selecciona Archivo -> Nuevo -> Nuevo proyecto para crear un nuevo proyecto, llamado "GSDemo". Se ingresa el dominio de la empresa (nombre de desarrollador si no cuenta con empresa) y el nombre del paquete (en este proyecto se utiliza "com.dji.GSDemo.GoogleMap") que se desea y presione Siguiente.

Luego se selecciona "Actividad vacía" y, por último, se deja el nombre de la actividad como "MainActivity", y el nombre del diseño como "activity_main", y se presiona "Finalizar" para crear el proyecto.

Configuración de los servicios de Google Play

El servicio de Google Play cuenta con un paquete API para dispositivos Android, de esta manera se utilizará la API de Google Maps, debido a que el SDK de DJI para Android trae dos opciones de API para la programación de mapas geográficos; estos son Gaode Maps y Google Maps.

Si bien Gaode Maps y Google Maps son servidores de aplicaciones de mapas; Gaodle Maps es de origen chino y es en este idioma en el cual muestra su interfaz, además, en China es donde el uso de este servidor tiene más incidencia.
En comparación Google Maps es utilizado por aproximadamente la mitad de todos los usuarios activos de Google a nivel mundial, es decir mil millones de dispositivos, según D. Perez. [48]

La función principal del servicio de Google Play es la comprobación de que todas las aplicaciones instaladas estén en la última versión disponible; es decir que, si la versión de Google Maps tiene una actualización reciente, Google Play Service se encargará de descargar e instalar esta última versión de manera automática, siendo esta una razón más para el uso de Google y no Gaode. [49]

Además de todo esto, Google Play Services mejora la experiencia general de uso del terminal. Permite agilizar las búsquedas sin conexión, proporcionando de esta manera mapas más envolventes.

La API de Google Maps está incluida dentro del paquete de servicios de Google Play y se puede instalar el SDK de servicios de Google Play utilizando el Administrador de Android SDK como se muestra en la figura 29 a continuación:

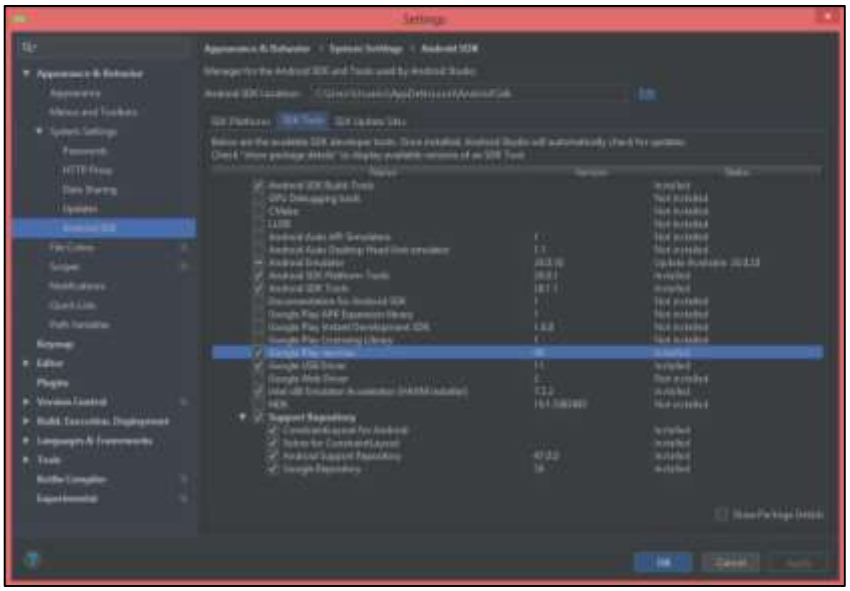


Fig. 29: Instalación de los servicios de Google Play

Una vez instalado el paquete de servicios de Google Play, nos dirigimos al archivo **AndroidManifest.xml** de la aplicación en donde se agrega el siguiente elemento como subelemento de **<aplicación>**

```
<meta-data android:name="com.google.android.gms.version"
  android:value="@integer/google_play_services_version" />
```

De esta manera se puede empezar a utilizar la API de Google Maps en la aplicación.

Generación de una clave API de Google Maps.

La clave API de Android de Google Maps es un algoritmo que utilizan los programas de dispositivos Android para identificar el programa con el que se comunican. La clave API además actúa como identificador único y autenticador secreto, la cual tendrá un conjunto de derechos de acceso asociado. [50]

La API de Google Maps para Android permite que las aplicaciones incluyan Google Maps o Street View sin la necesidad de abrir una aplicación independiente, lo que permite un control total sobre este y además proporciona medios para agregar marcadores personalizados y superposiciones sobre el mapa. [51]

En la página de *console.developers.google.com* en la sección de credenciales, se selecciona “crear credenciales” donde se mostrará una pantalla como se muestra en la figura 30. [52]

Fig. 30: Generación de clave API de Google. [52]

Se procede a crear la clave, seguidamente de darle una restricción y una huella digital del certificado SHA-1 (Opcional); esta restricción especifica los sitios web, las direcciones IP y las aplicaciones que pueden usar esta clave.

Generación de la clave SHA-1

El SHA1 es un método de criptografía que convierte una cadena de texto de información, en otra cadena cifrada de cuarenta caracteres sin importa la longitud de la cadena original, como se muestra en la figura 31; de esta manera se hace más difícil poder obtenerla, ya que SHA1 no tiene método de reversa para obtener la clave original a partir de una ya cifrada. [53]

SHA significa *Secure Hash Algorithm*, Algoritmo de Hash Seguro.

Hash hace referencia a algoritmos que consiguen crear a partir de una entrada (ya sea un texto, una contraseña o un archivo) una salida alfanumérica de longitud normalmente fija que representa un resumen de toda la información que se le ha dado. [54]

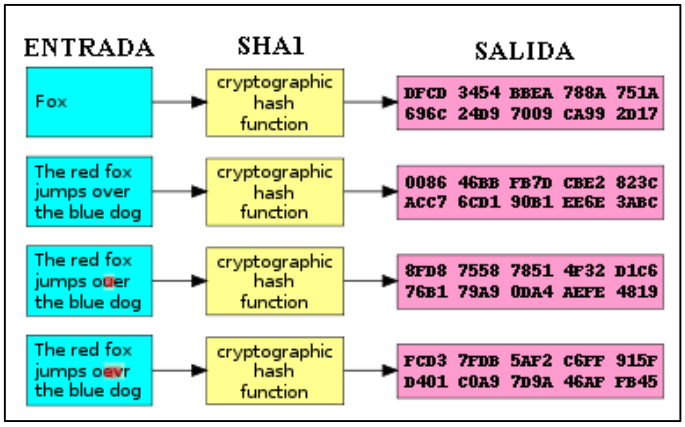


Fig. 31: Método de criptografía SHA-1 [51]

Se puede usar Android Studio para generar una clave SHA-1 fácilmente. Haciendo clic en el botón Gradle en el lado derecho de Android Studio. Luego se selecciona el proyecto y navegue a Task -> Android -> signingReport.

Conjuntamente, se da un doble clic en signignReport y se verifica el área de la Consola de Android Studio en donde se puede encontrar la clave SHA-1; figura 32.

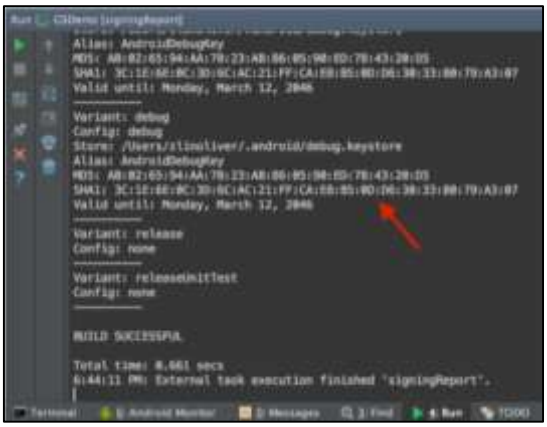


Fig. 32: Generación de clave SHA-1 en Android Studio.

Una vez generada esta clave se copia y agrega en la página de las credenciales de API de Google, figura 33.

El nombre del paquete se obtiene del archivo *AndroidManifest.xml.*

Fig. 33: Restricción de las aplicaciones y clave SHA-1.

Agregar la clave API de Google

En el archivo AndroidManifest.xml, se agrega el siguiente elemento como hijo de “element” y se sustituye la clave API de Google que se acaba de generar en el atributo “value” como se muestra en el siguiente apartado.

```
<meta-data
   android:name="com.google.android.geo.API_KEY"
   android:value="" /> //: Ingrese aqui su clave API de Google.
```

El primer elemento establece la clave "com.google.android.geo.API_KEY" al valor de su clave API de Google. El segundo establece el número de versión de los servicios de Google Play.

Además, se especifica los permisos de las necesidades de la aplicación, agregando elementos <uses-permission> como elementos secundarios del elemento <manifest> en el archivo "AndroidManifest.xml".

```
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.CHANGE_CONFIGURATION" />
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW" />
<uses-permission android:name="android.permission.WRITE_SETTINGS" />
<uses-permission android:name="android.permission.VIBRATE" />
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS" /
>
```

Maps SDK para Android usa OpenGL ES versión 2 para representar el mapa.

OpenGL ES (Open Graphics Library for Embedded Systems) es una variante de la API OpenGL; esta permite producir y manipular gráficos 2D y 3D y está planteada para sistemas robustos, como las videoconsolas o dispositivos móviles. [55]

```
<uses-feature
    android:glEsVersion="0x00020000"
    android:required="true" />
```

La sentencia de código prevista evita que Google Play Store muestre la aplicación en dispositivos que no son compatibles con OpenGL ES versión 2. [56]

Para más detalles de la codificación en el archivo AndroidManifest.xml, véase el anexo A.

Soporte Multidex con Gradle

El servicio de Google Play va de la mano con la agregación del soporte multidex con el fin de evitar el límite de 64K con gradle. Esto significa que la aplicación supera los 65.536 métodos. La especificación de DEX (Dalvik Executable) limita la cantidad total de métodos a los que se puede hacer referencia un archivo, incluidos los métodos del framework de Android, de biblioteca y del propio código. [57]

Para habilitar la función de multidex se debe modificar la configuración de archivos a nivel de módulo para incluir la biblioteca de soporte y de la misma manera dar acceso a la salida de multidex en las partes de configuración predeterminada y dependencias, como se muestra a continuación:

```
android {
        ......
    // Enabling multidex support.
      multiDexEnabled true
}
```

Una vez en este paso se realiza la sincronización del proyecto con el gradle; este es un sistema de compilación que reúne en un solo, las mejores prestaciones de otros sistemas de compilación. Está basado en JVM (Java Virtual Machine), lo que significa que se puede escribir el propio script en java, de esta manera Android Studio lo entenderá y lo usará.

Si esta acción se realiza con éxito, aparecerá una interfaz como se muestra en la figura 34.

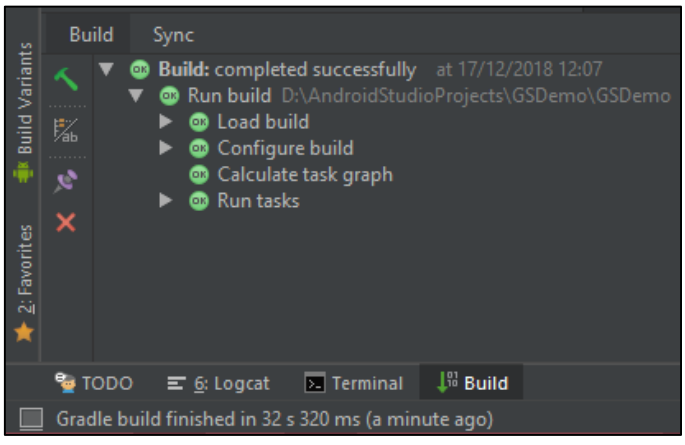


Fig. 34: Sincronización exitosa del proyecto con el gradle.

Importación de la dependencia del Maven

Maven es una herramienta de gestión de proyectos el cual define todo lo que este necesita, siendo las dependencias su punto más fuerte; esta herramienta es capaz de ir a Internet buscar los jar (Java Archive) que el proyecto necesita y bajárselos automáticamente. [58]

Para realizar la importación de la dependencia más reciente de DJI Android SDK Maven, se abre la pestaña Gradle Script -->build.gradle(module:app) y se añade las siguientes líneas de código:

```
dependencies {
 implementation ('com.dji:dji-sdk:4.8')
        compileOnly ('com.dji:dji-sdk-provided:4.8')
}
```

Se puede realizar una doble verificación de la dependencia del Maven, en la barra de herramientas seleccione File-->Project Structure, luego se selecciona el módulo "app" y en la pestaña de dependencias se mostrará las dependencias de DJI instaladas como se muestra en la figura 35.

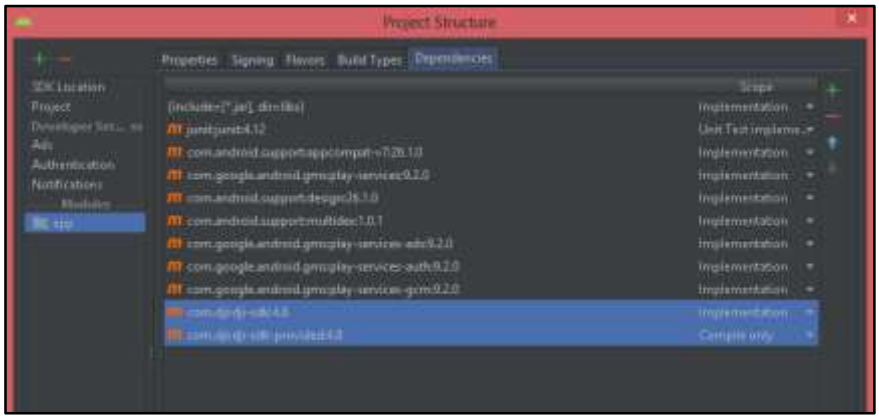


Fig. 35: Importación de la dependencia del Maven.

Construcción de los diseños del MainActivity.

Implementación de diseño de la actividad principal.

El diseño de la actividad principal se lo realiza en el archivo activity_main.xml, donde se tendrá en cuenta la visualización del mapa de Google y los siguientes botones con sus respectivas funciones.

- **(Locate) Localización** del vehículo aéreo no tripulado.
- **(Add) Añadir** los puntos de ruta para el seguimiento de la misión.
- **(Clear) Limpiar** los puntos de la misión de una misión ya establecida.
- **(Config) Configuración** de la misión, como:
    - ✓ Altura de vuelo.
    - ✓ Velocidad de vuelo.
    - ✓ Acción después de finalizar la misión.
    - ✓ Modo de vuelo

- **(Upload) Cargar** la misión a memoria.
- **(Start) Comienzo** de la misión.
- **(Stop) Detención** de la misión.

El anexo B muestra todo el código de la creación de un LinearLayout, los botones y el fragmento de vista de mapa que se colocó en la parte inferior.

Además, dentro de la carpeta res-->drawable se colocan las imágenes "aircraft.png" y "ic_launcher.png" con el objetivo de dar una representación en el mapa de Google, de la ubicación del vehículo aéreo no tripulado y la presentación del ícono en el teléfono móvil para el usuario; respectivamente, véase la figura 36.

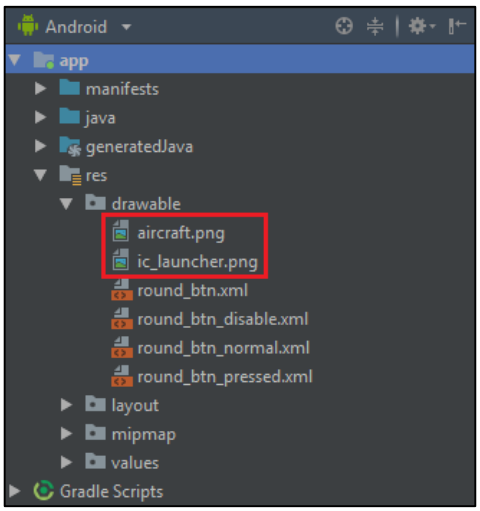


Fig. 36: Colocación de imágenes, utilizadas por el activity_main.xml

De igual manera, en el archivo AndroidManifest.xml se actualiza el elemento de actividad "MainActivity" con los atributos que se muestra a continuación:

```
<activity
        android:name=".MainActivity"
        android:configChanges="orientation|screenSize"
        android:label="@string/title_activity_mainactivity"
        android:theme="@android:style/Theme.NoTitleBar.Fullscreen" >
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
</activity>
```

Así mismo, en el archivo "strings.xml" de la carpeta values se agrega el siguiente contenido de cadena:

```
<string name="title_activity_demo">GSDemoActivity</string>
<string name="google_app_id">Google_App_ID_String</string>
```

Una vez finalizado los pasos previos; si se marca el archivo "activity_main.xml", se visualiza la captura de pantalla de vista previa de MainActivity como se muestra en la figura 37:

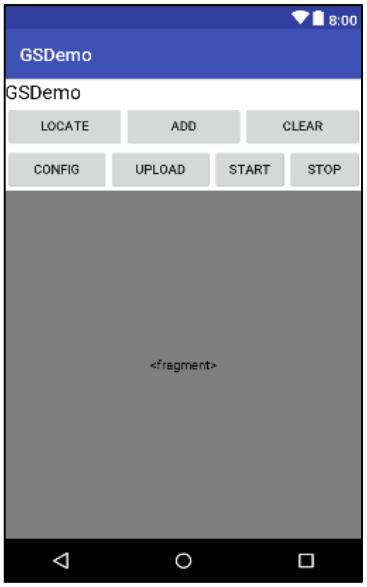


Fig. 37: Vista previa de la estructura de MainActivity.

De esta manera como acto seguido, se crea un nuevo archivo xml llamado "dialog_waypointsetting.xml" en la carpeta de diseño, este archivo ayudará a la configuración de una vista de texto para ingresar "Altitud" y crear tres grupos de botones de radio para seleccionar Velocidad, Acción después de finalizado y modo de vuelo, para el código fuente de programación véase el anexo C.

Ahora, si se verifica el archivo mencionado, se podrá observar la captura de pantalla de vista previa del Diálogo de Configuración de punto de ruta como se muestra en la imagen 38.

Fig. 38: Vista previa del archivo "dialog_waypointsetting.xml"

La clase MainActivity.

En este archivo, véase el anexo D, se implementa las siguientes funciones.

- Se crea una variable de GoogleMap y 7 variables de miembro de botón para la interfaz de usuario. De la misma forma, se crea el método initUI() para iniciar las 7 variables del botón e implementar así el método setOnClickListener y pasarlo como parámetro. [59]
- Se implementó el método showSettingDialog para mostrar el cuadro de diálogo de alerta de Configuración de Waypoint o cuadro de diálogo de configuración cuando se presione el botón Config .
- Por último, se anula el método onMapReady() para inicializar el mapa de Google e invocar el método setUpMap() para implementar el método setOnMapClickListener(). Luego se agregó un marcador en Ambato – Ecuador la misma que será el punto de home, entonces, cuando se cargue el mapa de Google, verá una etiqueta roja en esta posición.

Registro de la aplicación

Realizado los pasos anteriores, se procede a la registración de la aplicación, generando una clave que se aplica desde el sitio web de DJI Developer. [60]

Registro como desarrollador DJI

Durante el proceso de registro, se deberá proporcionar información de correo electrónico y una tarjeta de crédito o un número de teléfono para verificar el registro. Es importante mencionar que cualquier información de la tarjeta de crédito proporcionada solo se utilizará para la verificación y no se cobrará.

Se proporciona de un correo electrónico y una contraseña en la interfaz de registro de desarrolladores de DJI, figura 39.

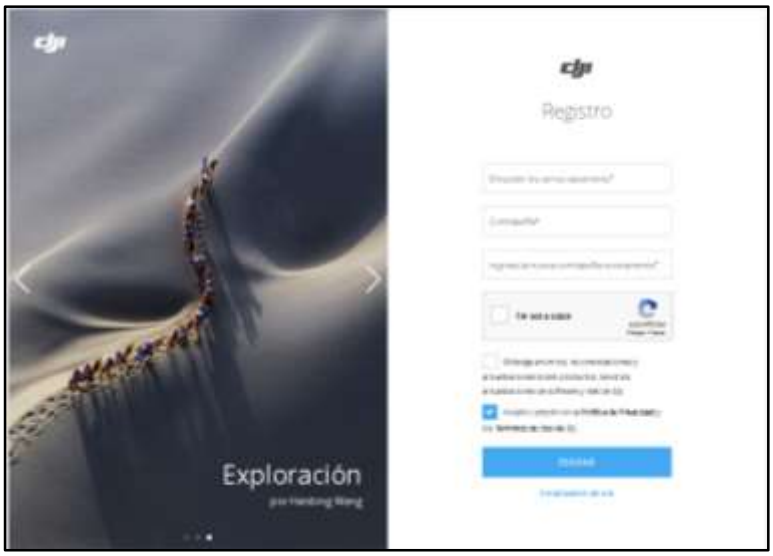


Fig. 39: Interfaz de registro de desarrollador de DJI

Cada aplicación necesita una clave de aplicación única para inicializar el SDK.

Luego de realizar el registro exitosamente, se procede a la creación de una App Key para la aplicación de punto de ruta de este proyecto.

- En el centro de Usuarios de desarrolladores de DJI, se selecciona la pestaña aplicaciones de la izquierda y se crea una aplicación en el botón de la derecha.

- Se ingresa el nombre de la plataforma, el identificador del paquete, la categoría y la descripción de la aplicación; es importante mencionar que el identificador de paquete es el nombre del paquete, figura 40.

Fig. 40: Creación de la App Key de DJI

- Se enviará un correo electrónico de activación de la aplicación para completar la generación de la clave de la aplicación.
- La clave de la aplicación aparecerá en el centro del usuario y se puede copiar y pegar dentro del código fuente de la aplicación.

Es de carácter sustancial detallar que, cuando la aplicación se ejecuta por primera vez, esta clave se remitirá a un servidor DJI para confirmar que la aplicación puede usar el SDK. Si tiene éxito, el resultado se almacenará localmente en la memoria caché del dispositivo móvil. Cada inicialización subsiguiente de la aplicación en el dispositivo móvil verificará el caché local si no hay conexión a Internet.

Por lo tanto, la primera vez que se ejecuta una aplicación en un dispositivo móvil, éste necesitará tener una conexión a la nube. Después de eso, no se requerirá de la conexión para iniciar la misión de punto de ruta, pero se utilizará la conectividad cuando esté disponible, de esta manera se confirma que la aplicación aún tiene autorización para controlar los productos DJI. [61]

Modificación del archivo AndroidManifest

Se agrega la clave generada anteriormente, así como la clave de Google Maps en esta parte de código del mismo archivo AndroidManifest.xml.

```
<uses-library android:name="com.android.future.usb.accessory" />
<meta-data
   android:name="com.dji.sdk.API_KEY"
   android:value="Favor ingrese aqui su clave de aplicacion." />
```

Implementación del registro del ConnectionActivity.

Luego de la implementación de la lógica de registro "ConnectionActivity.java" (véase el anexo E); se ejecuta el proyecto en el dispositivo Android en donde se visualizará el siguiente texto “Register Success” como se muestra en la figura 41.

Esta parte del proyecto de punto de ruta verifica la comunicación estable que se canalice entre el dispositivo móvil y el UAV, mostrando en la pantalla que la conexión se ha realizado y el tipo de aeronave conectado para el desarrollo de la misión.

Cabe mencionar que, si no existe ningún dispositivo conectado entre el dispositivo móvil y la aeronave, la aplicación no se puede inicializar.

Fig. 41: Registro correcto de la app a los servidores DJI

Configuración final de la aplicación de punto de ruta

Configuración del botón "LOCATE"

Se procede a mostrar la ubicación de la aeronave en Google Map e intentarla ampliar automáticamente para de esta manera poder observar el área circundante del vehículo aéreo no tripulado.

De esta manera se muestra en la figura 42 el seudocódigo del botón "locate" que se escribió en el archivo MainActivity.java para la ubicación del UAV.

```
private void updateDroneLocation(){

    LatLng pos = new LatLng(droneLocationLat, droneLocationLng);
    //Create MarkerOptions object
    final MarkerOptions markerOptions = new MarkerOptions();
    markerOptions.position(pos);
    markerOptions.icon(BitmapDescriptorFactory.fromResource(R.drawable.aircraft));

    runOnUiThread(new Runnable() {
        @Override
        public void run() {
            if (droneMarker != null) {
                droneMarker.remove();
            }

            if (checkGpsCoordination(droneLocationLat, droneLocationLng)) {
                droneMarker = gMap.addMarker(markerOptions);
            }
        }
    });
}
```

Fig. 42: Codificación del botón "Locate" para la ubicación del UAV en Google Map.

También se necesita dar a conocer el estado de conexión del drone y una vez conectado invocar el método onReceive() para este cambio de estado; de la misma manera también se activa la cámara para acercar el mapa de Google Map y de esta manera dar a conocer la ubicación del avión no tripulado, cabe mencionar que la cámara no realizará captura de ninguna imagen ni grabación de video, solo trabaja en segundo plano para acceder a la ubicación actual en donde se encuentre la aeronave.

Si se verifica la aplicación en este instante, se mostrará un pequeño avión rojo en el mapa; de la misma manera si se desea acercar la vista o no se encuentra la aeronave se puede ayudar del botón LOCATE, como se muestra en la figura 43.

Si se abre la aplicación o si se desea ejecutar este proyecto en un área distinta por primera vez, es necesario la conexión a una red de Internet para lograr proporcionar la ubicación exacta del drone.

De igual manera se debe realizar la calibración del Compas, con la ayuda de la aplicación “DJI GO” disponible en la tienda Google Play para Android.

Fig. 43: Localización de la aeronave por GPS.

Codificación de los marcadores de Waypoint, botón “Add”

El botón “Add” permite añadir los puntos de recorrido del UAV en el mapa; las características de cada punto (longitud, latitud, altitud) son guardados en un array para posteriormente cargarlos en memoria y proceder a ejecutar la misión.

Una vez que el mapa muestre de manera correcta la ubicación de la aeronave, se procede a la añadidura de los marcadores de la misión de punto de ruta agregando mMarkers, como se muestra en la figura 44.

```
@Override
public void onMapClick(LatLng point) {
    if (isAdd == true){
        markWaypoint(point);
        Waypoint mWaypoint = new Waypoint(point.latitude, point.longitude, altitude);
        //Add Waypoints to Waypoint arraylist;
        if (waypointMissionBuilder != null) {
            waypointList.add(mWaypoint);
            waypointMissionBuilder.waypointList(waypointList).waypointCount(waypointList.size());
        }else
        {
            waypointMissionBuilder = new WaypointMission.Builder();
            waypointList.add(mWaypoint);
            waypointMissionBuilder.waypointList(waypointList).waypointCount(waypointList.size());
        }
    }else{
        setResultToToast("Cannot Add Waypoint");
    }
}
```

Fig. 44: Codificación de los marcadores de Waypoint.

De igual manera se implementa métodos onMapClick() el cual se invocará cuando el usuario toque en la Vista del mapa. Cuando el usuario toque en una posición diferente de esta, se crea los determinados “punto de ruta”

De esta manera si se vuelve a compilar el proyecto, se debería observar la siguiente interfaz, como se muestra en la figura 45.

Fig. 45: Marcadores de la misión Waypoint.

Codificación del botón “CLEAR”

“Clear” borra todos los parámetros de una misión ya realizada para tener facilidad de configurar otra misión, de esta manera los puntos de recorrido se eliminan y los valores en el sensor de giroscopio con respecto a los ejes de inclinación, balanceo y giro se restablecen.

Clear invoca al método *waypointList.clear* como se muestra en la figura 46.

```
case R.id.clear:{
    runOnUiThread(new Runnable() {
        @Override
        public void run() {
            gMap.clear();
        }

    });
    waypointList.clear();
    waypointMissionBuilder.waypointList(waypointList);
    updateDroneLocation();
    break;
}
```

Fig. 46: Codificación del botón clear para borrar datos de la previa misión.

Codificación del botón “UPLOAD”

El botón “Upload” permite cargar la misión de puntos de recorrido a través del método uploadWayPointMission, véase la figura 47, de esta manera el UAV estará listo para el despegue y recorrido por los puntos de ruta previamente establecidos y parametrizados.

```
private void uploadWayPointMission(){

    getWaypointMissionOperator().uploadMission(new
CommonCallbacks.CompletionCallback() {
        @Override
        public void onResult(DJIError error) {
            if (error == null) {
                setResultToToast("Mission upload successfully!");
            } else {
                setResultToToast("Mission upload failed, error: " +
error.getDescription() + " retrying...");
                getWaypointMissionOperator().retryUploadMission(null);
            }
        }
    });

}
```

Fig. 47: Codificación del botón upload, para cargo de misión.

Codificación del botón “START”

El botón “start” dará inicio a la misión que fue previamente cargada y de esta manera realizará el recorrido por los puntos de ruta programados, la figura 48 muestra el seudocódigo de la programación del botón Start.

```
private void startWaypointMission(){

    getWaypointMissionOperator().startMission(new CommonCallbacks.CompletionCallback() {
        @Override
        public void onResult(DJIError error) {
            setResultToToast("Mission Start: " + (error == null ? "Successfully" : error.getDescription()));
        }
    });
}
```

Fig. 48: Codificación del botón start, para inicio de misión.

Codificación del botón “STOP”

Este botón permite detener la misión en cualquier momento en que esta se esté realizando, una vez detenida la misión, el UAV se mantendrá en el aire esperando instrucciones del usuario a través del JoyStick, la figura 49 muestra el seudocódigo de la programación del botón Stop.

```
private void stopWaypointMission(){

    getWaypointMissionOperator().stopMission(new CommonCallbacks.CompletionCallback() {
        @Override
        public void onResult(DJIError error) {
            setResultToToast("Mission Stop: " + (error == null ? "Successfully" : error.getDescription()));
        }
    });

}
```

Fig. 49: Codificación del botón stop, para detener de misión.

Configuración del botón "CONFIG"

El botón de configuración "Config" abre una interfaz auxiliar como se mostró en la figura 38, en donde el usuario establece parámetros importantes de la misión de punto de ruta como:

- Altura de vuelo
- Velocidad (Low, Mid, High)
- Acción después de finalizar la misión.
- Heading, encabezamiento o modo de vuelo.

A continuación, se describe cada parámetro descrito en la lista anterior.

Altura de vuelo

Para verificar la altura de vuelo, el drone pone en acción el sensor barómetro programado de manera transparente gracias al SDK Móvil de Android, este sensor mide la presión inicial antes del despegue y la presión después del despegue para ajustar a la altura parametrizada por el usuario gracias a un sistema de lazo cerrado, como se muestra en la figura 50.

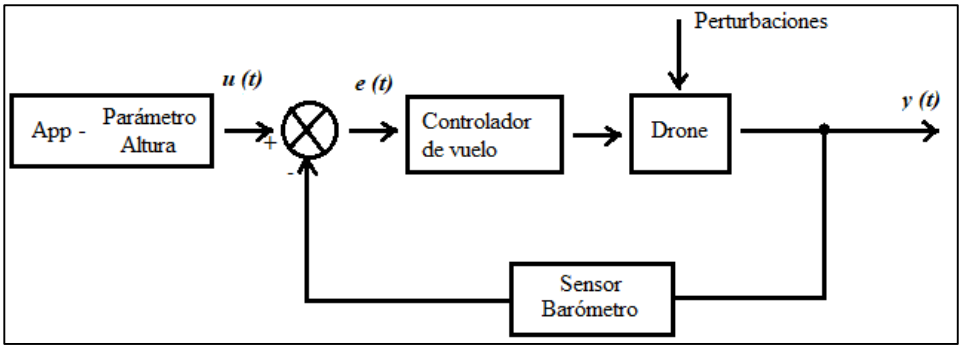


Fig. 50: Diagrama de bloques de la altura de vuelo, desarrollado por el investigador.

Velocidad de vuelo.

La velocidad de vuelo se mide de acuerdo al tiempo que se demora en recorrer la aeronave una trayectoria.

El sensor Throttle o acelerómetro es el encargado de aumentar o disminuir la velocidad de los rotores para establecer la velocidad de vuelo deseada por el usuario

Las velocidades establecidas en la aplicación de puntos de ruta fueron las siguientes:

- Low – 1 metro por segundo
- Mid – 1.5 metros por segundo
- High – 2 metros por segundo

Los detalles completos del análisis de la velocidad de vuelo se presentan en la sección 4.4.2 de este documento.

La figura 51 muestra el diagrama de bloques de la velocidad de vuelo.

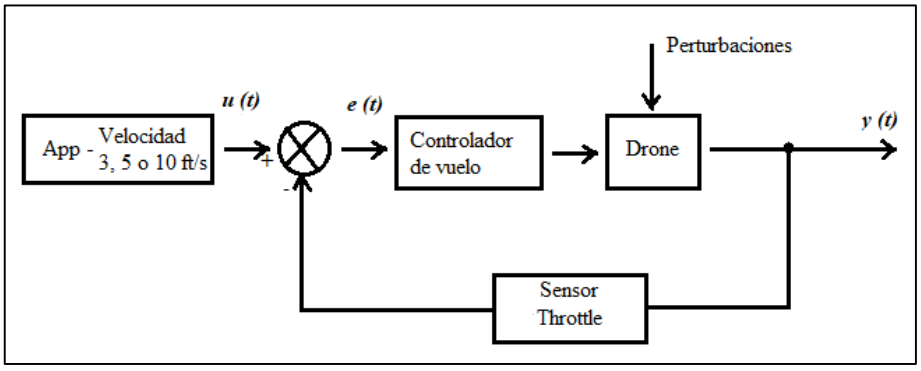


Fig. 51: Diagrama de bloques de la altura de vuelo, desarrollado por el investigador.

La figura 52 muestra el seudocódigo de la programación de la velocidad de vuelo.

```
public void onCheckedChanged(RadioGroup group, int checkedId) {
        if (checkedId == R.id.lowSpeed){
            mSpeed = 3.0f;
        } else if (checkedId == R.id.MidSpeed){
            mSpeed = 5.0f;
        } else if (checkedId == R.id.HighSpeed){
            mSpeed = 10.0f;
        }
    }

});
```

Fig. 52: Codificación de la altura de vuelo del UAV.

Acción después de finalizar.

En la configuración de los parámetros de la misión de puntos de ruta se puede seleccionar una tarea como último acto de la misión los cuales le mencionan a continuación:

- **Autoland**

  La función Autoland o aterrizaje autónomo hace que el avión aterrice automáticamente en el último punto de ruta establecido.

- **Back to 1st.**

  La tarea de Back to 1st, desplazará la aeronave al primer punto de referencia establecido en la misión y se mantendrá en el aire esperando por instrucciones a través del Joystick de la aeronave.

- **Go Home.**

  La función “ve a casa” desplazará al vehículo aéreo no tripulado a casa cuando la misión esté completa.

  La localización de casa se configura en la programación con datos de longitud y latitud de la ubicación en donde el usuario establezca el punto de casa, véase la figura 53.

  Cabe mencionar que el punto de casa deberá estar a no más de 500 metros de donde se realiza la misión, para que el avión ejecute esta tarea, si casa se encuentra fuera del limite el avión esperará en posición del punto de ruta por instrucciones del usuario mediante el Joystick.

```
    public void onMapReady(GoogleMap googleMap) {
        if (gMap == null) {
            gMap = googleMap;
            setUpMap();
        }

        LatLng ambato = new LatLng(-1.230298, -78.604545);
        gMap.addMarker(new MarkerOptions().position(ambato).title("Marker in Ambato"));
        gMap.moveCamera(CameraUpdateFactory.newLatLng(ambato));
    }

}
```

Fig. 53: Codificación del punto casa.

- **NO_ACCION.**

  Esta función mantendrá al UAV en posición de vuelo en el último punto de ruta establecido luego de terminar el recorrido. Las acciones que se realicen después se las harán a través del JoyStick de la aeronave.

Para detalles de la programación del botón "Config" véase el anexo D.

Modo de vuelo

El modo de vuelo determina el giro que realiza la aeronave en cada punto de ruta de acuerdo al siguiente en su secuencia, es decir, el UAV determinará la posición del siguiente punto de ruta de acuerdo al actual y moverá sus ejes (roll, pitch, yaw) de acuerdo al modo establecido, para desplazarse al punto siguiente.

Los modos de vuelo son los siguientes:

- **AUTO.**

  El modo de vuelo "Auto" se basa en el sistema de coordenadas de cuerpo puesto que establece el sentido del eje X o frente de la aeronave hacia el siguiente punto de ruta, en otras palabras, el drone gira sobre el eje Z en movimiento yaw para hacer coincidir el frente de la aeronave con la dirección del punto siguiente, mírese la figura 54, sin embargo el movimiento sobre el eje X (roll) no se efectúa en esta acción, debido a que la aeronave no hace un despliegue hacia la derecha ni hacia la izquierda.

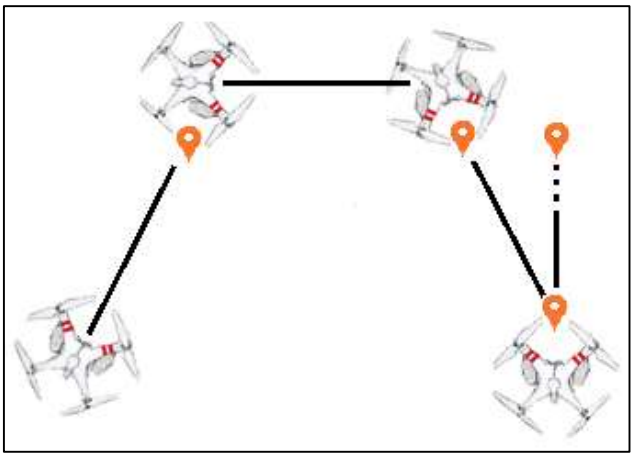


Fig. 54: Modo de vuelo "AUTO", desarrollado por el investigador.

- **INITIAL.**

La función "Initial" hace que el drone se desplace a través de los puntos de ruta de acuerdo a la posición inicial de despegue, es decir, el sistema de coordenadas de cuerpo no se altera, manteniendo así la posición inicial hasta el último punto de ruta establecido, sin importar la dirección de los mismos, véase la figura 55.

En esta función de giro no interviene movimientos sobre el eje Z (yaw) por lo que el frente de la aeronave permanece estable, sin embargo, los movimientos de roll y pitch si intervienen independientemente de la posición de puntos para el despliegue del avión sobre el espacio aéreo.

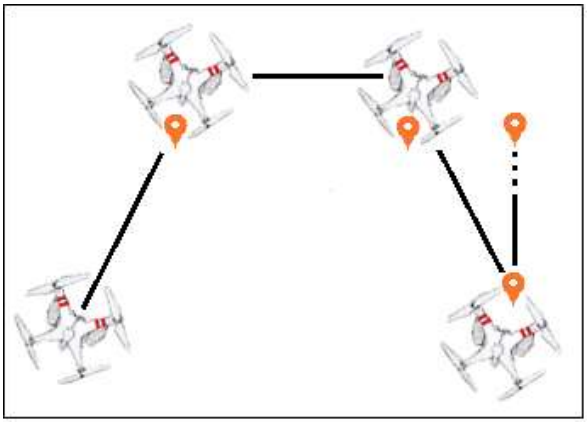


Fig. 55: Modo de vuelo "INITIAL"

- **USING_WAYPOINT.**

  La función del uso de Waypoint hará que el drone tome decisiones propias de giro en cada punto de ruta de acuerdo a donde se encuentre el punto siguiente y así establecer rumbo hacia este punto.

  Es decir, cuando el UAV llega a un punto de ruta toma la mejor decisión de giro (Auto, Initial) de acuerdo al punto de ruta siguiente y establece su rumbo hasta éste.

4.4 Resultados

Para el estudio de resultados se realizarán los siguientes análisis:

- Análisis del recorrido del UAV por los puntos de ruta.

  - ✓ Sin acoplamiento del sistema de pulverización.
    Examina si la aplicación diseñada para el dispositivo móvil Android funciona correctamente.
  - ✓ Con acoplamiento del sistema de pulverización.
    Verifica si el centro de masas del avión es afectado

- Análisis de la altura y velocidad de UAV.
- Análisis de la calidad de pulverización del prototipo realizado con respecto al uso de bombas de espalda.
- Análisis de las condiciones ambientales para realizar la pulverización autónoma.
- Análisis de la cantidad de volumen de material fitosanitario para la aplicación

4.4.1 Análisis del recorrido del UAV por los puntos de ruta.

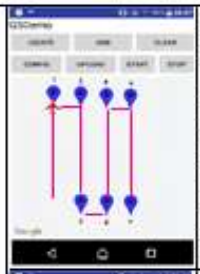
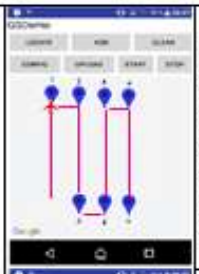
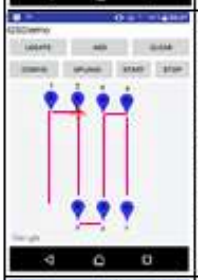
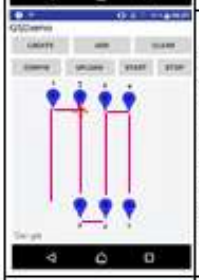
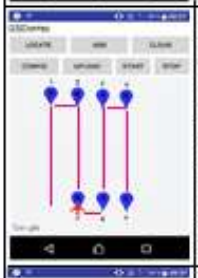
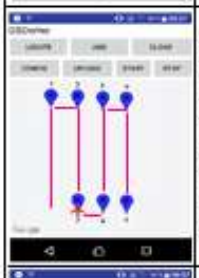
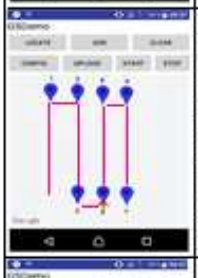
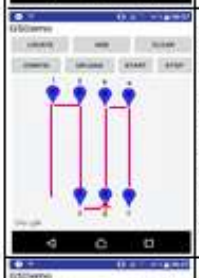
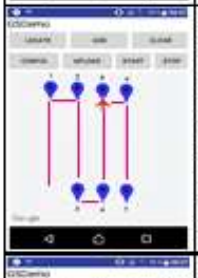
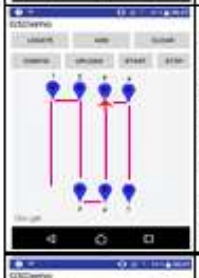
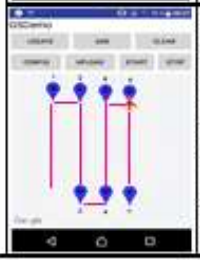
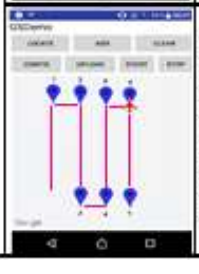
Sin acoplamiento del sistema de pulverización.

Para el análisis de recorrido de puntos de ruta se estableció la trayectoria que se muestra en la figura 56, realizando las pruebas de movimiento de giro en cada punto establecido con los modos de vuelo Auto e Initial; sin peso adicional

Fig. 56: Trayectoria trazada para el análisis del recorrido del UAV, sin acoplamiento del sistema de pulverización.

En la tabla 9 se muestra el estudio de los giros realizados en los modos de vuelo auto e initial, de acuerdo a la trayectoria mostrada en la figura 56.

Tabla 9. – Análisis del recorrido del UAV por los puntos de ruta sin acoplamiento del sistema de pulverización.

Análisis del recorrido del UAV por los puntos de ruta.						
Sin sistema de pulverización						
Prueba 1	Modo de vuelo Auto		**Observaciones** El UAV llega al primer punto de ruta y gira su frente de manera adecuada sin complicaciones, hacia el proximo punto.	Modo de vuelo Inicial		**Observaciones** El UAV llega al primer punto de ruta y se mueve hacia el siguiente con la misma orientacion de despegue.
Prueba 2			**Observaciones** El UAV llega al segundo punto de ruta y gira su frente de manera adecuada sin complicaciones, hacia el proximo punto.			**Observaciones** El UAV llega al segundo punto de ruta y se mueve hacia el siguiente con la misma orientacion de despegue.
Prueba 3			**Observaciones** El UAV llega al tercer punto de ruta y gira su frente de manera adecuada sin complicaciones, hacia el proximo punto.			**Observaciones** El UAV llega al tercer punto de ruta y se mueve hacia el siguiente con la misma orientacion de despegue.
Prueba 4			**Observaciones** El UAV llega al cuarto punto de ruta y gira su frente de manera adecuada sin complicaciones, hacia el proximo punto.			**Observaciones** El UAV llega al cuarto punto de ruta y se mueve hacia el siguiente con la misma orientacion de despegue.
Prueba 5			**Observaciones** El UAV llega al quinto punto de ruta y gira su frente de manera adecuada sin complicaciones, hacia el proximo punto.			**Observaciones** El UAV llega al quinto punto de ruta y se mueve hacia el siguiente con la misma orientacion de despegue.
Prueba 6			**Observaciones** El UAV llega al sexto punto de ruta y gira su frente de manera adecuada sin complicaciones, hacia el proximo punto.			**Observaciones** El UAV llega al sexto punto de ruta y se mueve hacia el siguiente con la misma orientacion de despegue.

Cuando no se tiene anexado el sistema de pulverización al avión, el centro de empuje y el centro de masas coinciden, lo que se traduce en una buena estabilización al momento del desplazamiento por los puntos de ruta asignados para la misión.

De acuerdo a los resultados mostrados de la tabla 9, en modo "Auto" el UAV cumple con la trayectoria asignada sin ningún problema, realizando los giros característicos de este modo de vuelo en cada punto de ruta, sin embargo, el modo de vuelo "Initial" produce una leve desviación en los puntos establecidos, como se muestra en la figura 57, esta desviación genera un valor de error no mayor a 5 cm, lo cual se considera un valor despreciable en comparación a las superficies de campo de plantaciones de frutilla.

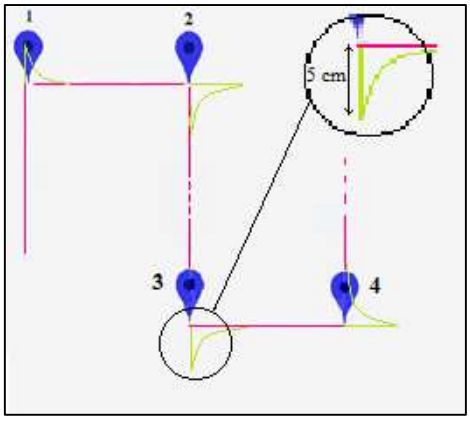


Fig. 57: Desviación del UAV en modo de vuelo "Initial"

Es importante mencionar también que el modo de vuelo "Auto" tarda en girar su frente hacia el siguiente punto a razón de 90^{o}/s, mientras que en modo "Initial" este retardo no se suscita debido a que no se hacen giros de frente en los puntos de ruta.

En conclusión, los modos de vuelo "Auto" e Initial", responden de manera adecuada "sin el sistema de pulverización" respecto a la trayectoria de puntos de ruta establecidos por el usuario en la aplicación desarrollada en Android; el punto de inicio y fin de la misión se muestran en la figura 58.

Fig. 58: Punto inicial y final del recorrido del UAV por los puntos de ruta sin acoplamiento del sistema de pulverización.

Con acoplamiento del sistema de pulverización.

Para el análisis del recorrido del UAV por los puntos de ruta con acoplamiento del sistema de pulverización se traza la trayectoria mostrada en la figura 59; cómo se puede observar esta trayectoria es similar a la mostrada en la figura 56, debido a que las plantaciones de frutilla se realizan en camellones paralelos entre sí, razón por la cual se plantea las trayectorias a cumplir por el UAV.

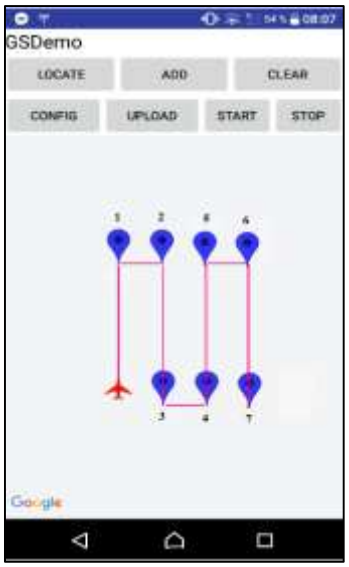
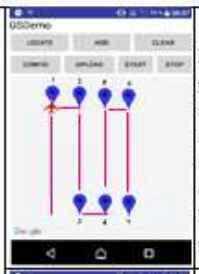
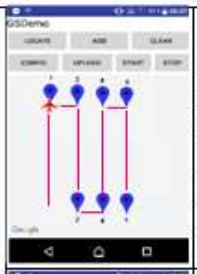
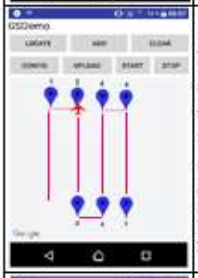
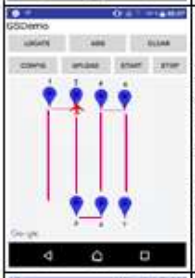
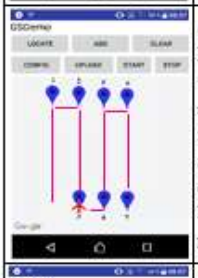
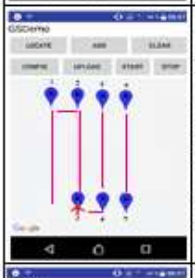
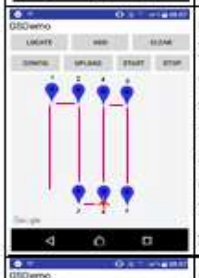
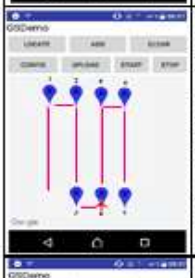
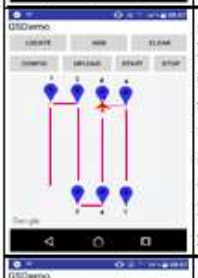
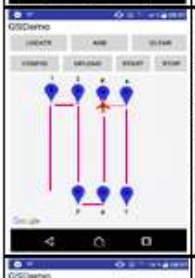
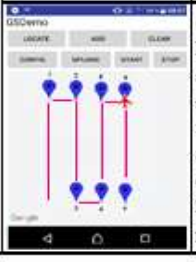
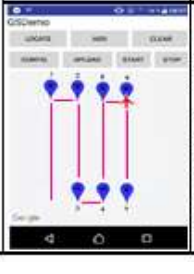


Fig. 59: Trayectoria trazada para el análisis del recorrido del UAV, con acoplamiento del sistema de pulverización.

En el análisis del centro de masas cuando el sistema de pulverización es acoplado al UAV realizado en el literal b de la sección 4.3.1, se indica la desviación de 4 milímetros del centro de masas, por lo que en la tabla 10 se hace el estudio de la actitud de la aeronave en los modos de vuelo “Auto” e “Initial” con dicho desvío, de acuerdo a la trayectoria trazada mostrada en la figura 59.

La tabla 10 se muestra el estudio del recorrido del UAV por los puntos de ruta con acoplamiento del sistema de pulverización.

Tabla 10. – Análisis del recorrido del UAV por los puntos de ruta con acoplamiento del sistema de pulverización.

Análisis del recorrido del UAV por los puntos de ruta.						
Con sistema de pulverización						
Prueba 1	Modo de vuelo Auto		**Observaciones** El UAV llega al primer punto de ruta y gira su frente de manera adecuada sin complicaciones, hacia el proximo punto.	Modo de vuelo Inicial		**Observaciones** El UAV llega al primer punto de ruta y se mueve hacia el siguiente con la misma orientacion de despegue.
Prueba 2			**Observaciones** El UAV llega al segundo punto de ruta y gira su frente de manera adecuada sin complicaciones, hacia el proximo punto.			**Observaciones** El UAV llega al segundo punto de ruta y se mueve hacia el siguiente con la misma orientacion de despegue.
Prueba 3			**Observaciones** El UAV llega al tercer punto de ruta y gira su frente de manera adecuada sin complicaciones, hacia el proximo punto.			**Observaciones** El UAV llega al tercer punto de ruta y se mueve hacia el siguiente con la misma orientacion de despegue.
Prueba 4			**Observaciones** El UAV llega al cuarto punto de ruta y gira su frente de manera adecuada sin complicaciones, hacia el proximo punto.			**Observaciones** El UAV llega al cuarto punto de ruta y se mueve hacia el siguiente con la misma orientacion de despegue.
Prueba 5			**Observaciones** El UAV llega al quinto punto de ruta y gira su frente de manera adecuada sin complicaciones, hacia el proximo punto.			**Observaciones** El UAV llega al quinto punto de ruta y se mueve hacia el siguiente con la misma orientacion de despegue.
Prueba 6			**Observaciones** El UAV llega al sexto punto de ruta y gira su frente de manera adecuada sin complicaciones, hacia el proximo punto.			**Observaciones** El UAV llega al sexto punto de ruta y se mueve hacia el siguiente con la misma orientacion de despegue.

Según los resultados mostrados en la tabla 10, la desviación de 4 mm del centro de masas con respecto al centro de empuje, no interfiere en el momento de realizar la trayectoria a través de los puntos de ruta en el modo de vuelo “Auto”, donde los giros de frente que instaura este modo se realizan de manera correcta, sin embargo, el modo de vuelo “Initial” muestra el mismo error de 5 cm, mostrado en la figura 57, lo que de acuerdo a la superficie de los campos de frutilla se considera también despreciable.

No obstante, se recomienda utilizar el modo de vuelo “Auto” debido a que este no presenta ningún tipo de error en el desplazamiento por los puntos de trayectoria, mírese la figura 60.

Fig. 60: Desplazamiento de la aeronave por los puntos de trayectoria.

Además, al girar el frente de la aeronave, el barrido realizado por las boquillas de pulverización ubicadas en el tren de aterrizaje quedará de forma correcta para la aspersión de material fitosanitario, es decir, de forma transversal a los camellones; en la figura 61 se muestra el punto inicial y final del recorrido del UAV por los puntos de ruta con acoplamiento del sistema de pulverización.

Fig. 61: Punto inicial y final del recorrido del UAV por los puntos de ruta con acoplamiento del sistema de pulverización.

Es importante mencionar que cada punto de ruta de la misión estará separado el uno del otro de manera horizontal (lateral) por una distancia mínima de aproximadamente 2.5 metros debido a que el GPS tiene este margen de error en la localización, por lo que, el UAV realizará el recorrido de las hileras haciendo el salto de una como se muestra en la figura 62, dado que cada camellón tiene entre 70 - 80 cm de grosor y están separados entre ellos por un surco de 50 cm. [19]

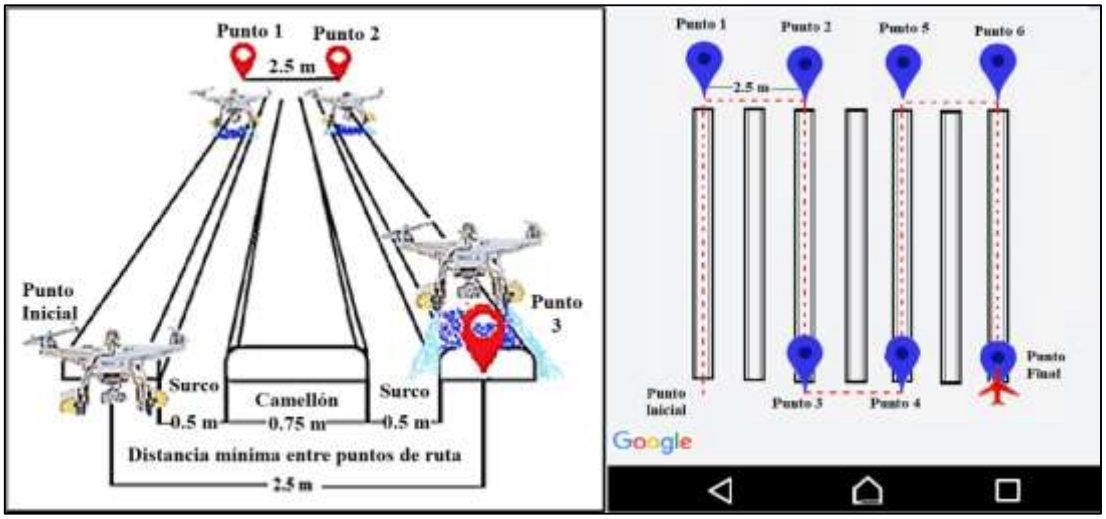


Fig. 62. Distancia mínima entre puntos de ruta.

4.4.2 Análisis de la altura y velocidad de UAV.

La determinación de la altura de la aeronave se realiza enfocado al ancho de rocío del sistema pulverizador debido a que existe una estrecha relación entre estas dos. Conjuntamente el estudio de la velocidad del UAV se realiza enfocado a la efectividad de la calidad de cobertura de pulverización en las plantaciones.

Análisis de la altura del UAV.

La altura del avión está ligada al ancho de trabajo que asperje el sistema de pulverización, puesto que, más altura genera más anchura de pulverización.

De acuerdo a [19], los estándares de las medidas para la confección de camellones, cada una debe medir 60 cm en la parte superior y 70-80 cm en la base, separados entre ellos por un surco de 50 cm, es decir se necesita un ancho de pulverización aproximado de 1.1 metros,

De esta manera se procede al cálculo de la anchura de trabajo mediante la ecuación 2, con valores de altura de 2, 2.5 y 3 metros desde su lugar de despegue.

$$at = (ab * h) + [db(nb - 1)]$$

$Ecuación\ (2)$

Donde:

at: Ancho de trabajo del sistema de pulverización.
ab: ancho de asperje de cada boquilla a un metro de altura; este dato se toma a través de medición manual. (0.5 metros)
h: altura del UAV. (experimento a 2, 2.5, y 3 metros)
db: distancia entre boquillas pulverizadoras. (0.15 metros)
nb: número de boquillas pulverizadoras. (2)

Medición del ancho de trabajo a 2 metros de altura.

$$at = (ab * h) + [db(nb - 1)]$$
$$at = (0.5 * 2\ m) + [0.15\ m\ (2 - 1)]$$
$$at = (1\ m) + [0.15\ m\ (1)]$$
$$at = 1\ m + [0.15\ m\ ]$$
$$at = 1.15\ m$$

Medición del ancho de trabajo a 2.5 metros de altura.

$$at = (ab * h) + [db(nb - 1)]$$
$$at = (0.5 * 2.5\ m) + [0.15\ m\ (2 - 1)]$$
$$at = (1.25\ m) + [0.15\ m\ (1)]$$
$$at = 1.25\ m + [0.15\ m\ ]$$
$$at = 1.40\ m$$

Medición del ancho de trabajo a 3 metros de altura.

$$at = (ab * h) + [db(nb - 1)]$$
$$at = (0.5 * 3\ m) + [0.15\ m\ (2 - 1)]$$
$$at = (1.5\ m) + [0.15\ m\ (1)]$$
$$at = 1.5\ m + [0.15\ m\ ]$$
$$at = 1.65\ m$$

Debido a que se necesita una anchura de trabajo de 1.1 metros y conforme a los cálculos realizados previamente, se selecciona la altura de 2 metros, puesto que, a esta altura de vuelo se genera un ancho de 1.15 metros, descartando las mediciones de 2.5 y 3 metros ya que generan un ancho de trabajo mayor, además, se calcula la concentración de barrido a la mediante la ecuación 3, la cual dependerá del resultado obtenido en la ecuación 2.

$$ac = at - 2\,db$$

$$Ecuación\ (3)$$

Donde:

ac: ancho de concentración.
at: Ancho de trabajo del sistema de pulverización (1.15 metros)
db: Distancia entre las boquillas (0.15 metros)

$$ac = at - 2\,db$$
$$ac = 1.15\,m - (2 * 0.15\,m)$$
$$ac = 1.15\,m - 0.3\,m$$
$$ac = 0.85\,m$$

Es así como, a 2 metros de altura del avión, se tiene un ancho de trabajo de 1.15 metros y ancho de concentración de 0.85 metros; estos valores dependen exclusivamente de la medición del ancho de asperje de una boquilla a 1 metro de altura, la separación entre las boquillas y el número de boquillas, véase la ilustración de la figura 63.

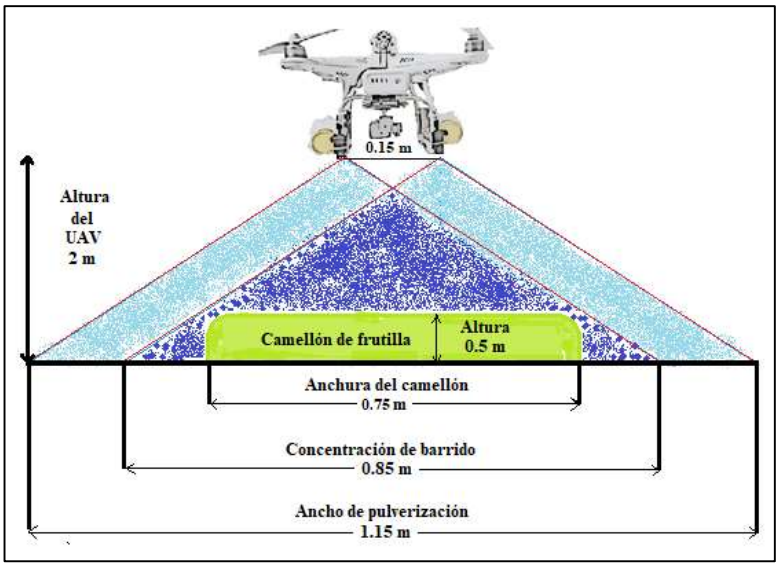


Fig. 63: Anchura de trabajo, respecto a la altura de 2 m del UAV, realizado por el investigador

Si bien los datos anteriores son teóricos; la tabla 11 muestra el análisis de datos teóricos y reales arrojados de la anchura de pulverización respecto a la altura del UAV, cuando se lleva a cabo la misión.

Tabla 11. – Análisis de la anchura de barrido del sistema de pulverización con respecto a la altura del AUV

Análisis de la anchura de barrido del sistema de pulverización con respecto a la altura del AUV				
Pruebas	**Altura parametriza en la app (metros)**	**Altura real del UAV (metros)**	**Anchura de trabajo calculada(metros)**	**Anchura de trabajo real (metros)**
1	2	2	1,15	1,13
2		1,99		1,17
3		2,01		1,15
4		2		1,14
5		2,03		1,16
6		1,97		1,15
7		2		1,16
8		2		1,17
9		1,99		1,15
10		2		1,13
	Promedio	1,999	**Promedio**	1,151

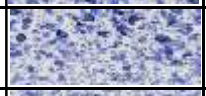
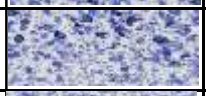
En el análisis de la tabla anterior, los datos arrojados presentan un error de ±2%, por lo que se concluye que la altura apropiada y recomendada para que el avión lleve a cabo la misión es de 2 m, sin embargo, el tiempo de autonomía del drone se ve afectado con una reducción de 20%, es decir, se reduce 5 minutos de autonomía de vuelo, debido al levantamiento del peso adicional del sistema de pulverización.

Análisis de la velocidad del UAV.

El estudio de la correcta velocidad del UAV se realiza enfocado a la efectividad de la calidad de cobertura de pulverización en las plantaciones.

Según el Manual de manejo agronómico de frutilla [19], una pulverización de calidad se daría en un 50% de cobertura, es por lo que en la tabla 12 se hace el estudio de la calidad de pulverización emanada mediante la ayuda de papel hidrosensible, a velocidades de 1, 1.5 y 2 metros por segundo, cuyas velocidades fueron establecidas en la aplicación de puntos de ruta desarrollada en Android.

Tabla 12. – Análisis de calidad de cobertura de pulverización de acuerdo a la velocidad de la aeronave.

Análisis de la calidad de cobertura de pulverización de acuerdo a la velocidad del AUV			
Velocidad parametrizada app (metros/segundo)	Velocidad real del UAV (metros/segundo)	Calidad de pulverización	Porcentaje de cobertura
Low (1 m/s)	1		50 aprox
	1,02		50 aprox
	0,99		50 aprox
	1		50 aprox
	0,99		50 aprox
Mid (1,5 m/s)	1,48		30 aprox
	1,5		30 aprox
	1,49		30 aprox
	1,51		30 aprox
	1,5		30 aprox
High (2 m/s)	2		20 aprox
	2,01		20 aprox
	1,99		20 aprox
	2		20 aprox
	1,98		20 aprox

La determinación de cada velocidad real de la aeronave se realiza mediante la ecuación 1, en donde se estableció una distancia de 20 metros y se midió el tiempo de duración que el drone tarda en recorrer dicha distancia.

$$v\left(\frac{m}{s}\right) = \frac{d(m)}{t(s)}$$

$Ecuación\ (1)$

$$v\left(\frac{m}{s}\right) = \frac{20\ m}{19.5\ s}$$

$$v = 1.02\ m/s$$

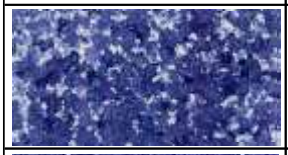
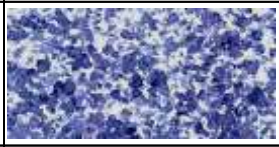
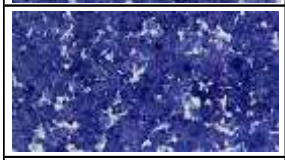
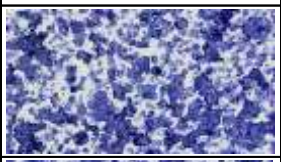
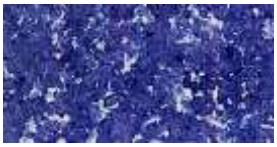
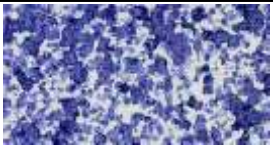
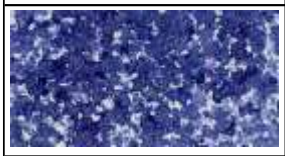
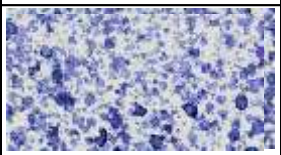
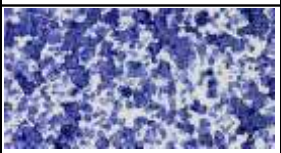
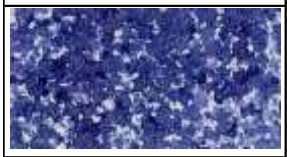
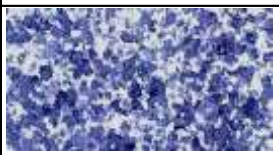
De acuerdo a los resultados arrojados de los tres valores de estudio de velocidad de vuelo en la tabla 12, se descarta las velocidades de 1.5 y 2 metros/segundo, puesto que con estas velocidades el porcentaje de cobertura es ineficiente, sin embargo, a velocidad de 1 metro/segundo el sistema de pulverización genera una cobertura aproximada de 50%. Como ya se explicó anteriormente dicho porcentaje es el apropiado para que la pulverización sea de calidad.

4.4.3 Análisis de la calidad de pulverización del prototipo realizado en esta investigación con respecto al uso de bombas de espalda.

El análisis de la comparación del uso de bombas de espalda y este prototipo, se desarrolla mediante la comparación de porcentaje de cobertura que emana cada uno de los dispositivos mencionados.

Para lo cual en la tabla 13 se hace la comparación del barrido que genera la aeronave a 2 metros de altura y 1 metro por segundo de velocidad con respecto a la pulverización que genera un agricultor con el uso de bombas de espalda a la misma velocidad.

Tabla 13. – Análisis comparativo del porcentaje de pulverización del uso de bombas de espalda y este prototipo.

Análisis de la calidad de pulverización del prototipo realizado con respecto al uso de bombas de espalda.			
Bombas de espalda		Prototipo	
Papel hidrosensible	% de cobertura	Papel hidrosensible	% de cobertura
	85		50
	90		50
	90		50
	90		45
	90		50
	90		50

Cabe mencionar que, los agricultores al momento del uso de mochilas de bombas de espalda realizan la aspersión de producto fitosanitario a una altura del rango de 10-20 centímetros del follaje de las plantaciones de frutilla, lo que hace que la pulverización de los químicos tenga una sobre-concentración con un porcentaje aproximado del 90% de cobertura, esto deriva a una posible contaminación del fruto; además, el uso de las bombas de espalda en el trabajo de la agricultura ocupa pulverizadores con aspersión de tipo abanico lo cual no está considerado en la pulverización de bandas o camellones.

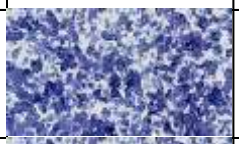
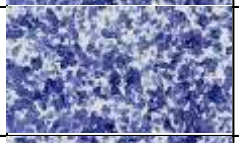
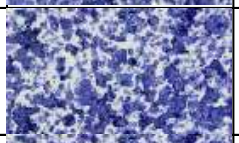
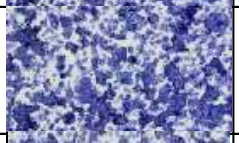
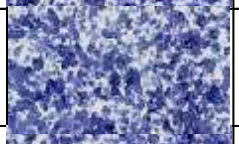
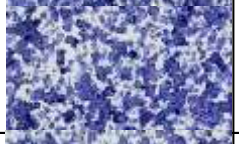
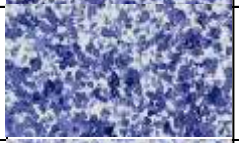
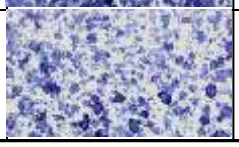
Es por eso que este prototipo hace mención en boquillas con pulverización de aspersión uniforme, a una altura 2 metros, para que se tenga un 50% de cobertura a razón de 40 gotas por cm^2 lo cual es recomendado para tener una óptima calidad de pulverización en estas plantaciones según el manual de manejo agronómico de frutilla escrito por C. Morales [18]

4.4.4 Análisis de las condiciones ambientales para realizar la pulverización autónoma.

El análisis de las condiciones ambientales toma en consideración aspectos relacionados al medio ambiente como la dirección del viento, velocidad del viento, la temperatura y las precipitaciones de lluvia que se puedan presentar en el momento que se desee operar el sistema de pulverización de plantaciones de frutilla asistido por un drone; los datos mencionados anteriormente darán una síntesis al operador para determinar si es propicio o no el uso de dicho sistema.

La dirección del viento, velocidad del viento, temperatura y precipitaciones de lluvia fueron obtenidas gracias a la aplicación “windy” disponible de forma gratuita en la tienda de Google Play para Android, de esta manera en la tabla 14 se examina la cobertura en el follaje de las plantaciones con la ayuda de papel hidrosensible a diferentes condiciones ambientales.

Tabla 14. – Análisis de las condiciones ambientales para conocer si es viable o no realizar la pulverización autónoma.

Dirección y velocidad del viento	Temperatura Ambiente	Hora	Precipitaciones	Gotas por cm2	% de cobertura	Imagen	Vialidad	Observaciones
NO - 1,9 m/s	20° C	14:37	53%	50 aprox	45%		✓	
NO - 1,6 m/s	17° C	16:53	59%	50 aprox	45%		✓	
NE - 0,6 m/s	12° C	7:43	61%	40 aprox	50%		✓	
NO - 1,2 m/s	15° C	9:00	64%	40 aprox	50%		✓	
SE - 1,0 m/s	10° C	7:30	89%	50 aprox	45%		X	Chubascos
NE - 0,9 m/s	11° C	10:34	74%	40 aprox	50%		✓	
NO - 1,2 m/s	13° C	16:52	75%	50 aprox	45%		✓	
SE - 4,3 m/s	11° C	17:01	91%	60 aprox	30%		X	Fuertes vientos

Como se puede observar de acuerdo con la tabla anterior, se debe tener en consideración que no se puede hacer uso del sistema autónomo de pulverización en condiciones ambientales en las cuales la velocidad del viento supere los 4,2 m/s y las precipitaciones de lluvia sean iguales o superiores al 89%; debido a que esto genera inestabilidad en el vehículo aéreo no tripulado cuando trabaja de forma autónoma.

4.4.5 Análisis de la cantidad de volumen de material fitosanitario para la aplicación

Antes de determinar la cantidad de volumen que se necesita en una hectárea de plantación, se procede a la medición del caudal generado por las boquillas acopladas.

Medición del caudal de las boquillas.

Para el caso de la medición del caudal de las boquillas, se toma en cuenta las dos boquillas, y se procede a activar la bomba de presión de agua para que de esta manera se realice la aspersión dentro de un recipiente graduado, teniendo como resultado de esta acción que los 200 ml se realiza en aproximadamente 18 segundos, es decir 0.66 litros por minuto.

Calculo de la cantidad de volumen para la aplicación.

Se realiza el cálculo correspondiente con la ayuda de la ecuación 4.

$$Q = \frac{q * 600}{a * v}$$

$Ecuación\ (4)$

Donde:

Q: Volumen de aplicación en litros por hectárea (Litros por hectárea)
q: Caudal de la boquilla (0.66 litros por minuto)
a: Ancho de trabajo de la/s boquilla/s (1.15 metros)
v: Velocidad del operador (3.6 kilómetros por hora)

$$Q = \frac{0.66 * 600}{1.15 * 3.6}$$

$$Q = 95.65\ {}^{lt}/_{ha}$$

De acuerdo con la velocidad de avance del drone, el ancho de trabajo que este ejerce sobre el campo y el caudal de sus dos boquillas pulverizadoras se tiende a utilizar aproximadamente 96 litros de material fitosanitario para la fumigación autónoma de una hectárea de plantación de frutilla.

CAPÍTULO V

CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

- Si bien el centro de masas de la aeronave es desplazado en 4 milímetros con respecto al centro de empuje debido al acoplamiento del sistema de pulverización; la aeronave realiza el cumplimiento de trayectoria por los puntos de ruta de manera correcta sin pasar por alto que entre cada punto de ruta la distancia mínima es 2.5 metros debido al margen de error en la localización que presenta el GPS, esto se traduce a que el UAV realizará el recorrido de las hileras haciendo el salto de una, dado que cada camellón tiene entre 75 cm de grosor en su base y están separados entre ellos por un surco de 50 cm.

- La altura de vuelo del UAV se determina respecto a la anchura que emerge el sistema de pulverización, de esta manera se asume una altura de vuelo de 2 metros desde el punto de despegue puesto que a dicha altura el sistema de pulverización formado por dos boquillas de aspersión uniforme y espaciamiento de 0.15 metros entre estas, genera un barrido con ancho de trabajo de 1.15 metros y una concentración de gotas de 0.85 metros, medidas con las cuales se cubre de manera adecuada las hileras de plantaciones de frutilla.

- La velocidad de vuelo de la aeronave se establece de acuerdo a la efectividad en la calidad de cobertura en las plantaciones de frutilla, es por esto que en la aplicación de puntos de ruta se ocupa la opción Low, que pertenece a una velocidad de 1 metro por segundo; dicha velocidad origina

una cobertura de pulverización del 50% a razón de 40 gotas por cm^2 la cual se cual se considera apropiada para el tratamiento de fumigación en plantaciones de frutilla.

- Los valores de altura y velocidad arrojados por el avión se cumplen con un rango de error de ±2% en relación a los parametrizados en la aplicación de puntos de ruta diseñada en Android, es decir, una desviación de dichas variables no mayor a ±4 centímetros para el caso de altura y ±2 cm/s para el caso de velocidad, lo cual se considera despreciable, conjuntamente a esto, el tiempo de autonomía de vuelo del drone se ve reducido en 20% de su duración total, debido al mayor trabajo de la aeronave para lograr levantar el peso de 342 gr perteneciente al sistema de pulverización.
- El uso de mochilas o bombas de espalda para la fumigación de plantaciones de frutilla se lo realiza a 10-20 cm de altura de cada follaje, lo que genera una cobertura de pulverización del 90%, la cual se considera excesiva para dar un tratamiento adecuado en dichas plantaciones, no obstante, los valores de pulverización arrojados por el prototipo realizado en esta investigación muestran un porcentaje de cobertura del 50%, cuyo porcentaje es el recomendado para el tratamiento de fumigación en plantaciones de frutilla.

5.2 Recomendaciones

- Antes de realizar la acción de pulverización, se recomienda cargar cada una de las baterías, como la que excita a la bomba de agua del sistema de pulverización, la batería del control remoto y la batería del mismo drone siendo esta última las más indispensable; de igual forma se recomienda trabajar hasta un mínimo de batería del UAV de un 25% para de esta manera evitar posibles accidentes por falta de energía.

- Se debe tener en consideración que no se puede hacer uso del sistema autónomo de pulverización en condiciones ambientales en las cuales la velocidad del viento supere los 4,2 m/s y las precipitaciones de lluvia sean iguales o superiores al 89%; debido a que esto genera inestabilidad cuando el vehículo aéreo no tripulado trabaja de forma autónoma.

- El manejo y configuración en cuanto al uso de la aplicación de puntos de ruta se debe realizar de manera responsable, considerando una altura de vuelo de 2 metros y velocidad de 1 m/s (velocidad baja), de esta manera se dará un máximo provecho en cuanto a la calidad de cobertura en las plantaciones de frutilla.

- Si se desea llevar a cabo este sistema autónomo en un área diferente y por primera vez, es indispensable una conexión a internet o datos móviles en el dispositivo móvil, esto facilitará la localización en el mapa Google de donde se encuentra el UAV y a su vez una mejor interacción en la configuración de cada punto de recorrido.

- Antes del desarrollo de cada misión de punto de ruta se recomienda realiza una limpieza previa del sistema de pulverización, tomando en cuenta que no exista tapones en los conductos transportadores de líquido fitosanitario ni en las boquillas pulverizadoras.

- Se recomienda propiamente para el agro hacer uso de esta tecnología con aeronaves industriales, debido a que estas cuentan con motores de más potencia y es así como se eliminaría la limitación en el momento del levantamiento de peso, de esta manera se podrá acarrear más material fitosanitario e incluso hacer el uso de alguna otra tecnología para ampliar su rendimiento.

BIBLIOGRAFÍA

[1] J. Elcacho, «El herbicida más utilizado del mundo entra en la lista negra del cáncer,» 24 abril 2015. [En línea]. Available: http://www.lavanguardia.com/natural/20150322/54428349874/iarc-incluye-glifosato-monsanto-lista-probable-causa-cancer.html. [Último acceso: 01 mayo 2018].

[2] A. Rebossio, «Los médicos ligan el cáncer de un pueblo argentino a los agroquímicos,» 13 abril 2015. [En línea]. Available: https://elpais.com/internacional/2015/04/13/actualidad/1428944889_293771.html. [Último acceso: 02 mayo 2018].

[3] A. Naranjo, «La otra guerra: Situación de los plaguicidas en Ecuador,» Quito, 2017.

[4] N. Villacrés, «Uso de plaguicidas en el cultivo de papa,» 2014. [En línea]. Available: http://repo.uta.edu.ec/bitstream/123456789/7003/1/tesis-011%20Maestr%C3%ADa%20en%20Agroecolog%C3%ADa%20y%20Ambiente%20-%20CD%20227.pdf. [Último acceso: 02 mayo 2019].

[5] M. A. Correa, «Problemas fitosanitarios del cultivo de frutilla que justifican la desinfección del suelo,» de *Cultivo de Frutilla*, Santiago de Chile, 2015, p. 150.

[6] Agriculturers, «Drones para agricultura,» 10 febrero 2017. [En línea]. Available: http://agriculturers.com/drones-para-agricultura-beneficios-y-casos-reales/. [Último acceso: 02 mayo 2018].

[7] D. Yappalla, «Development and evaluation of drone mounted sprayer for pesticide applications to crops,» 21 octubre 2017. [En línea]. Available: https://ieeexplore.ieee.org/document/8239330/. [Último acceso: 2018 mayo 02].

[8] S. D. Kale, «Agriculture Drone for Spraying Fertilizer and Pesticides,» diciembre 2015. [En línea]. Available: http://ijarcsse.com/Before_August_2017/docs/papers/Volume_5/12_December2015/V5I12-0222.pdf. [Último acceso: 02 mayo 2018].

[9] F. Matsuura, «Research of crop-sprayer for dotted farmland using airflow induced by UAV,» 13 noviembre 2017. [En línea]. Available: https://ieeexplore.ieee.org/document/8105593/. [Último acceso: 02 mayo 2018].

[10] T. F. Oliver Nuñez, «Monitorización de cultivos utilizando drones,» 18 febrero 2015. [En línea]. Available: http://vinculacion.dgire.unam.mx/Congreso-Trabajos-pagina/Trabajos-2015/2-Ciencias%20Fisicomatem%C3%A1ticas%20y%20de%20as%20Ingenier%C3%ADas/1.F%C3%ADsica/13.%20CIN2015A20121.pdf. [Último acceso: 03 mayo 2018].

[11] D. S. Guerrero, «Interacción hombre-robot con vehículos aéreos no tripulados basada en visión,» 14 noviembre 2012. [En línea]. Available: http://www.cs.cinvestav.mx/TesisGraduados/2012/TesisDanielSoto.pdf. [Último acceso: 03 mayo 2018].

[12] K. Cressman, «Mirando desde arriba los sistemas de riego de los arrozales de África,» *ICT Update - Drones para la agricultura*, nº 82, p. 28, 2016.

[13] M. Litter, Farmacología Experimental y Clínica, El Ateneo, 1972.

[14] C. d. S. A. y. Fertilizantes, Guía de productos fitosanitarios para la República Argentina, Mendoza - Argentina, 1993.

[15] I. A. S. Ambrosoni, Normas de producción de la frutilla, 2007|.

[16] I. A. A. Correa, Cultivo de frutiila, en una realidad sin bromuro de metilo en Chile, Santiago de Chile , 2015.

[17] R. A. Española, «Diccionario de la lengua española,» 2019. [En línea]. Available: https://dle.rae.es/srv/fetch?id=ISjyrn6%7CISkwvnu.

[18] M. R. Muñoz, Frutilla, consideraciones productivas y manejo, Villa Alegre, 2012.

[19] C. Morales., «Confección de platabandas o camellones,» de *Manual de manejo agronómico de la frutilla*, Santiago de Chile, 2017, p. 102.

[20] J. Perez, «Definición de:,» 2015. [En línea]. Available: https://definicion.de/pulverizacion/. [Último acceso: 15 diciembre 2018].

[21] B. Ventageneradores, «Como funciona una bomba de agua,» 05 abril 2016. [En línea]. Available: http://www.ventageneradores.net/blog/funcionamiento-como-funciona-una-bomba-agua-motobomba-electrobomba/. [Último acceso: 15 diciembre 2018].

[22] C. d. S. A. y. Fertilizantes, «Tipos de boquillas de pulverización,» 13 septiembre 2016. [En línea]. Available: http://www.casafe.org/tipos-de-boquillas-de-pulverizacion/. [Último acceso: 15 diciembre 2018].

[23] L. Ortega, «Baterías Li-Ion vs LiPo,» 15 agosto 2017. [En línea]. Available: https://www.androidpit.es/bateria-li-ion-vs-lipo-tipos-comparacion. [Último acceso: 14 diciembre 2018].

[24] I. i. e. d. m. digital, «que es drone,» [En línea]. Available: https://iiemd.com/drone/que-es-drone. [Último acceso: 14 diciembre 2018].

[25] wondershare, «Que es un drone y como funciona,» 21 junio 2017. [En línea]. Available: https://filmora.wondershare.com/es/drones/what-is-drone-how-does-it-work.html. [Último acceso: 14 diciembre 2018].

[26] Significados, «Significado de Wifi,» 23 octubre 2018. [En línea]. Available: https://www.significados.com/wifi/. [Último acceso: 14 octubre 2018].

[27] G. Developers, «Conoce Android Studio,» Android Studio, 25 abril 2018. [En línea]. Available: https://developer.android.com/studio/intro/?hl=es-419. [Último acceso: 14 octubre 2018].

[28] DJI, «Especificaciones de Phantom 3 Standar,» DJI , 2018. [En línea]. Available: https://www.dji.com/phantom-3-standard/info. [Último acceso: 10 noviembre 2018].

[29] DJI, «Mavic 2 Especificaciones,» DJI , 2018. [En línea]. Available: https://www.dji.com/mavic-2/info#specs. [Último acceso: 10 noviembre 2018].

[30] Xataca, «Parrot Bebop 2, Análisis: Autonomíay facilidad de vuelo,» 20 abril 2016. [En línea]. Available: https://www.xataka.com/drones/parrot-bebop-2-analisis-autonomia-y-facilidad-de-uso-para-la-renovacion-de-este-divertido-companero-de-vuelo. [Último acceso: 10 noviembre 2018].

[31] Parrot, «Parrot Bebop 2 Power,» 2018. [En línea]. Available: https://www.parrot.com/es/drones/parrot-bebop-2-power-pack-fpv#accesorios. [Último acceso: 10 noviembre 2018].

[32] A. Garcia, «DJILATINO,» 28 enero 2018. [En línea]. Available: https://www.youtube.com/watch?v=47vOhcZxuLg. [Último acceso: 14 diciembre 2019].

[33] DJI, «Phantom 3 Standar,» 2018. [En línea]. Available: https://www.dji.com/phantom-3-standard. [Último acceso: 10 noviembre 2018].

[34] DJI, «Introducción al SDK móvil,» 04 septiembre 2018. [En línea]. Available: https://developer.dji.com/mobile-sdk/documentation/introduction/mobile_sdk_introduction.html. [Último acceso: 14 diciembre 2018].

[35] DJI Developers, «Creando una aplicación MapView y Waypoint,» 20 noviembre 2018. [En línea]. Available: https://developer.dji.com/mobile-sdk/documentation/android-tutorials/GSDemo-Google-Map.html. [Último acceso: 15 diciembre 2018].

[36] Móvil DJI SDK, «Controlador de vuelo,» DJI Developers, 15 mayo 2018. [En línea]. Available: https://developer.dji.com/mobile-sdk/documentation/introduction/component-guide-flightController.html#sensors. [Último acceso: 17 diciembre 2018].

[37] F. DJI, «El controlador de vuelo,» 14 enero 2016. [En línea]. Available: https://forum.dji.com/thread-39810-1-1.html. [Último acceso: 17 diciembre 2019].

[38] DJI Developers, «Control de vuelo,» 13 diciembre 2016. [En línea]. Available: https://developer.dji.com/mobile-

sdk/documentation/introduction/flightController_concepts.html. [Último acceso: 5 enero 2019].

[39] «Rambal Automatización y Robótica,» [En línea]. Available: https://rambal.com/bomba-valvula-solenoide/694-bomba-de-diafragma-6-12v-dc-r385.html. [Último acceso: 12 diciembre 2018].

[40] «ElectroniLab,» [En línea]. Available: https://electronilab.co/tienda/mini-bomba-de-agua-dc-3-12v-rs-360sh/. [Último acceso: 12 diciembre 2018].

[41] M. M. C., «RS-360SH,» [En línea]. Available: http://www.robotstorehk.com/motors/doc/rs_360sh.pdf. [Último acceso: 12 diciembre 2018].

[42] S. S. Co., «Sección 2: Nociones fundamentales acerca de las boquillas de pulverización.,» de *Guía del usuario de Boquillas de Pulverización.*, Illionis, 2004, p. 56

[43] I. Blazquez, «Centro de gravedad,» Multicopters.es, 25 noviembre 2014. [En línea]. Available: http://www.multicopters.es/foro/vbulletin/showthread.php?9726-Centro-de-gravedad. [Último acceso: 14 diciembre 2019].

[44] T. Martín, «Centro de masas,» [En línea]. Available: http://www2.montes.upm.es/dptos/digfa/cfisica/dinamsist/cdm.html. [Último acceso: 15 diciembre 2018].

[45] A. Studio, «Android Studio/Dowload,» 11 diciembre 2018. [En línea]. Available: https://developer.android.com/studio/?hl=es-419#downloads. [Último acceso: 14 diciembre 2018].

[46] E. telégrafo, «A Ecuador llega el nuevo “salvavidas” de Windows,» 03 noviembre 2012. [En línea]. Available: https://www.eltelegrafo.com.ec/noticias/tecnologia/1/a-ecuador-llega-el-nuevo-salvavidas-de-windows. [Último acceso: 16 diciembre 2018].

[47] A. S. Developers, «Android Studio - Descargas,» [En línea]. Available: https://developer.android.com/studio. [Último acceso: 16 diciembre 2018].

[48] D. Pérez, «Las enormes cifras de Google,» El Android libre, 18 mayo 2017. [En línea]. Available: https://elandroidelibre.elespanol.com/2017/05/cifras-de-google-2000-millones-usuarios-android.html. [Último acceso: 16 diciembre 2018].

[49] «Google Play Service,» 31 enero 2019. [En línea]. Available: https://google-play-services.uptodown.com/android. [Último acceso: 28 ferbrero 2019].

[50] T. Telematics, «¿QUÉ ES UNA CLAVE DE API?,» 16 mayo 2018. [En línea]. Available: https://es.support.telematics.tomtom.com/app/answers/detail/a_id/2058/~/%C2%BFqu%C3%A9-es-una-clave-de-api%3F. [Último acceso: 16 diciembre 2019].

[51] A. Developers, «Agregar Mapas,» [En línea]. Available: https://developer.android.com/training/maps. [Último acceso: 12 enero 2019].

[52] G. APIs, «API y servicios credenciales,» Google developers, 2018. [En línea]. Available: https://console.developers.google.com. [Último acceso: 14 diciembre 2018].

[53] P. Gutierrez, «¿Qué son y para qué sirven los hash?,» 15 enero 2013. [En línea]. Available: https://www.genbeta.com/desarrollo/que-son-y-para-que-sirven-los-hash-funciones-de-resumen-y-firmas-digitales. [Último acceso: 13 enero 2019].

[54] desarrolloweb.com, «¿Qué es un Hash?,» [En línea]. Available: https://desarrolloweb.com/wiki/como-agregar-seguridad-a-una-clave-web-con-sha1.html. [Último acceso: 15 enero 2019].

[55] «Gráficos en Android: descripción de OpenGL,» 25 septiembre 2014. [En línea]. Available: https://academiaandroid.com/graficos-en-android-descripcion-opengl/. [Último acceso: 15 enero 2019].

[56] G. developers, «Google Maps Platform/Configuración del proyecto,» 16 julio 2018. [En línea]. Available: https://developers.google.com/maps/documentation/android-sdk/config. [Último acceso: 14 diciembre 2018].

[57] A. developers, «Configurar apps con métodos de más de 64K,» 13 noviembre 2018. [En línea]. Available: https://developer.android.com/studio/build/multidex. [Último acceso: 14 diciembre 2018].

[58] Mangrar, «Introducción a Maven,» Genbeta, 26 julio 2011. [En línea]. Available: https://www.genbeta.com/desarrollo/introduccion-a-maven. [Último acceso: 10 enero 2019].

[59] G. M. Platform, «Maps SDK para la configuración de la utilidad de Android,» 16 julio 2018. [En línea]. Available: https://www.youtube.com/watch?v=Ec7VUcdB-ww. [Último acceso: 14 diciembre 2018].

[61] DJI. Developers, «Administrador de SDK,» DJI, 24 junio 2016. [En línea]. Available: https://developer.dji.com/mobile-sdk/documentation/introduction/sdk-guide-sdkmanager.html. [Último acceso: 14 diciembre 2018].

ANEXOS

ANEXO A: CODIFICACIÓN DEL ARCHIVO *"androidmanifest.xml"* DE ANDROID STUDIO.

El archivo androidmanifest.xml contiene
las configuraciones de los
servicios de Google Play,
además, es en este archivo
donde se agrega la clave de aplicación,
API Google Maps y SHA-1

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.dji.GSDemo.GoogleMap"
    android:versionCode="1"
    android:versionName="1.0">
    <uses-permission android:name="android.permission.BLUETOOTH" />
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
    <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.READ_PHONE_STATE" />
    <uses-permission
android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission   android:name="android.permission.ACCESS_FINE_LOCATION"
/>
    <uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission
android:name="android.permission.CHANGE_CONFIGURATION" />
    <uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW"
/>
    <uses-permission android:name="android.permission.WRITE_SETTINGS" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission
android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS" />

    <uses-feature
        android:name="android.hardware.usb.accessory"
        android:required="true" />
    <uses-feature
        android:name="android.hardware.usb.host"
        android:required="false" />
    <uses-feature
        android:glEsVersion="0x00020000"
        android:required="true" />

    <application
        android:name=".MApplication"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">

        <!-- DJI SDK -->
        <uses-library android:name="com.android.future.usb.accessory" />

        <meta-data
            android:name="com.dji.sdk.API_KEY"
```

```
        android:value="df39532ccefbd55c74ed68d0" /> //Ingrese aqui su clave de
applicacion
      <meta-data
        android:name="com.google.android.geo.API_KEY"
        android:value="AIzaSyAsLKU_3gW0qVUdlAkaxtT01eIA-P6HCwY"          />
//TODO: Enter your Google API Key here
      <meta-data
        android:name="com.google.android.gms.version"
        android:value="@integer/google_play_services_version" />

      <activity
        android:name="dji.sdk.sdkmanager.DJIAoaControllerActivity"
        android:theme="@android:style/Theme.Translucent">
        <intent-filter>
          <action
android:name="android.hardware.usb.action.USB_ACCESSORY_ATTACHED" />
        </intent-filter>

        <meta-data
android:name="android.hardware.usb.action.USB_ACCESSORY_ATTACHED"
          android:resource="@xml/accessory_filter" />
      </activity>
      <service android:name="dji.sdk.sdkmanager.DJIGlobalService"></service>

      <!-- DJI SDK -->
      <activity
        android:name=".ConnectionActivity"
        android:configChanges="orientation|screenSize|keyboardHidden|keyboard"
        android:label="@string/app_name"
        android:screenOrientation="portrait">
        <intent-filter>
          <action android:name="android.intent.action.MAIN" />
          <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
      </activity>
      <activity android:name=".MainActivity"
        android:screenOrientation="portrait"></activity>
    </application>
</manifest>
```

ANEXO B: CODIFICACIÓN DEL ARCHIVO *"activity_main.xml"* DE ANDROID STUDIO.

El archivo activity_main.xml
muestra el código del diseño
de la actividad principal, como
la creación de un LinearLayout,
los botones y el fragmento de vista de mapa.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#FFFFFF"
    android:orientation="vertical"
    tools:context="com.dji.GSDemo.GoogleMap.MainActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <TextView
            android:id="@+id/ConnectStatusTextView"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:gravity="center"
            android:text="GSDemo"
            android:textColor="#000000"
            android:textSize="21sp" />
    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <Button
            android:id="@+id/locate"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Locate" />
```

```
    <Button
        android:id="@+id/add"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Add" />

    <Button
        android:id="@+id/clear"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Clear" />
</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">

    <Button
        android:id="@+id/config"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="0.9"
        android:text="Config" />

    <Button
        android:id="@+id/upload"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="0.9"
        android:text="Upload" />

    <Button
```

```
        android:id="@+id/start"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Start" />

    <Button
        android:id="@+id/stop"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Stop" />
  </LinearLayout>

  <fragment
    android:id="@+id/map"
    class="com.google.android.gms.maps.SupportMapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />

</LinearLayout>
```

ANEXO C: CÓDIGO DE PROGRAMACIÓN DEL ARCHIVO *"dialog_waypointsetting.xml"* DE ANDROID STUDIO.

El archivo dialog_waypointsetting.xml
Crea el diseño de interfaz que ayudará
a la configuración de una vista de texto
para ingresar la altitud de vuelo, tres
grupos de botones de radio para seleccionar
velocidad, acción después de finalizado y modo de vuelo

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:orientation="vertical"
   android:layout_marginBottom="10dp"
   android:layout_marginTop="10dp"
   android:layout_marginLeft="10dp"
   android:layout_marginRight="10dp">

   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Waypoint Configuration"
      android:layout_gravity="center_horizontal">
   </TextView>

   <LinearLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:orientation="horizontal"
      android:layout_marginBottom="10dp"
      android:layout_marginTop="10dp"
      android:layout_marginLeft="10dp"
      android:layout_marginRight="10dp">
      <TextView
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="Altitude:">
      </TextView>
      <EditText
         android:id="@+id/altitude"
         android:layout_width="40dp"
         android:layout_height="wrap_content">
      </EditText>
```

```
</LinearLayout>

<LinearLayout
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:orientation="horizontal"
   android:layout_marginBottom="10dp"
   android:layout_marginTop="10dp"
   android:layout_marginLeft="10dp"
   android:layout_marginRight="10dp">
   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Speed:">
   </TextView>
   <RadioGroup
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:orientation="horizontal"
      android:id="@+id/speed"
      android:layout_gravity="center_horizontal">
      <RadioButton
         android:id="@+id/lowSpeed"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="Low"/>
      <RadioButton
         android:id="@+id/MidSpeed"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="Mid"/>
      <RadioButton
         android:id="@+id/HighSpeed"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
```

```
            android:text="High"/>
    </RadioGroup>
</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:layout_marginBottom="10dp"
    android:layout_marginTop="10dp"
    android:layout_marginLeft="10dp"
    android:layout_marginRight="10dp">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Action After Finished:"/>
</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal" >
    <RadioGroup
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:id="@+id/actionAfterFinished"
        android:layout_gravity="center_horizontal">
        <RadioButton
            android:id="@+id/finishNone"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textSize="5pt"
            android:text="None"/>
        <RadioButton
```

```
            android:id="@+id/finishGoHome"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textSize="5pt"
            android:text="GoHome"/>
        <RadioButton
            android:id="@+id/finishAutoLanding"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textSize="5pt"
            android:text="AutoLand"/>
        <RadioButton
            android:id="@+id/finishToFirst"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textSize="5pt"
            android:text="BackTo 1st"/>
    </RadioGroup>
</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:layout_marginBottom="10dp"
    android:layout_marginTop="10dp"
    android:layout_marginLeft="10dp"
    android:layout_marginRight="10dp">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Heading:"/>
</LinearLayout>

<LinearLayout
```

```
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal" >
        <RadioGroup
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:orientation="horizontal"
            android:id="@+id/heading"
            android:layout_gravity="center_horizontal">
            <RadioButton
                android:id="@+id/headingNext"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:textSize="5pt"
                android:text="Auto"/>
            <RadioButton
                android:id="@+id/headingInitDirec"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:textSize="5pt"
                android:text="Initial"/>
            <RadioButton
                android:id="@+id/headingRC"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:textSize="5pt"
                android:text="RC Control"/>
            <RadioButton
               android:id="@+id/headingWP"
               android:layout_width="wrap_content"
               android:layout_height="wrap_content"
               android:textSize="5pt"
               android:text="Use Waypoint"/>
        </RadioGroup>
    </LinearLayout>
</LinearLayout>
```

ANEXO D: CÓDIGO DE PROGRAMACIÓN DEL ARCHIVO "*MainActivity.java"* DE ANDROID STUDIO.

El archive MainActity,java
contiene la codificación de las
acciones que realizarán cada uno
de los botones, áreas de texto y
botones de radio dentro de la aplicación.

```
package com.dji.GSDemo.GoogleMap;

import android.Manifest;
import android.app.AlertDialog;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.IntentFilter;
import android.os.Build;
import android.support.annotation.Nullable;
import android.support.v4.app.ActivityCompat;
import android.support.v4.app.FragmentActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.LinearLayout;
import android.widget.RadioGroup;
import android.widget.TextView;
import android.widget.Toast;
import com.google.android.gms.maps.CameraUpdate;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;
import com.google.android.gms.maps.model.MarkerOptions;
import java.util.ArrayList;
import java.util.List;
import java.util.Map;
import java.util.concurrent.ConcurrentHashMap;

import dji.common.flightcontroller.FlightControllerState;
import dji.common.mission.waypoint.Waypoint;
import dji.common.mission.waypoint.WaypointMission;
import dji.common.mission.waypoint.WaypointMissionDownloadEvent;
import dji.common.mission.waypoint.WaypointMissionExecutionEvent;
import dji.common.mission.waypoint.WaypointMissionFinishedAction;
import dji.common.mission.waypoint.WaypointMissionFlightPathMode;
import dji.common.mission.waypoint.WaypointMissionHeadingMode;
import dji.common.mission.waypoint.WaypointMissionUploadEvent;
import dji.common.useraccount.UserAccountState;
import dji.common.util.CommonCallbacks;
```

```
import dji.sdk.base.BaseProduct;
import dji.sdk.flightcontroller.FlightController;
import dji.common.error.DJIError;
import dji.sdk.mission.waypoint.WaypointMissionOperator;
import dji.sdk.mission.waypoint.WaypointMissionOperatorListener;
import dji.sdk.products.Aircraft;
import dji.sdk.sdkmanager.DJISDKManager;
import dji.sdk.useraccount.UserAccountManager;

public class MainActivity extends FragmentActivity implements View.OnClickListener,
GoogleMap.OnMapClickListener, OnMapReadyCallback {

    protected static final String TAG = "GSDemoActivity";

    private GoogleMap gMap;

    private Button locate, add, clear;
    private Button config, upload, start, stop;

    private boolean isAdd = false;

    private double droneLocationLat = 181, droneLocationLng = 181;
    private final Map<Integer, Marker> mMarkers = new ConcurrentHashMap<Integer,
Marker>();
    private Marker droneMarker = null;

    private float altitude = 100.0f;
    private float mSpeed = 10.0f;

    private List<Waypoint> waypointList = new ArrayList<>();

    public static WaypointMission.Builder waypointMissionBuilder;
    private FlightController mFlightController;
    private WaypointMissionOperator instance;
    private WaypointMissionFinishedAction mFinishedAction =
WaypointMissionFinishedAction.NO_ACTION;
    private WaypointMissionHeadingMode mHeadingMode =
WaypointMissionHeadingMode.AUTO;

    @Override
    protected void onResume(){
        super.onResume();
        initFlightController();
    }

    @Override
```

```
protected void onPause(){
    super.onPause();
}

@Override
protected void onDestroy(){
    unregisterReceiver(mReceiver);
    removeListener();
    super.onDestroy();
}

/**
 * @Description : RETURN Button RESPONSE FUNCTION
 */
public void onReturn(View view){
    Log.d(TAG, "onReturn");
    this.finish();
}

private void setResultToToast(final String string){
    MainActivity.this.runOnUiThread(new Runnable() {
        @Override
        public void run() {
            Toast.makeText(MainActivity.this, string, Toast.LENGTH_SHORT).show();
        }
    });
}

private void initUI() {

    locate = (Button) findViewById(R.id.locate);
    add = (Button) findViewById(R.id.add);
    clear = (Button) findViewById(R.id.clear);
    config = (Button) findViewById(R.id.config);
    upload = (Button) findViewById(R.id.upload);
    start = (Button) findViewById(R.id.start);
    stop = (Button) findViewById(R.id.stop);

    locate.setOnClickListener(this);
    add.setOnClickListener(this);
    clear.setOnClickListener(this);
    config.setOnClickListener(this);
    upload.setOnClickListener(this);
    start.setOnClickListener(this);
    stop.setOnClickListener(this);
```

```
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // When the compile and target version is higher than 22, please request the
        // following permissions at runtime to ensure the
        // SDK work well.
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            ActivityCompat.requestPermissions(this,
                    new String[]{Manifest.permission.WRITE_EXTERNAL_STORAGE,
Manifest.permission.VIBRATE,
                            Manifest.permission.INTERNET,
Manifest.permission.ACCESS_WIFI_STATE,
                            Manifest.permission.WAKE_LOCK,
Manifest.permission.ACCESS_COARSE_LOCATION,
                            Manifest.permission.ACCESS_NETWORK_STATE,
Manifest.permission.ACCESS_FINE_LOCATION,
                            Manifest.permission.CHANGE_WIFI_STATE,
Manifest.permission.MOUNT_UNMOUNT_FILESYSTEMS,
                            Manifest.permission.READ_EXTERNAL_STORAGE,
Manifest.permission.SYSTEM_ALERT_WINDOW,
                            Manifest.permission.READ_PHONE_STATE,
                    }
                    , 1);
        }

        setContentView(R.layout.activity_main);

        IntentFilter filter = new IntentFilter();
        filter.addAction(DJIDemoApplication.FLAG_CONNECTION_CHANGE);
        registerReceiver(mReceiver, filter);

        initUI();

        SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager()
                .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);

        addListener();

    }

    protected BroadcastReceiver mReceiver = new BroadcastReceiver() {
```

```
        @Override
        public void onReceive(Context context, Intent intent) {
            onProductConnectionChange();
        }
    };

    private void onProductConnectionChange()
    {
        initFlightController();
        loginAccount();
    }

    private void loginAccount(){

        UserAccountManager.getInstance().logIntoDJIUserAccount(this,
                new CommonCallbacks.CompletionCallbackWith<UserAccountState>() {
                    @Override
                    public void onSuccess(final UserAccountState userAccountState) {
                        Log.e(TAG, "Login Success");
                    }
                    @Override
                    public void onFailure(DJIError error) {
                        setResultToToast("Login Error:"
                                + error.getDescription());
                    }
                });
    }

    private void initFlightController() {

        BaseProduct product = DJIDemoApplication.getProductInstance();
        if (product != null && product.isConnected()) {
            if (product instanceof Aircraft) {
                mFlightController = ((Aircraft) product).getFlightController();
            }
        }

        if (mFlightController != null) {
            mFlightController.setStateCallback(new FlightControllerState.Callback() {

                @Override
                public void onUpdate(FlightControllerState djiFlightControllerCurrentState) {
                    droneLocationLat =
djiFlightControllerCurrentState.getAircraftLocation().getLatitude();
                    droneLocationLng =
```

```
djiFlightControllerCurrentState.getAircraftLocation().getLongitude();
                updateDroneLocation();
            }
        });
    }
}

//Add Listener for WaypointMissionOperator
private void addListener() {
    if (getWaypointMissionOperator() != null){
        getWaypointMissionOperator().addListener(eventNotificationListener);
    }
}

private void removeListener() {
    if (getWaypointMissionOperator() != null) {
        getWaypointMissionOperator().removeListener(eventNotificationListener);
    }
}

private WaypointMissionOperatorListener eventNotificationListener = new
WaypointMissionOperatorListener() {
    @Override
    public void onDownloadUpdate(WaypointMissionDownloadEvent downloadEvent)
{

    }

    @Override
    public void onUploadUpdate(WaypointMissionUploadEvent uploadEvent) {

    }

    @Override
    public void onExecutionUpdate(WaypointMissionExecutionEvent executionEvent)
{

    }

    @Override
    public void onExecutionStart() {

    }

    @Override
    public void onExecutionFinish(@Nullable final DJIError error) {
```

```
            setResultToToast("Execution finished: " + (error == null ? "Success!" :
error.getDescription()));
        }
    };

    public WaypointMissionOperator getWaypointMissionOperator() {
        if (instance == null) {
            if (DJISDKManager.getInstance().getMissionControl() != null){
                instance =
DJISDKManager.getInstance().getMissionControl().getWaypointMissionOperator();
            }
        }
        return instance;
    }

    private void setUpMap() {
        gMap.setOnMapClickListener(this);// add the listener for click for amap object

    }

    @Override
    public void onMapClick(LatLng point) {
        if (isAdd == true){
            markWaypoint(point);
            Waypoint mWaypoint = new Waypoint(point.latitude, point.longitude, altitude);
            //Add Waypoints to Waypoint arraylist;
            if (waypointMissionBuilder != null) {
                waypointList.add(mWaypoint);

waypointMissionBuilder.waypointList(waypointList).waypointCount(waypointList.size(
));
            }else
            {
                waypointMissionBuilder = new WaypointMission.Builder();
                waypointList.add(mWaypoint);

waypointMissionBuilder.waypointList(waypointList).waypointCount(waypointList.size(
));
            }
        }else{
            setResultToToast("Cannot Add Waypoint");
        }
    }

    public static boolean checkGpsCoordination(double latitude, double longitude) {
        return (latitude > -90 && latitude < 90 && longitude > -180 && longitude < 180)
```

```
&& (latitude != 0f && longitude != 0f);
    }

    // Update the drone location based on states from MCU.
    private void updateDroneLocation(){

        LatLng pos = new LatLng(droneLocationLat, droneLocationLng);
        //Create MarkerOptions object
        final MarkerOptions markerOptions = new MarkerOptions();
        markerOptions.position(pos);
        markerOptions.icon(BitmapDescriptorFactory.fromResource(R.drawable.aircraft));

        runOnUiThread(new Runnable() {
            @Override
            public void run() {
                if (droneMarker != null) {
                    droneMarker.remove();
                }

                if (checkGpsCoordination(droneLocationLat, droneLocationLng)) {
                    droneMarker = gMap.addMarker(markerOptions);
                }
            }
        });
    }

    private void markWaypoint(LatLng point){
        //Create MarkerOptions object
        MarkerOptions markerOptions = new MarkerOptions();
        markerOptions.position(point);

markerOptions.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.
HUE_BLUE));
        Marker marker = gMap.addMarker(markerOptions);
        mMarkers.put(mMarkers.size(), marker);
    }

    @Override
    public void onClick(View v) {
        switch (v.getId()) {
            case R.id.locate:{
                updateDroneLocation();
                cameraUpdate(); // Locate the drone's place
                break;
            }
            case R.id.add:{
```

```
            enableDisableAdd();
            break;
        }
        case R.id.clear:{
            runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    gMap.clear();
                }

            });
            waypointList.clear();
            waypointMissionBuilder.waypointList(waypointList);
            updateDroneLocation();
            break;
        }
        case R.id.config:{
            showSettingDialog();
            break;
        }
        case R.id.upload:{
            uploadWayPointMission();
            break;
        }
        case R.id.start:{
            startWaypointMission();
            break;
        }
        case R.id.stop:{
            stopWaypointMission();
            break;
        }
        default:
            break;
    }
}

private void cameraUpdate(){
    LatLng pos = new LatLng(droneLocationLat, droneLocationLng);
    float zoomlevel = (float) 18.0;
    CameraUpdate cu = CameraUpdateFactory.newLatLngZoom(pos, zoomlevel);
    gMap.moveCamera(cu);

}

private void enableDisableAdd(){
```

```
        if (isAdd == false) {
            isAdd = true;
            add.setText("Exit");
        }else{
            isAdd = false;
            add.setText("Add");
        }
    }

    private void showSettingDialog(){
        LinearLayout wayPointSettings =
(LinearLayout)getLayoutInflater().inflate(R.layout.dialog_waypointsetting, null);

        final TextView wpAltitude_TV = (TextView)
wayPointSettings.findViewById(R.id.altitude);
        RadioGroup speed_RG = (RadioGroup)
wayPointSettings.findViewById(R.id.speed);
        RadioGroup actionAfterFinished_RG = (RadioGroup)
wayPointSettings.findViewById(R.id.actionAfterFinished);
        RadioGroup heading_RG = (RadioGroup)
wayPointSettings.findViewById(R.id.heading);

        speed_RG.setOnCheckedChangeListener(new
RadioGroup.OnCheckedChangeListener(){

            @Override
            public void onCheckedChanged(RadioGroup group, int checkedId) {
                if (checkedId == R.id.lowSpeed){
                    mSpeed = 3.0f;
                } else if (checkedId == R.id.MidSpeed){
                    mSpeed = 5.0f;
                } else if (checkedId == R.id.HighSpeed){
                    mSpeed = 10.0f;
                }
            }

        });

        actionAfterFinished_RG.setOnCheckedChangeListener(new
RadioGroup.OnCheckedChangeListener() {

            @Override
            public void onCheckedChanged(RadioGroup group, int checkedId) {
                Log.d(TAG, "Select finish action");
                if (checkedId == R.id.finishNone){
                    mFinishedAction = WaypointMissionFinishedAction.NO_ACTION;
```

```
            } else if (checkedId == R.id.finishGoHome){
                mFinishedAction = WaypointMissionFinishedAction.GO_HOME;
            } else if (checkedId == R.id.finishAutoLanding){
                mFinishedAction = WaypointMissionFinishedAction.AUTO_LAND;
            } else if (checkedId == R.id.finishToFirst){
                mFinishedAction =
WaypointMissionFinishedAction.GO_FIRST_WAYPOINT;
            }
        }
    });

    heading_RG.setOnCheckedChangeListener(new
RadioGroup.OnCheckedChangeListener() {

        @Override
        public void onCheckedChanged(RadioGroup group, int checkedId) {
            Log.d(TAG, "Select heading");

            if (checkedId == R.id.headingNext) {
                mHeadingMode = WaypointMissionHeadingMode.AUTO;
            } else if (checkedId == R.id.headingInitDirec) {
                mHeadingMode =
WaypointMissionHeadingMode.USING_INITIAL_DIRECTION;
            } else if (checkedId == R.id.headingRC) {
                mHeadingMode =
WaypointMissionHeadingMode.CONTROL_BY_REMOTE_CONTROLLER;
            } else if (checkedId == R.id.headingWP) {
                mHeadingMode =
WaypointMissionHeadingMode.USING_WAYPOINT_HEADING;
            }
        }
    });

    new AlertDialog.Builder(this)
            .setTitle("")
            .setView(wayPointSettings)
            .setPositiveButton("Finish",new DialogInterface.OnClickListener(){
                public void onClick(DialogInterface dialog, int id) {

                    String altitudeString = wpAltitude_TV.getText().toString();
                    altitude = Integer.parseInt(nulltoIntegerDefalt(altitudeString));
                    Log.e(TAG,"altitude "+altitude);
                    Log.e(TAG,"speed "+mSpeed);
                    Log.e(TAG, "mFinishedAction "+mFinishedAction);
                    Log.e(TAG, "mHeadingMode "+mHeadingMode);
                    configWayPointMission();
```

```
                }

            })
            .setNegativeButton("Cancel", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                    dialog.cancel();
                }

            })
            .create()
            .show();
    }

    String nulltoIntegerDefalt(String value){
        if(!isIntValue(value)) value="0";
        return value;
    }

    boolean isIntValue(String val)
    {
        try {
            val=val.replace(" ","");
            Integer.parseInt(val);
        } catch (Exception e) {return false;}
        return true;
    }

    private void configWayPointMission(){

        if (waypointMissionBuilder == null){

            waypointMissionBuilder = new
WaypointMission.Builder().finishedAction(mFinishedAction)
                    .headingMode(mHeadingMode)
                    .autoFlightSpeed(mSpeed)
                    .maxFlightSpeed(mSpeed)
                    .flightPathMode(WaypointMissionFlightPathMode.NORMAL);

        }else
        {
            waypointMissionBuilder.finishedAction(mFinishedAction)
                    .headingMode(mHeadingMode)
                    .autoFlightSpeed(mSpeed)
                    .maxFlightSpeed(mSpeed)
                    .flightPathMode(WaypointMissionFlightPathMode.NORMAL);
```

```
        }

        if (waypointMissionBuilder.getWaypointList().size() > 0){

            for (int i=0; i< waypointMissionBuilder.getWaypointList().size(); i++){
                waypointMissionBuilder.getWaypointList().get(i).altitude = altitude;
            }

            setResultToToast("Set Waypoint attitude successfully");
        }

        DJIError error =
getWaypointMissionOperator().loadMission(waypointMissionBuilder.build());
        if (error == null) {
            setResultToToast("loadWaypoint succeeded");
        } else {
            setResultToToast("loadWaypoint failed " + error.getDescription());
        }
    }

    private void uploadWayPointMission(){

        getWaypointMissionOperator().uploadMission(new
CommonCallbacks.CompletionCallback() {
            @Override
            public void onResult(DJIError error) {
                if (error == null) {
                    setResultToToast("Mission upload successfully!");
                } else {
                    setResultToToast("Mission upload failed, error: " + error.getDescription() +
" retrying...");
                    getWaypointMissionOperator().retryUploadMission(null);
                }
            }
        });

    }

    private void startWaypointMission(){

        getWaypointMissionOperator().startMission(new
CommonCallbacks.CompletionCallback() {
            @Override
            public void onResult(DJIError error) {
                setResultToToast("Mission Start: " + (error == null ? "Successfully" :
error.getDescription()));
```

```
            }
        });
    }

    private void stopWaypointMission(){

        getWaypointMissionOperator().stopMission(new
CommonCallbacks.CompletionCallback() {
            @Override
            public void onResult(DJIError error) {
                setResultToToast("Mission Stop: " + (error == null ? "Successfully" :
error.getDescription()));
            }
        });

    }

    @Override
    public void onMapReady(GoogleMap googleMap) {
        if (gMap == null) {
            gMap = googleMap;
            setUpMap();
        }

        LatLng shenzhen = new LatLng(-1.230298, -78.604545);
        gMap.addMarker(new MarkerOptions().position(shenzhen).title("Marker in
Ambato"));
        gMap.moveCamera(CameraUpdateFactory.newLatLng(shenzhen));
    }

}
```

ANEXO E: CÓDIGO DE PROGRAMACIÓN DEL ARCHIVO "*ConnectionActivity.java"* DE ANDROID STUDIO.

El archivo ConnectionActivity.java
muestra el código con el cual se realiza
la verificación de comunicación que se canaliza
entre el dispositivo móvil y el UAV

```
package com.dji.GSDemo.GoogleMap;

import android.Manifest;
import android.app.Activity;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.content.pm.PackageManager;
import android.os.AsyncTask;
import android.os.Build;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;
import java.util.ArrayList;
import java.util.List;
import java.util.concurrent.atomic.AtomicBoolean;
import dji.common.error.DJIError;
import dji.common.error.DJISDKError;
import dji.log.DJILog;
import dji.sdk.base.BaseComponent;
import dji.sdk.base.BaseProduct;
import dji.sdk.products.Aircraft;
import dji.sdk.sdkmanager.DJISDKManager;

public class ConnectionActivity extends Activity implements View.OnClickListener {

    private static final String TAG = ConnectionActivity.class.getName();
```

```
private TextView mTextConnectionStatus;
private TextView mTextProduct;
private TextView mVersionTv;
private Button mBtnOpen;
private static final String[] REQUIRED_PERMISSION_LIST = new String[]{
        Manifest.permission.VIBRATE,
        Manifest.permission.INTERNET,
        Manifest.permission.ACCESS_WIFI_STATE,
        Manifest.permission.WAKE_LOCK,
        Manifest.permission.ACCESS_COARSE_LOCATION,
        Manifest.permission.ACCESS_NETWORK_STATE,
        Manifest.permission.ACCESS_FINE_LOCATION,
        Manifest.permission.CHANGE_WIFI_STATE,
        Manifest.permission.WRITE_EXTERNAL_STORAGE,
        Manifest.permission.BLUETOOTH,
        Manifest.permission.BLUETOOTH_ADMIN,
        Manifest.permission.READ_EXTERNAL_STORAGE,
        Manifest.permission.READ_PHONE_STATE,
};
private List<String> missingPermission = new ArrayList<>();
private AtomicBoolean isRegistrationInProgress = new AtomicBoolean(false);
private static final int REQUEST_PERMISSION_CODE = 12345;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    checkAndRequestPermissions();
    setContentView(R.layout.activity_connection);

    initUI();

    // Register the broadcast receiver for receiving the device connection's changes.
    IntentFilter filter = new IntentFilter();
    filter.addAction(DJIDemoApplication.FLAG_CONNECTION_CHANGE);
    registerReceiver(mReceiver, filter);
```

```
    }

    /**
     * Checks if there is any missing permissions, and
     * requests runtime permission if needed.
     */
    private void checkAndRequestPermissions() {
        // Check for permissions
        for (String eachPermission : REQUIRED_PERMISSION_LIST) {
            if (ContextCompat.checkSelfPermission(this, eachPermission) !=
PackageManager.PERMISSION_GRANTED) {
                missingPermission.add(eachPermission);
            }
        }
        // Request for missing permissions
        if (!missingPermission.isEmpty() && Build.VERSION.SDK_INT >=
Build.VERSION_CODES.M) {
            ActivityCompat.requestPermissions(this,
                    missingPermission.toArray(new String[missingPermission.size()]),
                    REQUEST_PERMISSION_CODE);
        }

    }


    /**
     * Result of runtime permission request
     */
    @Override
    public void onRequestPermissionsResult(int requestCode,
                                           @NonNull String[] permissions,
                                           @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
        // Check for granted permission and remove from missing list
        if (requestCode == REQUEST_PERMISSION_CODE) {
```

```
            for (int i = grantResults.length - 1; i >= 0; i--) {
                if (grantResults[i] == PackageManager.PERMISSION_GRANTED) {
                    missingPermission.remove(permissions[i]);
                }
            }
        }
        // If there is enough permission, we will start the registration
        if (missingPermission.isEmpty()) {
            startSDKRegistration();
        } else {
            showToast("Missing permissions!!!");
        }
    }

    private void startSDKRegistration() {
        if (isRegistrationInProgress.compareAndSet(false, true)) {
            AsyncTask.execute(new Runnable() {
                @Override
                public void run() {
                    showToast( "registering, pls wait...");
                    DJISDKManager.getInstance().registerApp(getApplicationContext(),     new
DJISDKManager.SDKManagerCallback() {
                        @Override
                        public void onRegister(DJIError djiError) {
                            if (djiError == DJISDKError.REGISTRATION_SUCCESS) {
                                DJILog.e("App                                                registration",
DJISDKError.REGISTRATION_SUCCESS.getDescription());
                                DJISDKManager.getInstance().startConnectionToProduct();
                                showToast("Register Success");
                            } else {
                                showToast( "Register sdk fails, check network is available");
                            }
                            Log.v(TAG, djiError.getDescription());
                        }
```

```
        @Override
        public void onProductDisconnect() {
            Log.d(TAG, "onProductDisconnect");
            showToast("Product Disconnected");

        }
        @Override
        public void onProductConnect(BaseProduct baseProduct) {
            Log.d(TAG, String.format("onProductConnect newProduct:%s",
baseProduct));
            showToast("Product Connected");

        }
        @Override
        public void onComponentChange(BaseProduct.ComponentKey
componentKey, BaseComponent oldComponent,
                                      BaseComponent newComponent) {

            if (newComponent != null) {
                newComponent.setComponentListener(new
BaseComponent.ComponentListener() {

                    @Override
                    public void onConnectivityChange(boolean isConnected) {
                        Log.d(TAG, "onComponentConnectivityChanged: " +
isConnected);
                    }
                });
            }
            Log.d(TAG,
                    String.format("onComponentChange key:%s, oldComponent:%s,
newComponent:%s",
                            componentKey,
                            oldComponent,
                            newComponent));
```

```
                }
            });
        }
    });
  }
}

@Override
public void onResume() {
  Log.e(TAG, "onResume");
  super.onResume();
}

@Override
public void onPause() {
  Log.e(TAG, "onPause");
  super.onPause();
}

@Override
public void onStop() {
  Log.e(TAG, "onStop");
  super.onStop();
}

public void onReturn(View view){
  Log.e(TAG, "onReturn");
  this.finish();
}

@Override
protected void onDestroy() {
  Log.e(TAG, "onDestroy");
  unregisterReceiver(mReceiver);
```

```
        super.onDestroy();
    }

    private void initUI() {

        mTextConnectionStatus = (TextView) findViewById(R.id.text_connection_status);
        mTextProduct = (TextView) findViewById(R.id.text_product_info);
        mBtnOpen = (Button) findViewById(R.id.btn_open);
        mBtnOpen.setOnClickListener(this);
        mBtnOpen.setEnabled(false);

        mVersionTv = (TextView) findViewById(R.id.textView2);
        mVersionTv.setText(getResources().getString(R.string.sdk_version,
DJISDKManager.getInstance().getSDKVersion()));
    }

    protected BroadcastReceiver mReceiver = new BroadcastReceiver() {

        @Override
        public void onReceive(Context context, Intent intent) {
            refreshSDKRelativeUI();
        }
    };

    private void refreshSDKRelativeUI() {
        BaseProduct mProduct = DJIDemoApplication.getProductInstance();

        if (null != mProduct && mProduct.isConnected()) {
            Log.v(TAG, "refreshSDK: True");
            mBtnOpen.setEnabled(true);

            String str = mProduct instanceof Aircraft ? "DJIAircraft" : "DJIHandHeld";
            mTextConnectionStatus.setText("Status: " + str + " connected");

            if (null != mProduct.getModel()) {
```

```
            mTextProduct.setText("" + mProduct.getModel().getDisplayName());
        } else {
            mTextProduct.setText(R.string.product_information);
        }

    } else {
        Log.v(TAG, "refreshSDK: False");
        mBtnOpen.setEnabled(false);

        mTextProduct.setText(R.string.product_information);
        mTextConnectionStatus.setText(R.string.connection_loose);
    }
}

@Override
public void onClick(View v) {
    switch (v.getId()) {

        case R.id.btn_open: {
            Intent intent = new Intent(this, MainActivity.class);
            startActivity(intent);
            break;
        }
        default:
            break;
    }
}

private void showToast(final String toastMsg) {
    runOnUiThread(new Runnable() {
        @Override
        public void run() {
            Toast.makeText(getApplicationContext(),                    toastMsg,
Toast.LENGTH_LONG).show();
        }       });    }}
```