

UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

CARRERA DE TELECOMUNICACIONES

Tema:

SISTEMA DE NAVEGACIÓN AUTÓNOMA DE UN UAV PARA VIDEOVIGILANCIA Y OPERACIONES DEL ESPACIO AÉREO DE LA UNIVERSIDAD TÉCNICA DE AMBATO

Trabajo de titulación modalidad Proyecto de Investigación, presentado previo a la obtención del título de Ingeniero en Telecomunicaciones

ÁREA: Comunicaciones

LÍNEA DE INVESTIGACIÓN: Tecnología de la información y Sistema de control

AUTORES: León Armijo Jean Carlos
Tualombo Nuñez Ricardo Feliciano

TUTOR: Ing. Franklin Wilfrido Salazar Logroño Mg.

Ambato - Ecuador

febrero – 2025

APROBACIÓN DEL TUTOR

En calidad de tutor del trabajo de titulación con el tema: SISTEMA DE NAVEGACIÓN AUTÓNOMA DE UN UAV PARA VIDEOVIGILANCIA Y OPERACIONES DEL ESPACIO AÉREO DE LA UNIVERSIDAD TÉCNICA DE AMBATO, desarrollado bajo la modalidad Proyecto de Investigación por los señores Jean Carlos León Armijo y Ricardo Feliciano Tualombo Nuñez, estudiantes de la Carrera de Telecomunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, me permito indicar que el estudiante ha sido tutorado durante todo el desarrollo del trabajo hasta su conclusión, de acuerdo a lo dispuesto en el Artículo 17 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.3 del instructivo del reglamento referido.

Ambato, febrero 2025.

Firmado electrónicamente por:
FRANKLIN WILFRIDO SALAZAR LOGRONO

Ing. Franklin Wilfrido Salazar Logroño, Mg.

TUTOR

AUTORÍA

El presente trabajo de titulación con el tema: SISTEMA DE NAVEGACIÓN AUTÓNOMA DE UN UAV PARA VIDEOVIGILANCIA Y OPERACIONES DEL ESPACIO AÉREO DE LA UNIVERSIDAD TÉCNICA DE AMBATO es absolutamente original, auténtico y personal y ha observado los preceptos establecidos en la Disposición General Quinta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato. En tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, febrero 2025.



Jean Carlos León Armijo
C.C. 1719024257
AUTOR



Ricardo Feliciano Tualombo Nuñez
C.C. 1804707758
AUTOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato para que reproduzca total o parcialmente este trabajo de titulación dentro de las regulaciones legales e institucionales correspondientes. Además, cedo todos mis derechos de autor a favor de la institución con el propósito de su difusión pública, por lo tanto, autorizo su publicación en el repositorio virtual institucional como un documento disponible para la lectura y uso con fines académicos e investigativos de acuerdo con la Disposición General Cuarta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato.

Ambato, febrero 2025.



Jean Carlos León Armijo
C.C. 1719024257
AUTOR



Ricardo Feliciano Tualombo Nuñez
C.C. 1804707758
AUTOR

APROBACIÓN DEL TRIBUNAL DE GRADO

En calidad de par calificador del informe final del trabajo de titulación presentado por los señores Jean Carlos León Armijo y Ricardo Feliciano Tualombo Nuñez, estudiantes de la Carrera de Telecomunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, bajo la Modalidad Proyecto de Investigación, titulado SISTEMA DE NAVEGACIÓN AUTÓNOMA DE UN UAV PARA VIDEOVIGILANCIA Y OPERACIONES DEL ESPACIO AÉREO DE LA UNIVERSIDAD TÉCNICA DE AMBATO, nos permitimos informar que el trabajo ha sido revisado y calificado de acuerdo al Artículo 19 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.4 del instructivo del reglamento referido. Para cuya constancia suscribimos, conjuntamente con el señor Presidente del Tribunal.

Ambato, febrero 2025.

Firmado electrónicamente por:
FRANKLIN OSWALDO
MAYORGA MAYORGA

Ing. Franklin Oswaldo Mayorga Mayorga, Mg.
PRESIDENTE DEL TRIBUNAL

Firmado electrónicamente por:
VICTOR SANTIAGO
MANZANO VILLAFUERTE

Ing. Víctor Manzano Villafuerte, Mg.
PROFESOR CALIFICADOR

Firmado electrónicamente por:
CARLOS DIEGO GORDON
GALLEGOS

Ing. Carlos Gordon Gallegos, PhD.
PROFESOR CALIFICADOR

DEDICATORIA

El esfuerzo y la disciplina es la base para la constancia y el trabajo eficiente en la carrera universitaria. Quiero dedicar mi trabajo de investigación de primeras a Dios, que me ha permitido tener las fuerzas y la sabiduría para poder culminar toda mi tesis y no rendirme en todo este camino.

También lo dedico a mis padres Edison y Betty que son mi pilar y apoyo en todo momento de mi vida quienes no se rinden conmigo, además de todos mis primos que son como mis hermanos y me han dado los ánimos y fuerzas para seguir adelante.

Dedico este trabajo también a todos los ingenieros que me han apoyado y formado en el transcurso de mi carrera universitaria.

Finalmente, dedico este trabajo a las personas que dedican cada uno de sus días en crear ciencia y tecnología para formar y ayudar a crear un mundo mejor.

Jean Carlos León

AGRADECIMIENTO

Agradezco a Dios, la Virgen María, y al patrón San Gerardo santito de mi tierra por darme las fuerzas y salud para no rendirme en todo este camino y ante todo seguir en pie cada día luchando.

Agradezco a mis padres Edison y Betty que siempre son mi apoyo incondicional y me han acompañado en toda mi carrera universitaria.

Agradezco a mis primos que siempre han estado para mí, que me apoyaron desde que empezó mi carrera universitaria estando a mi lado siempre como los hermanos que nunca tuve.

Finalmente agradezco a mi tutor de tesis, docentes y amigos que me han enseñado y formado como profesional y buena persona, para seguir por el camino del bien y ante todo por los conocimientos y enseñanzas que han cultivado en mí para ser un excelente profesional.

Jean Carlos León

DEDICATORIA

Dedico el presente proyecto de investigación a Dios en donde sin su bendición y compañía no habría podido lograr llegar aquí, que, aunque no conversemos mucho jamás deje de creer y confiar en Él y al final sabe cómo y cuándo hace las cosas.

A mis queridos padres, que me apoyaron en contra de todo pronóstico y apostaron porque llegaría a este momento tan valioso, sin ellos no sería el hombre que soy.

A mi querida hermana que siempre me repetía que yo era su "motor", sin saber que ella era el mío.

Con mucho cariño a mi Tío Avelino que, aunque ya no está a mi lado siempre vio lo mejor de mí y que sabía que llegaría a ser alguien de bien, lo quiero al cielo y más allá.

A mi novia Jessica Nataly que fue parte de este logro, con su apoyo y amor he llegado a este momento, siempre creyó en mí que estaría siendo el mejor.

Ricardo Tualombo

AGRADECIMIENTO

Quiero agradecer primeramente a Dios por darme la vida, salud y estar a mi lado cada momento de mi trayecto universitario y cada aspecto de mi vida.

Agradezco a mi Tutor de tesis Ing. Franklin Salazar por confiar en mí, brindarme su apoyo y tiempo para ayudarme a culminar mi trabajo de investigación y gracias a todos los docentes que fueron grandes mentores en mi formación.

Agradezco a mis padres que vieron como perdí mi camino y aun así no se rindieron conmigo gracias por cuidarme cuando más lo necesitaba incluso cuando no, a mi hermana por darme ese apoyo incondicional y a mis sobrinitos Majito y Dante que quizás en este instante no entiendan mucho lo que está pasando y espero que cuando lo hagan estén muy orgullosos de su ÑAÑO RICK los amo 3 Millones.

A Jessica Nataly por ver en mi lo que ni yo mismo puedo ver, has sido una gran motivación para que haga todo bien, o al menos un poco mejor, gracias por tu constante apoyo y tu amor.

A mis amigos que han sido muchos, pero especialmente a Alexis, Christopher, Paul, Estefanía, y a los "Chontapanas" que jamás dejamos de ayudarnos en el camino que estuvimos juntos, ¡Jean Carlos mi compañero de tesis gracias por tu apoyo y colaboración que ha sido muy importante para llegar aquí, lo logramos AMIGO!

Ricardo Tualombo

ÍNDICE GENERAL DE CONTENIDOS

APROBACIÓN DEL TUTOR ..... **ii**

AUTORÍA ..... **iii**

DERECHOS DE AUTOR ..... **iv**

APROBACIÓN DEL TRIBUNAL DE GRADO ..... **v**

DEDICATORIA ..... **vi**

AGRADECIMIENTO ..... **vii**

DEDICATORIA ..... **viii**

AGRADECIMIENTO ..... **ix**

ÍNDICE GENERAL DE CONTENIDOS ..... **x**

ÍNDICE DE TABLAS ..... **xiv**

ÍNDICE DE FIGURAS ..... **xvii**

ÍNDICE DE ANEXOS ..... **xxix**

RESUMEN EJECUTIVO ..... **xxx**

ABSTRACT ..... **xxxi**

CAPÍTULO I. MARCO TEÓRICO **1**

1.1 Tema de investigación 1

1.1.1 Planteamiento del problema 1

1.2 Antecedentes investigativos 3

1.3 Fundamentación teórica 5

1.3.1 UAV…. 5

1.3.2 Tipos de UAV 6

1.3.3 Vigilancia y Operaciones del Espacio Aéreo 12

1.3.4 Redes de Videovigilancia 13

1.3.5 Tecnologías para la videovigilancia y operaciones 14

1.3.6 Sistemas de monitorización 15

1.3.7 Tecnología y Sistema de navegación Autónoma 20

1.3.8 Características de cobertura de videovigilancia 27

1.3.9 Alcance 27

1.3.10 Velocidad 28

1.3.11 Precisión y alcance de los Sistemas de Navegación Autónoma 29

1.3.12 Implementación de Tecnologías de Navegación Autónoma 30

1.3.13 Sistemas de Ubicación 32

1.3.14 Aspectos de navegación 34

1.3.15 Frecuencia de operación en drones en el Ecuador 35

1.4 Objetivos 36

1.4.1 Objetivo general 36

1.4.2 Objetivos específicos 36

CAPÍTULO II. METODOLOGÍA 37

2.1 Materiales 37

2.2 Métodos 38

2.2.1 Modalidad de la investigación 38

2.2.2 Recolección de información 38

2.2.3 Procesamiento y análisis de datos 39

CAPÍTULO III. RESULTADOS Y DISCUSIÓN 40

3.1 Desarrollo de la propuesta 40

3.1.1 Delimitación del terreno 41

3.1.2 Obstáculos dentro del terreno 42

3.1.3 Trayectorias del UAV dentro del terreno delimitado 44

3.1.4 Esquema general del sistema 45

3.1.5 Selección de los elementos de hardware para el sistema de vuelo autónomo de un UAV… 49

3.1.6 Software para el sistema de vuelo autónomo de un UAV 54

3.1.7 Diseño de la estructura del UAV HEXACOPTER TAROT 680 55

3.1.8 Implementación electrónica del sistema de navegación autónoma del UAV ... 59

3.1.9 Configuración completa del sistema de navegación autónoma del UAV 63

3.1.10 Análisis del sistema de navegación autónoma del UAV 119

3.1.11 Simulación de trayectorias del UAV en el software MATLAB 125

3.1.12 Análisis de Resultados de la simulación de las trayectorias del UAV en MATLAB 133

3.1.13 Control PID 136

3.1.14 Simulación para Control Proporcional, Integral y Derivativo (PID) 136

3.1.15 Interfaz del sistema de control y monitoreo del hexacoptero basado en Python….. 139

3.1.16 Simulación de georreferencia para planificación de misiones basado en Python…. 156

3.1.17 Organización del código de simulación de trayectorias y control del hexacóptero basado en Python 166

3.1.18 Simulación de trayectorias del multirotor (hexacoptero) basado en el uso de librerías “Multirotor” en programación de Python 171

3.2 Resultados 187

3.2.1 Pruebas de validación del tiempo de autonomía de vuelo del UAV 187

3.2.2 Pruebas de funcionamiento de planificación de misiones con el programa de georreferencia y simulación de trayectorias del UAV. 195

3.2.3 Eficiencia del sistema de Georreferencia en Python para el sistema automatizado de vuelo del UAV 203

3.3 Presupuesto 205

CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES 208

4.1 Conclusiones 208

4.2 Recomendaciones 209

ANEXOS 222

ÍNDICE DE TABLAS

Tabla 1. UAV´s 11

Tabla 2. Módulos de Video Transmisión 18

Tabla 3. Gimbals 19

Tabla 4. Placa controladora de vuelo. 24

Tabla 5. Módulos GPS. 26

Tabla 6. Características de la cobertura de la videovigilancia de un UAV. 27

Tabla 7. Categorías de los UAV`s 28

Tabla 8. Características de capacidad de respuesta ante incidentes del dron. 30

Tabla 9. Métodos para navegación autónoma en UAV. 31

Tabla 10. Cámaras 31

Tabla 11. Características del sistema de ubicación del UAV 33

Tabla 12. Módulos de Telemetría. 34

Tabla 13. Tipos de sistemas de navegación autónoma. 35

Tabla 14. Recolección de Información 39

Tabla 15. Alturas de los diferentes objetos u obstáculos 44

Tabla 16. Baterias Lipo 54

Tabla 17. Parámetros ajustar para activar el puerto de Telemetría 67

Tabla 18. Frecuencias de operación (MHz) 86

Tabla 19. Selección de banda y canal del monitor FPV 87

Tabla 20. Distancias de conexión respecto al sistema FPV con el monitor FPV 88

Tabla 21. Parámetros ajustar para el control de servo motores..... 101

Tabla 22. Comandos para navegación autónoma en una misión 105

Tabla 23. Parámetros de comando TAKEOFF 107

Tabla 24. Parámetros de comando WAYPOINT..... 109

Tabla 25. Parámetros de comando SPLINE WAYPOINT 110

Tabla 26. Parámetros de comando LOITER TIME 111

Tabla 27. Parámetros de comando LAND 112

Tabla 28. Parámetros de comando DELAY..... 113

Tabla 29. Parámetros de comando CONDITION DISTANCE 114

Tabla 30. Parámetros de comando DO CHANGE SPEED 115

Tabla 31. Consumo energético del gimbal del UAV 122

Tabla 32. Consumo energético del sistema completo del UAV. 122

Tabla 33. Distribución de Pesos de los Componentes del Dron Hexacóptero..... 122

Tabla 34. Librerías utilizadas en el programa de sistema de control y monitoreo del UAV 140

Tabla 35. Librerías utilizadas en el programa de georreferencia 156

Tabla 36. Librerías adicionales utilizadas en la simulación de trayectorias de multirotor..... 171

Tabla 37. Parámetros de las pruebas de vuelo del UAV de la misión planificada .. 187

Tabla 38. Mediciones de tiempo respecto a parámetros establecidos de las pruebas de vuelo..... 188

Tabla 39. Parámetros de sistemas de georreferencia 196

Tabla 40. Comparación de puntos georreferenciados con cálculo de errores de la prueba 1 198

Tabla 41. Comparación de puntos georreferenciados con cálculo de errores de la prueba 6 200

Tabla 42. Comparación de puntos georreferenciados con cálculo de errores de la prueba 9 202

Tabla 43. Parámetro de la Eficiencia de cada sistema 204

Tabla 44. Comparativa eficiencia promedio de cada uno de los sistemas 205

Tabla 45. Costos. 206

ÍNDICE DE FIGURAS

Figura 1. Cronología de los nombres aplicados a las aeronaves robóticas 6

Figura 2. Dron en físico DJI Mavic 2. 7

Figura 3. Dron en físico Yuneec H520. 7

Figura 4. Dron en físico Autel Robotics Evo II Dual. 8

Figura 5. Dron en físico DJI Matrice 300 RTK. 8

Figura 6. Dron en físico Parrot Anafi USA. 9

Figura 7. Dron Tarot M860 cuadricóptero 9

Figura 8. Dron Readytosky ZD550 10

Figura 9. Dron Tarot FY690S 11

Figura 10. Cámara CCTV puesta en un parapente. 14

Figura 11. Sistema de grabador de video CCTV. 16

Figura 12. Vigilancia por personal contratado 16

Figura 13. Arquitectura de solución de Edificios Inteligentes Hikvision 17

Figura 14. Dron TAROT 680 PRO, hexacóptero con gimbal integrado 18

Figura 15. Descriptores en el entorno real empleando ORB-SLAM. 21

Figura 16. Mapa y localización empleando ORB-SLAM. 21

Figura 17. Diagrama de bloques de un sistema de visión artificial. 23

Figura 18. Alcance de un UAV. 29

Figura 19. Sistemas de navegación con drones. 30

Figura 20. Sistemas integrados en un UAV 33

Figura 21. Vista en 3D de Google Earth - Talleres FISEI 41

Figura 22. Vista en 3D de Google Earth – Parqueaderos Universidad Técnica de Ambato 42

Figura 23. Objetos como árboles y dentro del campus farolas 43

Figura 24. Objetos como farolas y señales de tránsito. 43

Figura 25. Tipos de ruta 45

Figura 26. Arquitectura del Sistema 47

Figura 27. Marco Principal del dispositivo 56

Figura 28. Diseño brazos exteriores. 56

Figura 29. Diseño de motores. 57

Figura 30. Diseño de Hélices. 57

Figura 31. Diseño de antena Satelital. 58

Figura 32. Diseño de Antena Rx. 58

Figura 33. Diseño renderizado del UAV 59

Figura 34. Conexión Electrónica del Sistema completo del UAV 59

Figura 35. Partes del Heads-up display (HUD) 64

Figura 36. Partes detalladas de Flight Data del software Mission Planner 65

Figura 37. Diagrama de flujo de funcionamiento de Telemetría 66

Figura 38. Esquema electrónico del sistema de telemetría del dron con la estación terrena 66

Figura 39. Módulo de telemetría montado en el UAV y en la estación terrena 67

Figura 40. Configuración de parámetros para el módulo de telemetría dentro del software Mission Planner 68

Figura 41. Desconexión y conexión mediante el software Mission Planner 68

Figura 42. Calibración Acelerómetro Inicio 69

Figura 43. Posiciones de calibración del dron 70

Figura 44. Mensaje en el software finalización correcta del Acelerómetro 71

Figura 45. Calibración Compass iniciación en el software Mission Planner 72

Figura 46. Movimiento del dron para calibración del Compas en físico 73

Figura 47. Calibración completada del Compas satisfactoriamente 73

Figura 48. Calibración del ESC iniciación en el software Mission Planner 74

Figura 49. Calibración del Radio Control iniciación en el software Mission Planner 76

Figura 50. Calibración del Radio control. 77

Figura 51. Módulo Radioreceptor FrSky R-XSR encendido 77

Figura 52. Confirmación para la calibración del radio control 78

Figura 53. Configuración de Modos de vuelo dentro del software Mission Planner. 79

Figura 54. Cargar parámetros para el canal de RC para cambios de Modo de vuelo dentro del software Mission Planner 79

Figura 55. Configuración de Canal en la interfaz del radio control TX16 RadioMaster 80

Figura 56. Configuración del interruptor para los tres modos de vuelo en la interfaz del radio control TX16 RadioMaster 80

Figura 57. Apartado de Plan (Planeo de misiones) dentro del software Mission Planner 81

Figura 58. Diagrama con waypoints de una misión para el UAV 82

Figura 59. Armado del UAV (Led verde) 83

Figura 60. Indicador GPS/BRUJULA del UAV (Led azul) 83

Figura 61. Esquema electrónico del Sistema FPV 84

Figura 62. Diagrama de flujo de la configuración y montaje del sistema FPV 85

Figura 63. Botón A de la placa MATEK HV 5.8 Ghz 85

Figura 64. Botón B de la placa MATEK HV 5.8 Ghz 86

Figura 65. Visualización del módulo de video transmisión en funcionamiento 86

Figura 66. Configuración canal y frecuencia en el monitor FPV 87

Figura 67. Diagrama de flujo de configuraciones y pruebas del gimbal 89

Figura 68. Esquema electrónico del sistema del gimbal 90

Figura 69. Indicadores Led y muestra de la placa de control del Gimbal 91

Figura 70. Ingreso por cmd al software AVRdude 91

Figura 71. Conexiones de la placa de control con el módulo USBtinyisp 92

Figura 72. Ingreso de comandos por cmd en AVRdude para la configuración del firmware de la placa de control 92

Figura 73. Ingreso de comando por cmd en AVRdude para la configuración de paquetes hexadecimal de la placa de control 93

Figura 74. Ingreso de comando por CMD en AVRdude para cargar el firmware descargado del archivo hexadecimal de la placa de control 94

Figura 75. Interfaz Gráfica GUI para configuraciones de la placa de control del gimbal 95

Figura 76. Parámetros configurados para la calibración de la placa de control del gimbal 96

Figura 77. Guardar perfil de calibración dentro del software SimpleBGC_GUI_2.2 97

Figura 78. Gimbal en forma física desarmado 97

Figura 79. Montaje del gimbal en el dron 98

Figura 80. Cambio de parámetros de potencia dentro del software SimpleBGC_GUI_2.2 98

Figura 81. Grafica de perturbaciones en tiempo real de datos recogidos por el software SimpleBGU_GUI_2.2 99

Figura 82. Configuración en la interfaz del radio control apartado de entradas en configuraciones de telemetría 100

Figura 83. Configuración de canales en mixes dentro de la interfaz del radio control 100

Figura 84. Partes del gimbal 101

Figura 85. Valores para configuración dentro de los parámetros SERVO10_FUNCTION y SERVO9_FUNCTION en Mission Planner 102

Figura 86. Configuración de parámetros para el gimbal dentro del software Mission Planner 103

Figura 87. Perillas S1 y S2 en el radio control 103

Figura 88. Diagrama de flujo de la misión de vuelo del hexacóptero 104

Figura 89. Diagrama de flujo de conexión y funcionamiento de telemetría 105

Figura 90. Partes del apartado de planificación de misiones en el software Mission Planner..... 107

Figura 91. Comando TAKEOFF dentro del software Mission Planner..... 108

Figura 92. Comando WAYPOINT dentro del software Mission Planner 109

Figura 93. Comando SPLINE WAYPOINT dentro del software Mission Planner. 110

Figura 94. Comando LOITER TIME dentro del software Mission Planner..... 111

Figura 95. Comando RETURN TO LAUCHN dentro del software Mission Planner 112

Figura 96. Comando LAND dentro del software Mission Planner..... 113

Figura 97. Comando DELAY dentro del software Mission Planner 114

Figura 98. Comando CONDITION DISTANCE dentro del software Mission Planner 115

Figura 99. Comando DO CHANGE SPEED dentro del software Mission Planner 116

Figura 100. Realizar una acción en el apartado de DATA, para Mission_Start en el software Mission Planner..... 117

Figura 101. Realizar una acción en el apartado de DATA, para Arm/Disarm en el software Mission Planner..... 118

Figura 102. Cargar y Leer misión en el apartado de PLAN en Mission Planner..... 118

Figura 103. Configuración cinemática del UAV en el espacio..... 120

Figura 104. Peso del dron en la balanza analógica 123

Figura 105. Parámetros del UAV..... 126

Figura 106. Ruta configurada por una matriz. 126

Figura 107. Posición inicial del dron. 127

Figura 108. Cargar el archivo de diseño 128

Figura 109. Dibujar los puntos de trayectorias. 128

Figura 110. Ajuste de Limites en el plano 3D 129

Figura 111. Configuración para simulación. 132

Figura 112. Visualización del dron en el espacio. 132

Figura 113. Ajuste de la posición del dron en tiempo real 133

Figura 114. Trayectoria del UAV. 134

Figura 115. Trayectoria del UAV a 45° 134

Figura 116. Trayectoria del UAV con altura constante 135

Figura 117. Trayectoria del UAV en forma circular 135

Figura 118. Trayectoria libre del UAV 136

Figura 119. Trayectoria base 137

Figura 120. Subsistema para PID 138

Figura 121. Sistema de control PID 139

Figura 122. Altura de estabilización control PID 139

Figura 123. Interfaz gráfica realizada en Tkinter para control y monitoreo del UAV 141

Figura 124. Función para conexión con el UAV y la estación terrena 142

Figura 125. Mensaje de alerta previo a la conexión al UAV. 142

Figura 126. Función para la desconexión con el UAV. 143

Figura 127. Mensaje de alerta previo a la desconexión con el UAV. 143

Figura 128. Función de armado de motores del UAV 144

Figura 129. Mensaje de alerta y visualización en físico del UAV armado..... 144

Figura 130. Función de desarmado del UAV..... 145

Figura 131. Mensaje de alerta y visualización en físico del UAV desarmado 145

Figura 132. Función de cambio de modo de vuelo del UAV 146

Figura 133. Mensaje de alerta de cambio de modo de vuelo Estabilizado y visualización en la pantalla OLED del UAV 147

Figura 134. Mensaje de alerta pasos requeridos para inicio de misión y visualización en la pantalla OLED del UAV 147

Figura 135. Función read_kml para cargar misión de la interfaz del UAV..... 148

Figura 136. Mensaje de alerta de archivo KML cargado del UAV 149

Figura 137. Función para cargar misión en el UAV 150

Figura 138. Mensaje de alerta de misión cargada correctamente en el UAV..... 150

Figura 139. Función para iniciar la misión cargada en el UAV..... 151

Figura 140. Mensaje de alerta en la interfaz e inicio de misión en físico del UAV 152

Figura 141. Funciones para ejecutar subprocesos de los scripts del sistema de georreferencia y simulación de trayectorias..... 153

Figura 142. Mensaje de alerta ejecución de scripts para planificación de misiones y simulación de trayectorias del UAV 153

Figura 143. Función para actualización y recolección de parámetros del UAV..... 154

Figura 144. Parámetros del UAV mostrados en la Interfaz. 155

Figura 145. Función de actualización de graficas en tiempo real de parámetros del UAV. 155

Figura 146. Graficas de parámetros del UAV en tiempo real 156

Figura 147. Interfaz gráfica realizada con la librería Tkinter en Python 157

Figura 148. Cargar archivo en formato MISSION. 158

Figura 149. Tabla de puntos con los datos procesados de la misión cargada 159

Figura 150. Creación de archivos KML, JSON y mapa generado en OpenStreetMap, además del punto inicial (HOME). 159

Figura 151. Función para generar el archivo KML en el programa de georreferencia 160

Figura 152. Mensaje de gratificación de archivo KML generado y visualización de puntos de la ruta planificada en Google earth. 160

Figura 153. Función para creación del mapa OpenStreet y trazar puntos y polígonos deseados del programa de georreferencia. 162

Figura 154. Mapa OpenStreet visualización de trayectoria cargada del programa de georreferencia. 162

Figura 155. Función para el cálculo de distancia entre puntos del mapa OpenStreet del programa de georreferencia. 163

Figura 156. Función de conversión de coordenadas geográficas a ECEF del programa de georreferencia. 164

Figura 157. Función de conversión de puntos ECEF a coordenadas cartesianas. ... 165

Figura 158. Formato del archivo json de las coordenadas cartesianas para la simulación de trayectorias. 165

Figura 159. Diagrama de clases de la simulación de multirrotores 166

Figura 160. Diagrama de clases de la simulación de control 169

Figura 161. Parámetros y clases utilizadas en la creación del UAV 172

Figura 162. Parámetros del UAV (Hexa rotor) de las características técnicas de las hélices. 173

Figura 163. Parámetros de estado y métodos usados para resultados de estado del UAV y propiedades adicionales. 174

Figura 164. Resultados de Estado del UAV, parámetros técnicos y propiedades adicionales 174

Figura 165. Código de visualización en un entorno 3D para el UAV 175

Figura 166. Resultados de la trayectoria del UAV con un indicador de la posición del dron en un entorno 3D 175

Figura 167. Función para la gráfica de mediciones temporales en base a la trayectoria del UAV 176

Figura 168. Gráficas de mediciones de series temporales de algunos parámetros de vuelo del UAV respecto a una trayectoria. 177

Figura 169. Método para la cinemática del UAV con el objeto dynamics 178

Figura 170. Resultados de la variable dynamics 178

Figura 171. Objeto de Velocidades del UAV (Hexarotor) de acuerdo con la dinámica 178

Figura 172. Resultados de la variable speeds 178

Figura 173. Objeto “state” para indicar el estado de las velocidades aplicadas al UAV (hexarotor) 179

Figura 174. Resultados de la velocidad aplicadas al UAV (Hexarotor) 179

Figura 175. Condicional y variables utilizadas para graficar los plots y escenario 3D 179

Figura 176. Resultados de la simulación de trayectorias en un entorno 3D y graficas de series temporales del UAV (Hexa rotor) 180

Figura 177. Función optimize en un objeto study para aplicación de parámetros del control PID óptimo. 181

Figura 178. Resultados de valores optimizados del controlador PID 182

Figura 179. Grafica de todos los parámetros de optimización del controlador de PID 182

Figura 180. Uso de la clase Trajectory para trayectorias del UAV (Hexarotor)..... 183

Figura 181. Resultados de simulación de trayectorias del UAV (Hexarotor) aplicando la clase Trajectory. 184

Figura 182. Funciones Cargar coordenadas json e Interpolación de puntos para trayectorias del UAV (Hexarotor)..... 185

Figura 183. Código para segunda gráfica de trayectorias del UAV (Hexarotor) en 3D. 186

Figura 184. Resultado de la simulación trayectorias del UAV (Hexarotor) en una segunda gráfica con una escala menor. 186

Figura 185. Datos de la velocidad del viento en los talleres tecnológicos FISEI y el campus Huachi de la UTA. 189

Figura 186. Mediciones de tiempo de las pruebas de vuelo. 190

Figura 187. Gráfico de barras del tiempo del vuelo del dron respecto a cada prueba realizada. 191

Figura 188. Pruebas realizadas en los talleres tecnológicos FISEI..... 192

Figura 189. Pruebas realizadas en el garaje del campus Huachi de la UTA..... 193

Figura 190. Pruebas realizadas en Google earth de los talleres tecnológicos de la FISEI. 194

Figura 191. Pruebas realizadas en Google earth del garaje del campus Huachi de la UTA..... 195

Figura 192. Aplicación móvil para georreferenciar puntos geoespaciales. 196

Figura 193. Visor geográfico del Instituto Geográfico militar de Ecuador 197

Figura 194. Ruta de la prueba 1en el mapa OpenStreet Maps y Google Earth..... 200

Figura 195. Ruta de la prueba 6 en el mapa OpenStreet Maps y Google Earth..... 201

Figura 196. Ruta de la prueba 9 en el mapa OpenStreet Maps y Google Earth..... 203

Figura 197. Gráfico de comparación de eficiencias promedio de cada sistema. 205

ÍNDICE DE ANEXOS

Anexo A. Listado de abreviaturas 222

Anexo B. Manual del software de configuración SIMPLEBGU de la placa de control del gimbal. 224

Anexo C. Monitorización desde la estación Base. 226

Anexo D. Pruebas de vuelo alrededor de los Talleres Tecnológicos. 226

Anexo E. Pruebas de vuelo alrededor de Garaje Campus Huachi. 227

Anexo F. Prueba de vuelo en el ESTADIO UNIVERSITARIO. 229

Anexo G. Manual de manejo del dispositivo 230

Anexo H. Comparativa de tecnologías y técnicas empleadas para sistemas de navegación autónoma 241

RESUMEN EJECUTIVO

La inseguridad es un problema creciente que afecta a diversos entornos, incluyendo instituciones educativas, donde se presentan riesgos relacionados con la protección de bienes y personas. En la Universidad Técnica de Ambato, específicamente en el campus Huachi Chico, se evidenciaron en áreas limitadas como los estacionamientos y el sector de talleres tecnológicos de la FISEI.

El trabajo de investigación, enmarcado en la línea de investigación de Tecnología de la Información y Sistemas de Control, propuso la implementación de un sistema de navegación autónoma para un hexacóptero, diseñado para brindar soporte en videovigilancia y operaciones aéreas en el campus. El sistema integra tecnología FPV para transmisión en tiempo real y un gimbal equipado con una cámara, que permitió capturar fotografías y videos. Además, la versatilidad del sistema pudo realizar distintas operaciones, como monitoreo, tareas de seguridad o incluso la recolección de datos audiovisuales según las necesidades de cada situación dentro del campus.

Se desarrolló en etapas como recolección de datos y video, procesamiento, monitoreo y control usando como placa principal del sistema una Pixhawk 2.4.8, configurando rutas diseñadas en una interfaz de monitoreo y control que incluye un sistema de georreferencia elaborado en Python utilizando tecnología GPS para sistemas GNSS, también simulaciones de trayectorias para visualizar el comportamiento del dron en las pruebas de vuelo respecto a la planificación de una misión.

Los resultados obtenidos fueron la eficiencia del sistema en cuanto a la comparación con otros sistemas de georreferencia de acuerdo con las rutas establecidas y el tiempo de autonomía de vuelo impuesto previamente bajo parámetros como velocidad de vuelo, altitud y tipo de misión preestablecida en un ambiente real.

Palabras clave: Autónomo, dron, geolocalización, Mission Planner, inseguridad, videovigilancia.

ABSTRACT

Insecurity is a growing problem that affects various environments, including educational institutions, where risks related to the protection of property and people arise. At the Technical University of Ambato, specifically in the Huachi Chico campus, issues were evident in limited areas such as parking lots and the technological workshop sector of FISEI.

The research, framed within the line of Information Technology and Control Systems, proposed the implementation of an autonomous navigation system for a hexacopter, designed to provide support in video surveillance and aerial operations on campus. The system integrates FPV technology for real-time transmission and a gimbal equipped with a camera, allowing for the capture of photographs and videos. Additionally, the versatility of the system enables it to perform various operations, such as monitoring, security tasks, or even collecting audiovisual data according to the needs of each situation within the campus.

The development occurred in stages, including data and video collection, processing, monitoring, and control, using a Pixhawk 2.4.8 as the main board of the system. It involved configuring routes designed in a monitoring and control interface that includes a georeferencing system developed in Python using GPS technology for GNSS systems, as well as trajectory simulations for visualizing the drone's behavior during flight tests in relation to mission planning.

The results obtained demonstrated the system's efficiency in comparison with other georeferencing systems according to the established routes and the previously imposed flight autonomy time under parameters such as flight speed, altitude, and pre-established mission type in a real environment.

Keywords: Autonomous, drone, geolocation, Mission Planner, insecurity, video surveillance.

CAPÍTULO I. MARCO TEÓRICO

1.1 Tema de investigación

SISTEMA DE NAVEGACIÓN AUTÓNOMA DE UN UAV PARA VIDEOVIGILANCIA Y OPERACIONES DEL ESPACIO AÉREO DE LA UNIVERSIDAD TÉCNICA DE AMBATO

1.1.1 Planteamiento del problema

En la actualidad, la inseguridad se ha convertido en una preocupación persistente en diversas regiones de Ecuador. Según reportes de la Policía Nacional, entre 2021 y 2022 se ha registrado un alarmante aumento en las acciones delictivas contra las personas y sus bienes materiales, con incrementos del 21% y 50%, respectivamente. Esta problemática no solo afecta a la sociedad en general, sino que también impacta significativamente en el entorno universitario. Las universidades del país han sido escenario de numerosos incidentes delictivos, particularmente robos, que perjudican a los estudiantes tanto dentro como en las inmediaciones de los campus. Esta situación genera un ambiente de inseguridad que compromete el bienestar y la calidad de vida de la comunidad universitaria, y plantea la urgente necesidad de desarrollar e implementar estrategias efectivas para mejorar la seguridad en estos espacios educativos [1].

Refiriéndose ahora a nivel de la provincia de Tungurahua, los índices delictivos han generado inquietud entre estudiantes y autoridades de las universidades, afectando no solo la tranquilidad cotidiana de los habitantes, sino también el desarrollo social y económico de la región esto según datos estipulados a personas encuestadas por la entidad OBEST (Observatorio económico y social de Tungurahua) el mayor temor fue la inseguridad y la delincuencia teniendo un aumento de 33,8%, este porcentaje pasó a 62,85% en 2022 [2].

En la ciudad de Ambato, específicamente en la Universidad Técnica de Ambato (UTA), la percepción de seguridad es un indicador crucial que refleja la sensación de seguridad o inseguridad en el entorno. En 2020, este índice se situó en el 73%, pero

para 2022 descendió al 68%, lo que indica que los estudiantes de la UTA se sienten más inseguros que antes, con una disminución del 6% en su percepción de seguridad. Esta problemática no solo exige respuestas efectivas por parte de las autoridades, sino también la implementación estratégica de tecnologías innovadoras, debido a la falta de suficientes autoridades o entidades de seguridad que puedan fortalecer las capacidades de vigilancia y protección en el campus [1].

Las afirmaciones de la ausencia de cámaras y la mala iluminación en ciertas áreas de los campus universitarios pueden aumentar la delincuencia deben evaluarse cuidadosamente. Si bien estas deficiencias pueden contribuir a un entorno menos seguro al promover la impunidad y dar cobertura a las actividades ilegales, la delincuencia es un fenómeno complejo influenciado por una gama más amplia de factores sociales y económicos. Además, instalar cámaras y mejorar la iluminación puede tener limitaciones prácticas y éticas que deben considerarse cuidadosamente para desarrollar una estrategia de seguridad efectiva y equilibrada en el campus.

La falta de videovigilancia eficaz y sistemas de seguridad ineficaces pueden causar preocupación y malestar en la comunidad universitaria, pero es necesario analizar críticamente cómo se define y evalúa la eficacia de estos sistemas. La percepción de seguridad está influenciada por la experiencia personal y factores objetivos como la calidad de la infraestructura y la formación del personal de seguridad. Es fundamental identificar áreas de mejora y desarrollar estrategias efectivas para abordar los problemas de seguridad dentro de la comunidad universitaria.

La insuficiencia de recursos humanos involucrados en la seguridad dentro de la Universidad Técnica de Ambato reduce las áreas protegidas en el campus siendo un desafío serio. Esta situación aumenta el riesgo de incidentes delictivos y afecta la sensación de seguridad en la comunidad universitaria. Sin embargo, además de examinar estrategias para el uso óptimo de estos recursos humanos, también se debe examinar la eficacia y preparación del personal de seguridad existente. Mencionando específicamente para el uso de tecnologías con drones que disminuirá mucho las áreas desprotegidas dentro de la Universidad Técnica de Ambato.

1.2 Antecedentes investigativos

En 2019, María Carolina Bravo Navarro y Diego Felipe Ruiz Barreto, realizaron la investigación "Navegación autónoma y evasión de obstáculos en UAV usando aprendizaje por refuerzo", en la Universidad Santo Tomás de Bogotá, Colombia. La investigación aborda la navegación autónoma de vehículos aéreos no tripulados (UAV) en entornos no estructurados, utilizando técnicas de aprendizaje por refuerzo, específicamente la técnica de Q-Learning, se propone un sistema de navegación autónoma en tiempo real que permite a los UAV adaptarse a los cambios del entorno. El sistema se implementa utilizando MATLAB y el entorno de simulación virtual V-REP, lo que permite simular escenarios realistas como el rescate de personas en desastres naturales o la entrega de paquetes. De acuerdo con el entramiento realizado por el logaritmo con la configuración de 2972 estados, tiene una fiabilidad del 89.6%, con 4 acciones entre adelante, izquierda, derecha y atrás determinaron un valor Q estable de 1389 [3].

En 2020, Amparo Meythaler, realizo la investigación “Desarrollo de un sistema de Navegación Autónoma para UAV basado en FPGA”, en la Universidad de las Fuerzas Armadas ESP-EL, Ecuador. En este trabajo se presenta el desarrollo de un sistema de navegación autónoma de vuelo para vehículos aéreos no tripulados (UAVs) empleando tecnología FPGA. Para ello, llevaron a cabo un análisis exhaustivo de la lógica de control del FPGA, estableciendo el formato de trama adecuado para la adquisición y procesamiento de las señales necesarias para el control de una tarjeta estabilizadora de vuelo y mostrado una interfaz de comunicación de entrada, así los resultados de acuerdo a los algoritmos de ejecución de rutas, teniendo en cuenta las señales de control de salida, el tiempo estimado para cubrir la ruta a una distancia de 200 metros obtuvieron un valor entre 5 minutos a 7 minutos, con una eficiencia del sistema para la ruta propuesta del 85.89% [4].

En el año 2022, Daniel Silva Iglesias, realizo la investigación "Aplicación de técnicas de Visión Artificial en la retransmisión y grabaciones realizadas por Drones" en la Facultad de Informática de la Universidad de da Coruña, España. El proyecto se adentró en la exploración de métodos y algoritmos de procesamiento de imágenes y vídeos, específicamente orientados a la captura realizada por aeronaves no tripuladas.

El propósito fundamental radicó en optimizar la capacidad perceptiva del entorno durante las operaciones aéreas, mediante el diseño e implementación de sistemas capaces de analizar y procesar el contenido multimedia en tiempo real. Se analizaron los resultados del entrenamiento utilizando las librerías de Python YOLO v4 de ambas redes neuronales mediante matrices de confusión, con una resolución de 416x416, 608x608 y 832x832, obtuvieron respectivamente una precisión de 0.69, 0.68 y 0,67, mientras que utilizando YOLO-tiny comparado con las dos últimas resoluciones obtuvieron respectivamente una precisión de 0.90, 0.86 y 0.83 [5].

En el año 2022, Pablo García García presento su investigación con el título "Sistema de seguimiento de personas desde un dron", en la Universidad de Alicante, España. En la investigación, se desarrolló un sistema que utiliza tecnologías avanzadas para mejorar la capacidad de detección y seguimiento de objetos por parte de los sistemas de vehículos aéreos no tripulados (UAVs). El sistema propuesto combina técnicas de visión artificial y aprendizaje autónomo para analizar y procesar las imágenes capturadas por los drones en tiempo real. Mediante el uso de algoritmos de IA, utilizando redes neuronales artificiales, los resultados obtenidos se evalúan con algunos tipos de métricas como mapeo, intersección y mediciones, se compara los resultados con redes neuronales en YOLO, con media de valores eficientes de 0.95 que es el 95% de todas las muestras, en cambio con YOLOv3 se obtiene un 0.957 que es el 95.7% de eficiencia y por último con YOLOv5 con un valor de estabilidad de 0.995 que es el 99.5% de eficiencia de adaptación [6].

En 2024, Fabian Misael Cango Sarango, realizo la investigación “Sistema de localización de personas en catástrofes utilizando vehículos aéreos no tripulados (UAV’S)”, en la Universidad Técnica de Ambato, Ecuador. La investigación se basó en el diseño de un sistema para la detección de individuos en situación de riesgo por una catástrofe natural utilizando un dron MATERK - FPV, una cámara térmica Tau 2 y otra cámara RGB, realizando un dispositivo incorporado al drone para la toma de datos y localización mediante un módulo de GPS Taoglas – CGGP.25, para mostrar en una interfaz realizada en python. En el sistema se utiliza una red neuronal con librerías de Python como son YOLOv8, los resultados obtenidos de acuerdo con una toma de 1000 imágenes para entrenar a la red neuronal genero una precisión del 90% [7].

1.3 Fundamentación teórica

En este proyecto es esencial comprender de manera detallada los conceptos, principios y enfoques que han sido previamente utilizados en investigaciones relacionadas con la video vigilancia utilizando drones con un sistema automatizado de trayectorias ya sea con visión artificial o por sonorización.

1.3.1 UAV

La aviación no tripulada comprende una amplia gama de aeronaves. El origen de estas aeronaves no tripuladas reside en la creación de los torpedos aéreos, que después evolucionaron a través de las bombas guiadas, los blancos aéreos (o drones), los señuelos, los modelos deportivos de radiocontrol, las aeronaves de investigación, las aeronaves de reconocimiento y las aeronaves de combate.
Un UAV es «un vehículo aéreo motorizado el cuál no lleva a bordo a un operador humano, que utiliza las fuerzas aerodinámicas para proporcionar la elevación del vehículo, que puede volar autónomamente o ser tripulado por control remoto, que puede ser sustituido o recuperado, y que puede transportar una carga letal o no» [8]. Dada esta definición quedan excluidos [9]:

- Los planeadores, debido a que no usan una planta propulsora.
- Los globos y dirigibles, ya que no se elevan mediante fuerzas aerodinámicas sino mediante fuerzas de flotabilidad.
- Los misiles balísticos, misiles de crucero y proyectiles de artillería.

La palabra UAV se comienza a hacer frecuente en los años 90 para representar a las aeronaves robóticas y así, reemplazar el término vehículo aéreo pilotado remotamente o RPV. UAV y RPV no son más que dos nombres entre aproximadamente la docena que han recibido las aeronaves robóticas no tripuladas a lo largo de la historia como se indica en la Figura 1 [9].

Así es como se acuñan los términos que a continuación se detallan, y que en la actualidad tienen una validez internacional y prácticamente única en todos los ámbitos. Estos términos son [9]:

- **Aeronave pilotada remotamente o RPA:** aeronave en la que el piloto al mando no está a bordo [9]:

- **Sistema de aeronave pilotada remotamente o RPAS:** conjunto de elementos configurables formado por un RPA, una estación de pilotaje remoto asociada o RPS, un sistema de enlace de mando y cualquier otro elemento requerido en cualquier punto durante la operación del vuelo [9]:

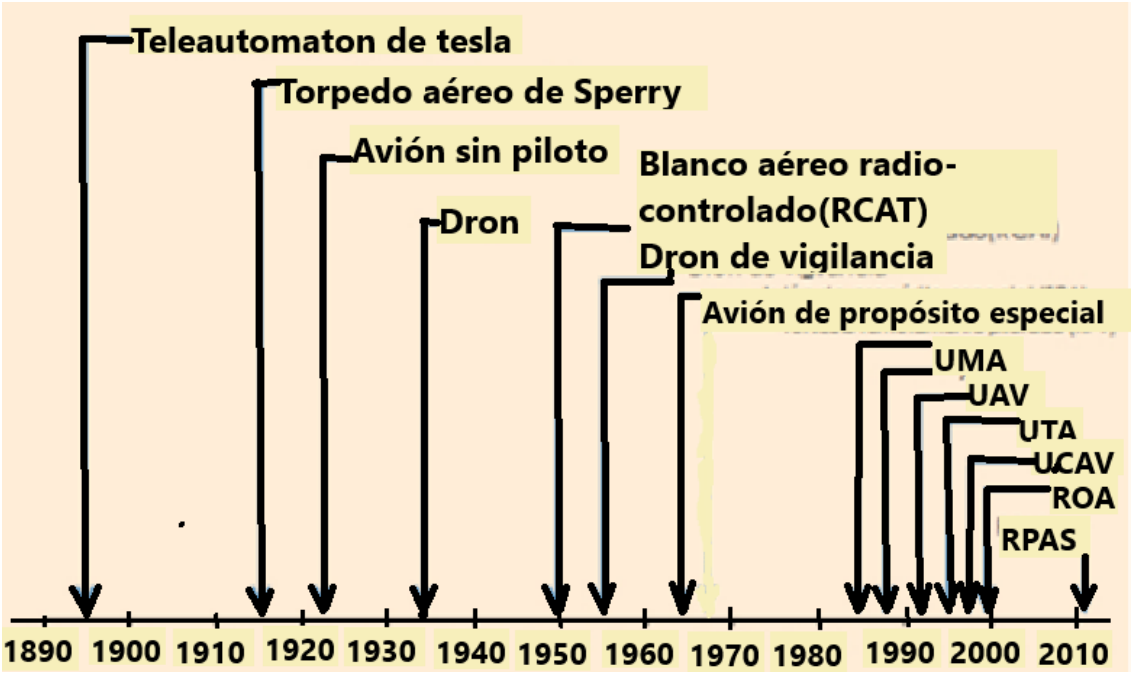


Figura 1. Cronología de los nombres aplicados a las aeronaves robóticas [9].

1.3.2 Tipos de UAV

Existe una variedad de UAV´s controladas remotamente, para la selección del dron de acuerdo con las necesidades y la aplicación primero es importante entender las características que tiene un dron específicamente para la tarea que va a ser aplicado. En el siguiente apartado, se hablará sobre algunos de los que se consideran, son los mejores drones para seguridad y vigilancia del mercado en la actualidad.

a. DJI Mavic 2 Enterprise Dual

Entre las características más notables en este equipo encontramos su cámara térmica y visual integrada como se muestra en la Figura 2, la cual ofrece 3 opciones de cámara: térmica, visual y ambas al mismo tiempo. Además, de un zoom de cámara de hasta 4x, batería de vuelo de 31 minutos con un alcance de hasta 8000 m, así como una capacidad de carga útil de hasta 1000 gr [10].



Figura 2. Dron en físico DJI Mavic 2.

b. Yuneec H520

El Yuneec H520 es un dron potente que ofrece una excelente calidad de imagen gracias a su cámara térmica y visual integrada y su zoom óptico de 3x como se muestra en la Figura 3. Además, este equipo tiene una autonomía de vuelo de 28 minutos y un alcance de hasta 100 m. Otras características notables incluyen el sistema de evitación de obstáculos de 360 grados y la capacidad de carga útil de hasta 500 gr [10].



Figura 3. Dron en físico Yuneec H520.

c. Autel Robotics Evo II Dual

El Autel Robotics Evo II Dual es un equipo que cuenta con una cámara térmica y visual intercambiable como se muestra en la Figura 4, la cual tiene 3 opciones de uso: térmica, visual y ambas, es decir, que pueden usarse al mismo tiempo. Este equipo posee una autonomía de vuelo de hasta 40 minutos y un alcance de 9000 m. Otras características notables incluyen un Zoom de cámara de hasta 8x y su capacidad de carga es de hasta 640 gr [10].



Figura 4. Dron en físico Autel Robotics Evo II Dual.

d. DJI Matrice 300 RTK

El DJI Matrice 300 RTK es un equipo de muy alta calidad que posee una cámara térmica y visual integrada con un Zoom óptico de 20x se podrá observar como se muestra en la Figura 5. Tiene un alcance de hasta 1500 m y su batería puede ofrecer una autonomía se vuelo de 55 minutos. Otras características que sobresalen en este gran equipo incluyen un sistema de evitación de obstáculos de 6 direcciones y una gran capacidad de carga útil de hasta 2000 gr [10].



Figura 5. Dron en físico DJI Matrice 300 RTK.

e. Parrot Anafi USA

El Parrot Anafi USA es un dron con cámara térmica y visual intercambiable. Posee un alcance de hasta 4000 m y una autonomía de vuelo de 32 minutos. Otras características en este equipo incluyen un Zoom de cámara de un rango de por 4 y una capacidad de carga útil de hasta 500 gr. Tiene un diseño plegable y cómodo que facilita su transporte y almacenamiento [10].



Figura 6. Dron en físico Parrot Anafi USA.

f. Tarot M860 cuadricóptero

El Tarot M860 es un cuadricóptero de nivel profesional como se muestra en la Figura 7, diseñado para aplicaciones industriales, que incluyen fotografía aérea, topografía y cartografía. Diseñado para usuarios que requieren una larga resistencia y una alta capacidad de carga útil, el M860 ofrece un rendimiento excepcional con una capacidad de carga útil de 3000 gr, distancia entre ejes de 860 mm y potentes motores brushlees 12S/6009/130KV. Con 65 minutos de tiempo de vuelo sin carga, es ideal para misiones prolongadas y tareas aéreas complejas [11].



Figura 7. Dron Tarot M860 cuadricóptero

g. Readytosky ZD550 Cuadricóptero

El cuadricóptero ZD550 es una plataforma versátil y personalizable utilizada en proyectos de investigación y desarrollo, destacándose por su estructura de fibra de carbono, que le confiere ligereza y durabilidad este se puede visualizar en la Figura 8. Con un diámetro de 550 mm, es compatible con una amplia variedad de componentes como controladores de vuelo (Pixhawk), sensores avanzados (LIDAR, cámaras RGB, GPS), y motores sin escobillas, lo que permite su adaptación para tareas complejas como la evasión de obstáculos y la navegación autónoma. Su modularidad lo convierte en una herramienta ideal para pruebas de tecnologías de control de vuelo autónomo, visión por computadora y recolección de datos aéreos, siendo una opción confiable tanto para proyectos académicos como industriales.



Figura 8. Dron Readytosky ZD550

h. Tarot FY690S Hexacóptero

El TAROT FY690S Pro Hexacóptero es un dron modular y plegable como se muestra en la Figura 9, ampliamente utilizado en aplicaciones profesionales como fotografía aérea, mapeo, inspecciones industriales y proyectos de investigación. Su estructura de fibra de carbono lo hace ligero y resistente, mientras que su diseño hexacóptero ofrece gran estabilidad y redundancia en vuelo, permitiendo continuar operando incluso si un motor falla. Tiene un diámetro de 680 mm, es compatible con controladores de vuelo como Pixhawk y DJI Naza, y soporta baterías LiPo de 4S o 6S, ofreciendo tiempos de vuelo entre 12 y 25 minutos dependiendo de la carga útil, que puede alcanzar hasta 3000 gr. Este dron admite la integración de gimbals, cámaras, sensores como LiDAR

o cámaras térmicas, lo que lo convierte en una plataforma versátil para aplicaciones técnicas y científicas.



Figura 9. Dron Tarot FY690S

Una vez detallado algunos de los drones específicamente para videovigilancia, se especifica algunos de los UAV´s disponibles para la investigación de forma que se detalla las características técnicas con una tabla comparativa para visualizar que UAV son útiles para el sistema de navegación se visualizara en la Tabla 1.

Tabla 1. UAV´s [12] [13] [14]

Características	**UAV multirotor**	**Realacc x210Pro FPV**	**Hexacoptero Tarot FY690S**
			
Peso	249 gr	500 gr	600 gr
Dimensiones	245 mm (ancho) × 290 mm (largo) × 55 mm (alto)	300 mm (alto) x 260 mm (largo) x 100 mm (alto)	600 mm (ancho) × 300 mm (largo) × 100 mm (alto)
Autonomía de Vuelo	12 min	14.33 min	16 min
Velocidad Máxima	5,556 m/s	5,556 m/s	20,83 m/s
Altura máxima de vuelo	120 m	122 m	100 m
Número de motores	4	4	4

Sistema de Batería	Lipo, 1300 mAh, 11.1 V, 14.43 Wh, 75 C 3s	Lipo, 1300 mAh, 11.1 V, 14.43 Wh, 75 C 3s	Lipo, 4-6S, 22.2 V, 6000 mhA, 45 C
Material de infraestructura del UAV	Fibra de carbono PLA	Plástico / PLA	Fibra de Carbono / PLA
Tarjeta controladora de vuelo	MAVERICK 2.0	NAZE32 SP Racing F3	PIXHAWK 2.4.8

1.3.3 Vigilancia y Operaciones del Espacio Aéreo

Los vehículos aéreos no tripulados han experimentado un importante desarrollo; debido al continuo avance tecnológico en los últimos años; tanto en el ámbito de operaciones militares como en el ámbito de aplicaciones civiles; se prevé que, en un futuro cercano, un tercio de las aeronaves civiles serán no tripuladas [15].

El uso de estos sistemas han sido un importante apoyo en el levantamiento cartográfico, fotogramétrico, desarrollo de investigación, operaciones técnicas de inteligencia, exploración aeromarítima y gestión de riesgos; ámbitos de empleo que han demostrado que estos equipos han dejado de ser una herramienta de trabajo para convertirse en una verdadera capacidad en apoyo a las operaciones [15].

Las misiones las cuáles pueden cumplir los drones antes mencionadas son una variedad amplia dependiendo de la aplicación a la cuál procede esto específicamente el uso el cuál ha sido configurado, de acuerdo aquello a continuación se específica las operaciones aéreas para un dron de videovigilancia.

a. Operaciones de fotografía y videografía aérea

Los drones vendrán equipados con sistemas con cámaras de alta resolución que permiten capturar imágenes y videos desde diferentes ángulos o posiciones únicas, siendo herramientas esenciales en el caso de cubrir eventos y promocionar publicidad.

b. Operación de alarma de eventos inesperados

En el caso de requerir avisos de acuerdo con un evento concurrido en algún sitio y se requiera de forma propicia o urgente un monitoreo de la zona los drones se utilizarán para la vigilancia de estas áreas solicitadas de vigilancia, ofreciendo una visión aérea en tiempo real que facilita la toma de decisiones de los operarios del dron.

c. Operación de patrullaje de perímetros y áreas extensas

Los drones son ideales para realizar patrullajes regulares y monitorear perímetros en instalaciones industriales, zonas fronterizas y áreas rurales. Su despliegue permite una vigilancia constante y la detección rápida de intrusiones o actividades ilegales. En el caso dentro de un campus universitario vigilar específicamente las afueras del campus que ayuden a detectar de forma rápida actividades irregulares [16].

d. Operación de vigilancia de grandes eventos y multitudes

En eventos públicos de gran escala, los drones proporcionan vigilancia aérea en tiempo real, facilitando la identificación de actividades sospechosas y mejorando la gestión de multitudes. Esta capacidad es crucial para mantener la seguridad en conciertos, manifestaciones y eventos deportivos [17].

1.3.4 Redes de Videovigilancia

Se definirá la videovigilancia a la captación de imágenes por cualquier medio de grabación, con fines de prevención y persecución de los delitos, esto dará la posibilidad de controlar y tener visión de áreas específicas que no estén bajo cargo de personal autorizado [18].

En el ámbito de la videovigilancia, el fin más comúnmente perseguido, como ya se ha indicado es garantizar la seguridad de personas e instalaciones, los servicios de videovigilancia consisten en el ejercicio de la vigilancia a través de sistemas de cámaras o videocámaras, fijas o móviles, capaces de captar y grabar imágenes y sonidos, incluido cualquier medio técnico o sistema que permita los mismos tratamientos que éstas [19].

Cuando la finalidad de estos servicios sea prevenir infracciones y evitar daños a las personas o bienes objeto de protección o impedir accesos no autorizados, serán prestados necesariamente por vigilantes de seguridad [19].

1.3.5 Tecnologías para la videovigilancia y operaciones

Las tecnologías modifican las fronteras y los campos de acción, debido a ello, los conceptos de seguridad se han reformulado, como se han modificado las formas de vida de los ciudadanos, de igual modo las instituciones han cambiado su mirada con respecto a la problemática delictual [20].

El uso de las nuevas tecnologías en las últimas décadas ha ido en aumento de manera sustancial y esto ha significado un cambio en los hábitos sociales, especialmente comerciales, financieros y comunicativos [20].

Las cámaras CCTV se han convertido en una herramienta esencial en los sistemas de seguridad modernos se podrá visualizar en la Figura 10. Estas cámaras no solo ayudan a prevenir delitos, sino que también pueden servir como evidencia en procedimientos legales, son las cámaras tradicionales que han estado en uso durante varias décadas. Estas cámaras utilizan cables coaxiales para transmitir señales de vídeo a una grabadora de vídeo digital (DVR) [21].



Figura 10. Cámara CCTV puesta en un parapente.

Algunas de las principales tecnologías utilizadas para la videovigilancia incluyen [21].

- Cámaras CCTV HD-CVI que son una actualización de las cámaras analógicas tradicionales.

- Cámaras de CCTV IP que son las últimas en tecnología CCTV. Estas cámaras capturan y transmiten señales de vídeo a través de una red, lo que permite el acceso remoto y el control.

- Cámaras de CCTV térmicas son únicas en el sentido de que capturan imágenes basadas en firmas de calor en lugar de luz visible.

- Cámaras CCTV Pan-Tilt-Zoom (PTZ) que se pueden controlar de forma remota para desplazarse, inclinarse y acercarse a áreas específicas.

Existen varias tecnologías diferentes de cámaras de CCTV disponibles, cada una con sus propias ventajas. Al elegir una cámara CCTV, es importante tener en cuenta las necesidades específicas de su ubicación y el nivel de seguridad requerido. Con la cámara CCTV adecuada, puede mejorar la seguridad de su hogar o negocio [21].

1.3.6 Sistemas de monitorización

Los UAV's de tipo multirotor son sistemas embebidos y eficientes para monitoreo e inspección, razón por la cual la elección de este modelo para este tipo de vehículo es para el desarrollo de aplicaciones como soluciones tecnológicas complementarias de sistemas que ayuden a solución de problemas varios siendo sencillos y complejos de la sociedad, en base algunos de los sistemas de monitorización se tendrán en cuenta algunos como los siguientes.

a. Sistemas tradicionales

El funcionamiento del sistema tradicional se basa su funcionamiento en cámaras preinstaladas que graban y se almacenan imágenes en una base de datos o un sitio web a través de DVR/NVR los cuales van conectados los dispositivos como se indica en la Figura 11, tiene un control total sobre los datos, sin dependencia de internet y un ahorro de costos a largo plazo. Teniendo como limitaciones el almacenamiento limitado y riego de pérdida de datos ya que requiere de mantenimiento cuando el almacenamiento está lleno [22].

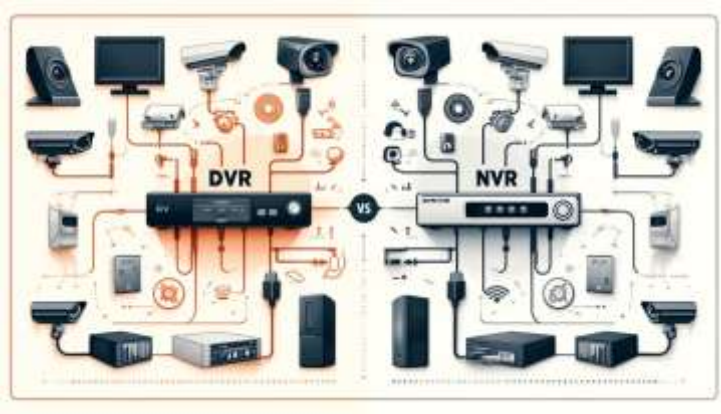


Figura 11. Sistema de grabador de video CCTV.

b. Sistema sin vigilancia (Seguridad solo para personas)

El funcionamiento de este sistema ya es más antiguo utilizado para seguridad de personal en conjuntos residenciales mayoritariamente ya depende únicamente del personal de seguridad para el monitoreo y la respuesta ante la vigilancia representado como la Figura 12. La toma de decisiones humanas es en tiempo real y puede manejar múltiples tipos de incidentes, pero son altos los costos laborales continuos por parte del personal contratado, la cobertura es limitada y se tendrá un potencial error humano y no existe registro visual para futuras referencias o evidencia [22].



Figura 12. Vigilancia por personal contratado

c. Sistemas de vigilancia avanzados (IA, sistemas inteligentes)

El funcionamiento del sistema es basado en una arquitectura como en la Figura 13 donde el almacenamiento es realizado en la nube, aplicando algoritmos de inteligencia artificial y análisis para la automatización y el monitoreo en tiempo real, las alertas

inteligentes y el análisis es en tiempo real, el acceso es remoto y el sistema completo es escalable, teniendo una menor dependencia de la mano de obra y tiempos de respuesta más rápidos. La única desventaja son los altos costos iniciales para montar el sistema completo y el mantenimiento de todo el sistema y se puede tener una dependencia del internet, la irrupción de datos puede ser conflictiva en cuanto a la seguridad digital [22].



Figura 13. Arquitectura de solución de Edificios Inteligentes Hikvision [23].

d. Sistemas de vigilancia con UAV

Un dron de vigilancia es un vehículo aéreo no tripulado (UAV por sus siglas en inglés Unmanned Aerial Vehicle) que puede programarse para realizar rutas específicas en horarios determinados. Estos dispositivos pueden ser controlados de forma remota y están equipados con cámaras de alta resolución que transmiten imágenes nítidas en tiempo real. Su uso abarca desde la seguridad vial hasta la protección de hogares, empresas e incluso la asistencia para los cuerpos de seguridad [24].

La principal función de los drones de seguridad y vigilancia es realizar un seguimiento visual utilizando su cámara incorporada. Estos drones como se muestra en la Figura 14 pueden seguir a personas, grupos, vehículos y estar atentos ante distintas situaciones para prevenir actos vandálicos o amenazas a la seguridad [24].



Figura 14. Dron TAROT 680 PRO, hexacóptero con gimbal integrado

Gracias a su tecnología avanzada, los drones con funciones incorporadas para poder identificar intrusos, prevenir incendios mediante alarmas, controlar fronteras e incluso asistir en rescates en zonas montañosas, como ocurre en el caso de los drones contra incendios. Para su correcto funcionamiento, es esencial una integración adecuada del hardware y software, así como personal capacitado para su manejo [24].

Algunas de las tecnologías más implementadas para los UAV son los sistemas FPV que sirve tanto para el manejo del dron mediante un monitor FPV o gafas FPV, como para videovigilancia es así como se especifica algunos de los módulos FPV de acuerdo con la frecuencia de operación, el número de canales y formato de video el cuál utiliza el módulo de Video transmisión, se podrá visualizar en la Tabla 2 una comparativa de características técnicas de algunos de los módulos.

Tabla 2. Módulos de Video Transmisión [25] [26] [27]

Caracteristicas	**Rush Tank II Ultimate 5.8GHz VTX**	**Yoidesu 1.2G VTX**	**MATEK HV 5.8 Ghz**
			
Frecuencia de Operación	5.8 GHz	1,2 GHz	5.8 GHz
Potencia de transmisión	PIT/25/200/500/800/ mW	PIT 25 200 1600 mW	25, 200, 500 mW
Canales	48 CH	9 CH (1060 MHz-1380 MHz)	40 CH
Formato de video	NTSC	NTSC/PAL	NTSC/PAL

Voltaje de entrada	7-36 V	8-24 V	6~27 V
Dimensiones	36 x 36 x 4,5 mm	35.05 x 35.56 mm	36 x 22 x 6.5 mm
Precio Aproximado	$161 USD	$96 USD	$70 USD

Además de aquello para tener un respaldo y utilizar una cámara de mucho más alta resolución y que permita tener comunicación hacía una estación terrena o de control que sirva de monitoreo y configuración, esta viene con una placa de control de servo motores para tener una perspectiva de grabación de video de varios ángulos utiliza un gimbal este ya sea de 3 o 2 ejes de cardan, se detalla en la Tabla 3 algunos de los gimbals con una comparativa de características técnicas de las placas de control, se tiene en cuenta la placa controladora utilizada para el UAV.

Tabla 3. Gimbals [28] [29] [30]

Caracteristicas	**PAMENET Controlador de cardán sin escobillas BGC 3.1 de 2 ejes**	**BaseCam-SimpleBGC Tiny C**	**SimpleBGC 32bit 3-Axis**
			
Dimensiones	50 mm x 50 mm	40 x 25 x 7 mm	55 mm, agujero de tornillo M3, paso de 45mm
Peso	17 gr	5,45 gr	10 gr
Voltaje Cardán	7,4 - 16,8 V Bateria Lipo 3S	5 - 26 V (2S-6S LiPo)	9 - 18 V(3-4S)
Alimentación Placa	5 V	4.5 V	5 V
Corriente Máxima / funcionamiento	10 A / 5 A	5 A pico ~ 1.5 A continuo por motor	1.5 A
Conectores	MicroUSB, JST / HX	MicroUSB, JST / HX -Conexión Bluetooth	7CH PWM/sum-ppm entrada/salida -Spektrum satélite puerto -Puerto I2C
Soporte Motor	Serie 2x - 8x (para cámara de cine digital GoPro a RED)	controlador de cardán de 3 ejes	Controlador de cardán de 3 ejes
Version Firmware	BGC2.2	SBGC32	DRV8313
Precio Aproximado	$22,40 USD	$192 USD	$39.70 USD

1.3.7 Tecnología y Sistema de navegación Autónoma

Las aeronaves autónomas son los aparatos voladores capaces de desarrollar una función de forma completamente independiente, sin intervención humana de ningún tipo. Para ello se requiere un entorno conocido a través del cual se diseñan unas rutas para la navegación del UAV o bien un entorno desconocido que es "sensado" por el aparato durante el vuelo, estos son utilizados para decidir y ejecutar las acciones pertinentes para cumplir una serie de objetivos, para ello se especifica algunos de las tecnologías y sistema para navegación con UAV [31].

a. ROS

Robot Operating System, proporciona una estructura de programación para el desarrollo enfocado hacia la robótica se define como técnica que aplica la informática al diseño y empleo de aparatos que, en sustitución de personas, realizan operaciones o trabajos, por lo general en instalaciones industriales siendo de código abierto. Este sistema se utiliza en aplicaciones dirigido a la investigación, también cuenta con la abstracción de hardware, el control de dispositivos de bajo nivel, la implementación de funciones de uso común, el paso de mensajes entre procesos y la gestión de paquetes. También proporciona herramientas y bibliotecas para obtener, construir, escribir y ejecutar código en varias computadoras [32].

El Sistema Operativo (ROS) ha revolucionado la comunidad de desarrolladores, proporcionándole un conjunto de herramientas, infraestructura y las mejores prácticas para construir nuevas aplicaciones y robots. ROS permite el diseño de manera individual de procesos que se pueden acoplar de manera flexible en tiempo de ejecución. Estos procesos se pueden agrupar en paquetes que pueden ser fácilmente compartidos y distribuidos, con lo que se logra la reutilización de código. Actualmente ROS solo puede ser ejecutado en plataformas basadas en linux [32].

b. ORB-SLAM

El algoritmo ORB-SLAM es un sistema monocular en tiempo real de localización y mapeo simultaneo basado en la detección de puntos de interés representados por descriptores ORB (del inglés, Oriented FAST and Rotated BRIEF). El sistema es

robusto a movimientos erráticos, provee las capacidades de relocalización y cierre de trayectoria, así como una inicialización automática de los puntos del ambiente. La Figura 15 muestra la imagen de correspondencia del seguimiento de descriptores en el ambiente real observado por la cámara.



Figura 15. Descriptores en el entorno real empleando ORB-SLAM.

La Figura 16 ilustra la representación del mapa de puntos en R3 del ambiente y la localización de la pose de la cámara estimada a través del algoritmo ORB-SLAM [33].

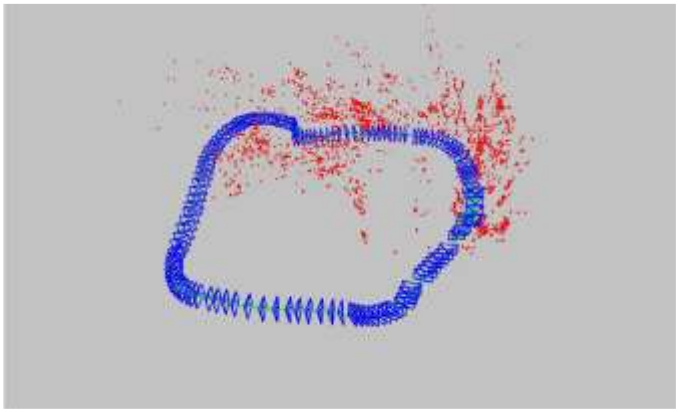


Figura 16. Mapa y localización empleando ORB-SLAM.

El algoritmo ORB-SLAM tiene tres componentes principales, el seguimiento, el mapeo local y el cierre de trayectoria usando descriptores visuales ORB. La etapa de seguimiento es la encargada de estimar las poses de la cámara mediante la búsqueda de correspondencias de los puntos de interés del mapa entre los frames anteriores y el frame actual. Además, la etapa de seguimiento también toma la decisión de cuando insertar un keyframe para actualizar el grafo basándose en el cambio de información

visual entre el keyframe actual y los anteriores. El componente de mapeo local es el que agrega nuevos puntos al mapa y los mantiene. La optimización del mapa construido se hace mediante Bundle Adjustment. Finalmente, la etapa del cierre de trayectoria se emplea cuando un lugar es revisitado [33].

c. Visión artificial

La visión artificial es un área de investigación que engloba los procedimientos y componentes que habilitan a una máquina para desarrollar habilidades visuales. En términos sencillos, se puede definir la visión artificial o comprensión de imágenes como el proceso de analizar una o múltiples imágenes de un entorno tridimensional con el propósito de describir sus características y propiedades [34].

- **Fundamentos de la visión artificial**

Una imagen se define como una representación visual de un objeto que se encuentra iluminado por una fuente de radiación. Las imágenes que percibimos en nuestras actividades diarias suelen originarse a partir de la luz reflejada por los objetos. La formación de una imagen implica diversos factores, que incluyen la fuente de radiación, el objeto en sí y el sistema responsable de generar la imagen. Este sistema consta principalmente de componentes como un sistema óptico, un sensor y un digitalizador [34].

- **Componentes de un sistema de visión artificial**

Los elementos fundamentales de hardware necesarios para implementar la visión artificial son los siguientes: un sensor óptico, que puede ser una cámara monocroma o una cámara de color, capaz de capturar una imagen completa del área de interés al menos cada 1/30 segundos. Igualmente se puede utilizar un escáner de línea como sensor óptico, donde el objeto en movimiento a lo largo de la línea de escaneo produce una imagen bidimensional [34].

Una tarjeta de adquisición de datos se encarga de digitalizar la señal de vídeo proveniente del sensor óptico mencionado anteriormente. Por último, se requiere un computador donde se almacene la imagen digitalizada para su posterior procesamiento

y manipulación mediante un programa especializado. El monitor de video o pantalla permite visualizar las imágenes, así como los resultados del procesamiento [34].

En la Figura 17, se observa un diagrama de bloques el cual muestra un sistema de visión artificial según lo descrito anteriormente [34].



Figura 17. Diagrama de bloques de un sistema de visión artificial.

d. Drone kit y MAVlink

DroneKit es un conjunto de herramientas de software que permite el desarrollo de aplicaciones autónomas y de control de drones que utilizan la plataforma ArduPilot o PX4. Con DroneKit, los desarrolladores pueden escribir aplicaciones en Python o en otros lenguajes de programación para interactuar con el hardware del dron a través de una API. Esto facilita la creación de scripts personalizados que permiten realizar tareas como el control de vuelo, la navegación y la obtención de datos en tiempo real [35].

MAVLink o "Micro Air Vehicle Link" es un protocolo para comunicación con pequeños vehículos no tripulados (UV) pequeños. Es utilizado principalmente para la comunicación entre la estación de tierra y el vehículo no tripulado, así como para la intercomunicación entre los subsistemas del vehículo. Este protocolo puede ser utilizado para transmitir datos como la orientación del vehículo, su ubicación GPS y la velocidad que lleva, así como para enviar órdenes a la placa controladora [35].

Dentro de estos protocolos cabe detallar que la placa controladora de vuelo del UAV es fundamental ya que, de acuerdo a ciertas características de la misma, tendrá cierta

compatibilidad con los protocolos de comunicación utilizados en cada placa además de tener la posibilidad de la incorporación de más sensores, GPS, conexión con diferentes placas de control o módulos, se detalla algunas de las placas que tienen los drones de la Tabla 1 las cuales se especifica las características técnicas de forma que se plantea una tabla comparativa se podrá observar en la Tabla 4.

Tabla 4. Placa controladora de vuelo [36] [37] [38].

Características	**Controlador de vuelo Pixhawk 2.4.8**	**Ardupilot APM 2.8**	**NAZE32 SP Racing F3**
			
Procesador	STM32F427VIT6 de 32 Bits	*ATMEGA2560*	STM32F3
Sensores /Puertos	-Giroscopio MPU600 -E-Compass OMC3883L -Puerto xh socket -Puerto Mavlink UART(HC) -2 CON CTSRTS -Bocina -ADC port -GPS UART - I2C -Salidas PWM -RC in (TX señal de soporte)	-Barometro MS5611-01BA03 -Giroscopio de 3 ejes - GPS -Aceletrometro -Magnetómetro. -3DR Radio Telemetría Kit 915 Mhz -Mavlink-osd V2.0 -Enlace serial -Soporte para LBT y AFA -Conexión I2C -Puerto PWM	-Barómetro MS5611 -Brújula HMC5883 -Acro 6DOF y Deluxe 10DOF -Acelerómetros y giroscopios MPU6050 - Pines JST-SH Socket (I2C, SWD, 2xuart) - Pines JST-SH Socket (PPM, PWM, SERIAL RX, GPIO, ADC, 3V, 5V, GND)
Conectividad	Puerto USB -Conexión Puerto de telemetría 1/2	Puerto serial USB-B -Puerto de telemetría	-Micro USB
Almacenamiento	MicroSD 18GB	MicroSD 12GB, Memoria de registro de 4MB	Datos Memoria FLASH
Peso	38 gr	50 gr	6 gr
Precio Aproximado	$159 USD	$77,10 USD	$10 USD
Alimentación	Power Module – 5.1 V	5 V (conector FTDI) y 12 V (conector Video I/O)	5 V / 12 V
Dimensiones	82,2 mm x 51,8 mm x 16,5 mm	30 mm x 15 mm x 10 mm	0,5 mm x 0,5 mm x 0,05 mm

e. Sensorización LIDAR

El origen de la tecnología LiDAR se remonta a la década de 1930, cuando se realizaron los primeros intentos de medir los perfiles de densidad del aire en la atmósfera, utilizando la intensidad de dispersión de haces de luz. En estos experimentos, la altura se determinaba a través del campo de visión del receptor de un telescopio colocado a cierta distancia a lo largo del haz de luz. Con el tiempo, después de la invención del láser en 1960 y del láser con Q-switch en 1976, la tecnología LiDAR experimentó un rápido desarrollo, impulsando numerosos estudios y demostraciones que contribuyeron a su evolución. Así, cada avance en LiDAR fue acompañado por mejoras en las tecnologías ópticas y electrónicas.

El sistema LiDAR se compone de un transmisor y un receptor; el transmisor emite rayos de luz intensos y enfocados, mientras que el receptor mide el tiempo que tardan en regresar los haces de luz reflejados. Este cálculo se realiza midiendo la diferencia de tiempo entre la emisión del pulso y su retorno, además del ángulo de emisión y la ubicación absoluta del sensor en la superficie terrestre. La tecnología LiDAR tiene múltiples aplicaciones según el tipo de sensor y su modo de operación, ya sea en plataformas móviles o fijas, tanto en aplicaciones aéreas como terrestres. En este caso de estudio, se considerará el uso de la tecnología LiDAR en aplicaciones terrestres.

f. GPS (Georreferencia)

El sistema de geoposicionamiento en drones es una de las tecnologías más usadas en UAV´s, ya que permite que el dron pueda realizar sus funciones o acciones de una forma más precisa y segura. Detallando como una definición de GPS (Sistema de Posicionamiento Global) es un sistema satelital que permite determinar la ubicación geográfica precisa de un objeto en cualquier parte del mundo [39].

Básicamente su funcionamiento se debe a una red de satélites que orbitan la Tierra y que transmiten señales a dispositivos receptores, que utilizan dichas señales para triangular su posición exacta [39].

El GPS en un dron permite que se conozca su ubicación precisa en todo momento. Actúa como un sistema de navegación y posicionamiento, ofreciendo información crucial al piloto o al software de control de vuelo. Con la ayuda del GPS, un dron puede realizar vuelos con total autonomía, seguir rutas predefinidas con precisión y mantener una posición estable, ya que podrá determinar todo lo que lo rodea en tiempo real [39].

De forma, que la tecnología GPS se podrá implementar módulos GPS comerciales y con características que se adecuen al sistema respecto a un modelo requerido con la recepción de satélites y tecnología soportada exacta que sea óptima para el sistema de navegación autónoma del UAV. Se presenta la Tabla 5 como una comparación de características técnicas de alguno de los módulos más utilizados para UAV´s.

Tabla 5. Módulos GPS [40] [41] [42].

Caracteristicas	**Módulo GPS NEO-8M**	**Holybro Micro M8N**	**M8 Engine PX4**
			
Peso	17 gr	20,6 gr con estuche	32 gr
Voltaje de Alimentaciòn	3-5 VDC	3 – 3,3 VDC	3.5 a 5.5 VDC
Conexión con Placa controladora	Serial UART	Serial UART	Serial UART / I2C
Antena Tipo	Antena patch cerámica activa	Antena patch cerámica	Antena de parche cerámico de 25x25x4mm
Frecuencia de operación	1575.42 MHz	1575.42 MHz	1575.42 MHz
Dimensiones	25 mm x 35 mm	38 mm x 38 mm x 11mm	50 mm
Tecnología soportada	Soporta SBAS (WAAS, EGNOS, MSAS, GAGAN)	Sistemas GNSS	GNSS de alta precisión, soporta GPS, GLONASS, Galileo y BeiDou.
Presición	2.5 m en condiciones óptimas	2.5 m en condiciones normales	2,5 m en condiciones estandar
Precio Aproximado	$90 USD	$40 USD	$48,99 USD

1.3.8 Características de cobertura de videovigilancia

Las características de la cobertura de la videovigilancia de un UAV pueden incluir algunas descritas en la Tabla 6:

Tabla 6. Características de la cobertura de la videovigilancia de un UAV [34].

CARACTERÍSTICAS	DESCRIPCIÒN
Alcance y área de cobertura	Para el monitorear y vigilar áreas específicas, esto varía dependiendo de la altitud a la que vuela y la duración del vuelo
Resolución de las imágenes	La calidad de las imágenes capturadas por las cámaras puede afectar la claridad y el detalle de la información visual recopilada
Flexibilidad y maniobrabilidad	Para ajustar su posición y ángulo de visión con el fin de obtener la mejor cobertura posible de la zona objetivo.
Estabilidad y control	La estabilidad en vuelo y la precisión al control de sus movimientos, lo que influye en la calidad y consistencia de las imágenes capturadas.
Adaptabilidad a las condiciones ambientales	La operación del UAV de manera efectiva en diferentes condiciones climáticas y ambientales, como viento, lluvia o niebla.
Duración del vuelo	El tiempo que puede permanecer en el aire antes de necesitar recargar las baterías o cargar combustible.
Capacidad de transmisión de datos	La capacidad para transmitir en tiempo real o almacenar las imágenes y datos recopilados para su análisis posterior.

1.3.9 Alcance

El alcance de un UAV, conocido como vehículo aéreo no tripulado, representa la distancia máxima que puede recorrer desde su lugar de partida antes de perder la conexión o agotar su energía. Este alcance puede variar significativamente según diversos factores, como el diseño del UAV, su tamaño, la capacidad de carga que puede transportar, así como el tipo de combustible o energía que emplea, y las regulaciones locales que rigen el vuelo de estos dispositivos [43].

Los UAV’s se dividen en distintas categorías, abarcando desde modelos de entretenimiento con un alcance limitado de unos pocos cientos de metros, hasta UAV’s de gran tamaño utilizados en aplicaciones militares o de vigilancia, capaces de cubrir distancias de miles de kilómetros. Por ejemplo, los drones recreativos más comunes

generalmente tienen un alcance de hasta varios kilómetros, en contraste, los UAV's de largo alcance empleados en tareas militares pueden realizar vuelos intercontinentales, con sistemas de reabastecimiento en vuelo o avanzada navegación autónoma [43].

La autonomía de vuelo de un UAV también puede estar restringida por la duración de su batería o la capacidad de su tanque de combustible. Por ejemplo, los UAV's eléctricos pueden volar desde unos pocos minutos hasta varias horas, dependiendo de factores como su peso total, la velocidad de vuelo y las condiciones climáticas. Por otro lado, los UAVs de combustible, como los aviones no tripulados de larga distancia, pueden tener una autonomía mucho mayor, ya que pueden reabastecerse de combustible durante el vuelo o transportar grandes cantidades de combustible líquido [43].

1.3.10 Velocidad

La velocidad de un UAV, o vehículo aéreo no tripulado, indica la rapidez con la que puede desplazarse en el aire durante su vuelo. Esta velocidad está sujeta a variaciones considerables, dependiendo de diversos aspectos como el diseño del UAV, su tamaño, su configuración aerodinámica, el tipo de propulsión que emplea y su propósito[43].

Los UAVs se clasifican en diferentes categorías según su velocidad, la velocidad es una característica principal dentro de estos dispositivos ya que medirán la eficiencia de recolección de datos ya sea por cámaras o sensores, entonces se podrá observar algunas de las categorías teniendo en cuenta el nivel de velocidad que tenga esa aeronave en la Tabla 7.

Tabla 7. Categorías de los UAV`s [43]

CATEGORIAS	DETALLES
UAVs de baja velocidad	Aplicado a vigilancia, cartografía y fotografía aérea con velocidad reducida (algunos km/h hasta pocas decenas km/h).
UAVs de velocidad media	Para misiones de velocidad moderada, monitorización de áreas extensas o distribución de suministros (50-200 km/h).
UAVs de alta velocidad	Para tareas de rápida respuesta como vigilancia de fronteras o reconocimiento táctico (>200 km/h).

La velocidad de un UAV también puede ser influenciada por factores tales como las condiciones meteorológicas, la altitud de vuelo y la carga que transporta. Además, algunas aplicaciones pueden requerir que el UAV mantenga una velocidad constante y estable, mientras que otras pueden necesitar maniobras rápidas y cambios de dirección [43].

De acuerdo con condiciones atmosféricas y ambientales las características del UAV variaran debido al aumento y disminución de velocidad esto para realizar trayectorias o maniobras adecuadas para no perder la estabilidad del vuelo, entonces deben ser muy efectivas en cuanto ajuste y velocidad para tomar pruebas del terreno en un alcance determinado como se puede observar en la Figura 18 [43].



Figura 18. Alcance de un UAV.

1.3.11 Precisión y alcance de los Sistemas de Navegación Autónoma

La capacidad de respuesta frente a incidentes de un UAV se refiere a la aptitud del dron para actuar con rapidez y eficacia frente a eventos imprevistos o emergencias durante su funcionamiento. Esto implica su habilidad para identificar y manejar adecuadamente situaciones como problemas técnicos, condiciones climáticas adversas, intrusiones no autorizadas en el espacio aéreo, u otros eventos inesperados que puedan surgir durante su tarea. Esta capacidad puede involucrar la implementación de protocolos de seguridad establecidos, sistemas de alerta temprana, procedimientos de comunicación y coordinación con operadores humanos, así como habilidades de navegación y control que le permitan al UAV tomar medidas correctivas o evasivas según sea necesario para minimizar riesgos y garantizar la seguridad de la operación.

De acuerdo con la cuál estima las cualidades de los incidentes que puede ocurrir en un dron entonces se muestra estas características en la Tabla 8 [44].

Tabla 8. Características de capacidad de respuesta ante incidentes del dron [44].

CARACTERÍSTICAS	DESCRIPCIÓN
Detección anticipada	La habilidad del dron para rápidamente identificar posibles inconvenientes.
Sistemas de aviso	La existencia de sistemas que alerten a los operadores sobre cualquier irregularidad detectada.
Flexibilidad en el control	La capacidad del UAV para adaptar su ruta, velocidad o altitud de vuelo.
Procedimientos de seguridad	Normativas previamente establecidas y protocolos de seguridad.
Comunicación eficaz	La habilidad para establecer y mantener una comunicación con los operadores.
Capacidades de maniobra	Destrezas de navegación y control que permitan al UAV realizar maniobras evasivas con rapidez.
Redundancia y respaldo	La presencia de sistemas redundantes y medidas de respaldo que aseguren la continuidad de la operación del UAV.



Figura 19. Sistemas de navegación con drones.

1.3.12 Implementación de Tecnologías de Navegación Autónoma

Las tecnologías implementadas en un UAV son específicamente escogidas de acuerdo con el modelo y placa la cuál acepte el tipo de sensorización o equipos para el modelado específico para navegación. Una vez especificado los métodos que se utiliza mencionando específicamente las tecnologías que se utiliza para navegación autónoma para los UAV´s, de forma que se detallarán modelos y características de los elementos los cuales se mencionan para poder tener en cuenta las características técnicas de cada

uno de los elementos que se podrá utilizar y existen para poder implementar en el sistema deseado, a continuación se muestran las tablas comparativas de algunos de los métodos detallados en la Tabla 9 que serán aplicados en la investigación.

Tabla 9. Métodos para navegación autónoma en UAV. [44]

MÉTODO	DESCRIPCIÓN
Sensores de detección	Permiten que los vehículos autónomos detecten y comprendan su entorno para tomar decisiones seguras y bien fundamentadas.
Cámaras	Capturan imágenes del entorno para identificar objetos y señales visuales en diferentes espectros.
Radar	Utilizan ondas de radio para detectar objetos y medir su velocidad y dirección, útil en baja visibilidad.
Ultrasonido	Emite ondas sonoras para detectar objetos cercanos, ideal para maniobras de estacionamiento.
Sensores inerciales	Incluyen acelerómetros y giroscopios para medir la aceleración y orientación del vehículo.
Sensores de proximidad	Detectan objetos cercanos mediante señales electromagnéticas o ultrasónicas, esenciales para evitar colisiones.

a. Cámaras

La cámara específicamente para el sistema de navegación autónoma se analiza las cuales se adecuen a los requerimientos del sistema, además de poder utilizar para el sistema de videovigilancia otro tipo de cámaras, en la Tabla 10 se detallan las características técnicas de algunas cámaras.

Tabla 10. Cámaras [45] [46] [47].

Caracteristicas	Cámara GoPro Hero 5	Caddx Cámara	Mini 1200TVL FPV cámara NTSC
			
Dimensiones	62 × 44.6 × 32.7mm	38 × 38 ×36 mm	24.98 mm x 24.98 mm
Peso	117 g	62.8 g	28,34 g
Batería	1220 mAh	600 mAh	Conexión a placa de FPV
Duración de Batería	150 minutos en su máxima calidad	30 minutos	Conexión a placa de alimentación de Video
Memoría	Micro SD hasta 128GB	MicroSD Máx. 256GB	Sin memoria
Resolución de Imagen	12 megapíxeles	4K/60 fps	1200TVL de alta definición para capturar imágenes vívidas y lera
Conectividad	Wifi, Bluetooth, GPS, HDMI	Wi-fi, conexión serial NTSC, UART	NTSC, Serial
Estabilización	Soporte de 4 ejes	Compatible con Gyroflow + ElS	Manteniendo con un soporte fijo
Precio Aproximado	$285 USD	$199 USD	$24 USD

1.3.13 Sistemas de Ubicación

a. Transmisión de Datos

La transmisión de datos en tierra del UAV implica el intercambio de información entre el vehículo aéreo no tripulado (UAV) y las estaciones terrestres durante sus operaciones, se especifica algunas características en la Tabla 11 sobre los sistemas del UAV [44].

Tabla 11. Características del sistema de ubicación del UAV [44]

CARACTERÍSTICAS	DETALLE
Sistemas de comunicación a bordo del UAV	El UAV está equipado con sistemas de comunicación, como radios, enlaces de datos por satélite, o tecnologías inalámbricas como Wi-Fi o Bluetooth.
Datos recopilados por el UAV	El UAV recolecta diversos tipos de datos, como información de navegación (ubicación GPS, altitud, velocidad), datos telemétricos (estado de la batería, temperatura, condiciones ambientales) y datos sensoriales (imágenes, videos, datos de sensores como Lidar o cámaras infrarrojas).
Protocolos de comunicación	Estos protocolos pueden incluir redes TCP/IP, técnicas de encriptación y compresión de datos para optimizar la eficiencia y seguridad de la transmisión.
Control remoto y monitoreo en tiempo real	Permiten a los operadores en tierra monitoreo en tiempo real del estado, ubicación e imágenes y videos del entorno.
Análisis y almacenamiento de datos	Los datos transmitidos por el UAV pueden ser almacenados y analizados posteriormente para evaluar su rendimiento, identificar patrones y tendencias, y tomar decisiones para futuras misiones.

Entonces este proceso es esencial para permitir el control a distancia, el monitoreo en tiempo real y el análisis de los datos recopilados por el UAV, se muestra algunos de los sistemas que puede estar integrado en la Figura 20 [44].



Figura 20. Sistemas integrados en un UAV

Algunos de los elementos de comunicación para envió de datos entre la estación terrena y el UAV, son módulos de comunicación que utilizan radiofrecuencia, wifi o microondas, pero el módulo más utilizado específicamente en UAV´s es el módulo de telemetría para ello se especifica alguno de ellos en la Tabla 12 especificando algunas características técnicas necesarias que se deben tener en cuenta.

Tabla 12. Módulos de Telemetría [48] [49] [50].

Caracteristicas	**HolyBro SiK radio de telemetría**	**Módulo de MAVlink Wifi bridge**	**Telemetría LoRa**
			
Frecuencia de Operación	915 MHz / 433 MHz	2,4 GHz - 2,5 GHz (2400 MHz – 2483 MHz)	915 MHz (890-915 MHz)
Protocolo de Comunicación	MAVLink (UART)	802,11 b/g/n Wifi	UART IC, LoRaWAN, LoRa
Rango de Alcance	Hasta 10000 m	30 a 250 m (exteriores)	30000 m (en función del entorno)
Velocidad de Tx de datos	250 kbps	11 - 54 Mbps	3.2 - 20 kbps
Consumo de energía	Consumo de Tx 100 mAh Consumo de Rx 25 mAh	20 - 110 mAh	10 - 30 mAh
Integración en Drones	Alta compatibilidad con Pixhawk / APM / Matek	Pixhawk, APM2.5/2,6, Mini APM (cable especial solicitado)	Pixhawk/Ardupilot/PX 4
Precio Aproximado	$40 - $70 USD	$15 USD	$59 USD

1.3.14 Aspectos de navegación

Estos sistemas permiten al dron navegar de manera autónoma, adaptándose dinámicamente a su entorno y asegurando operaciones precisas y seguras, lo cual se describe en la Tabla 13 describiendo los tipos de sistemas de acuerdo con las situaciones requeridas para cada uno de los modelos que se requiera implementar [51].

Tabla 13. Tipos de sistemas de navegación autónoma [51].

SISTEMAS DE NAVEGACIÓN	DESCRIPCIÓN
Sistemas de posicionamiento y navegación	Los UAV modernos suelen emplear sistemas GPS para ubicarse en tiempo real, para la navegación autónoma.
Sensores y percepción del entorno	Los UAV pueden utilizar diversos sensores, como cámaras, lidar y sistemas de detección de obstáculos.
Planificación de trayectorias	Los UAV pueden contar con software de planificación de rutas que genera automáticamente trayectorias óptimas.
Integración con sistemas de control y comando	La navegación autónoma del UAV se puede sincronizar con sistemas de control en tierra.
Comunicación y coordinación	La integración con sistemas de comunicación bidireccional posibilita el envío de datos, así como la recepción de instrucciones y comandos de vuelo actualizados.

1.3.15 Frecuencia de operación en drones en el Ecuador

En Ecuador, la operación de drones está regulada por la Dirección General de Aviación Civil (DGAC), que establece lineamientos para su uso seguro y responsable. Los drones operan en frecuencias de 2.4 GHz y 5.8 GHz, permitidas para garantizar un adecuado alcance y minimizar interferencias. Según la normativa vigente, es obligatorio registrar los drones ante la DGAC y obtener permisos específicos si se utilizarán con fines comerciales o si el peso del equipo supera los 2000 g. Además, el reglamento prohíbe el vuelo en zonas cercanas a aeródromos, helipuertos, áreas protegidas y de seguridad del Estado, exigiendo una distancia mínima de nueve kilómetros de estos espacios. Estas medidas buscan garantizar la seguridad aérea y la protección de espacios sensibles, permitiendo el desarrollo de operaciones dentro de un marco regulatorio claro y específico [52].

1.4 Objetivos

1.4.1 Objetivo general

Desarrollar un Sistema de navegación autónoma de un UAV para videovigilancia y operaciones del espacio aéreo de la Universidad Técnica de Ambato.

1.4.2 Objetivos específicos

- Determinar las tecnologías en los sistemas de navegación autónoma de un UAV (Vehículo Autónomo No Tripulado) para videovigilancia en la Universidad Técnica de Ambato.
- Seleccionar la tecnología más adecuadas para la implementación de control por inteligencia artificial o control electrónico tradicional en los sistemas de navegación autónoma de un UAV.
- Implementar un sistema de navegación autónoma de drones para videovigilancia en la Universidad Técnica de Ambato

CAPÍTULO II. METODOLOGÍA

2.1 Materiales

Para el desarrollo del proyecto de un sistema de navegación autómata de un dron para videovigilancia y operaciones en el espacio aéreo, se emplearon materiales clave que garantizaron tanto su diseño como su funcionamiento adecuado. Entre los principales equipos utilizados destaca el dron Tarot 680 Pro, que sirvió como base para la integración de los componentes necesarios. La placa controladora de vuelo PIXHAWK 2.4.8 se utilizó como componente central para conectar los motores brushless, con los controles de velocidad Electrónico (ESC) y el módulo de telemetría para la comunicación bidireccional con la estación terrena. Además, se incluyó un radioreceptor FrSky R-XSR para el control remoto que es un Radiomaster TX-16S, un buzzer para alertas, un sensor de nivel de batería y el módulo GPS MN-8.

Como fuente de alimentación, se usó una batería Lipo de 22.2 V y 6 A de 6 celdas para el sistema principal y para el sistema del gimbal otra batería Lipo de 11.1 V y 1.5 A de 3 celdas. El dron también cuenta con un sistema FPV para transmisión en tiempo real y un gimbal controlador de dos ejes BGC de 32 bits que proporciona estabilidad a la cámara GoPro Hero 5, necesarias para capturar imágenes de videovigilancia y realizar operaciones específicas en el espacio aéreo. Finalmente, como estación terrena se utilizó una Laptop Asus G512, que permitió gestionar y monitorear las funciones del UAV de manera eficiente, también se utilizó un monitor FPV para capturar el video en tiempo real del sistema FPV integrado en el dron.

Para la parte de la configuración de los elementos previamente mencionados, se utilizó Mission Planner de Ardupilot como base para la configuración de la placa de control de vuelo, además de una interfaz de control y monitoreo creada en Python utilizando los dos softwares tanto para trazar las trayectorias y para monitorear el comportamiento del dron. Adicionalmente, se utilizó para el análisis y simulación de trayectorias y cálculos para el vuelo estable del UAV programación en Python esto editado con el editor de código VISUAL STUDIO CODE y programación en Matlab. Así también para el modelado de la estructura del UAV se usó FUSION360. Estos

recursos permitieron cumplir con la funcionalidad del sistema automatizado de vuelo del UAV.

2.2 Métodos

2.2.1 Modalidad de la investigación

Para el desarrollo del proyecto de investigación, se realizó las siguientes modalidades:

El presente proyecto se llevó a cabo una modalidad de investigación aplicada, donde se aprovechan los conocimientos adquiridos en la formación académica para realizar investigaciones que sustentaron el desarrollo del proyecto. Se llevó a cabo una búsqueda y recopilación de información sobre la tecnología a implementarse en la automatización de trayectorias, se escogió el sistema electrónico por tecnología GPS utilizando una placa central controladora de vuelo capaz de implementar más módulos para automatización de trayectorias para los drones.

La metodología de investigación adoptada fue predominantemente bibliográfica, dado que se centró en la revisión y análisis de tesis, artículos científicos, publicaciones en revistas y libros relacionados con la optimización de trayectorias en UAV´s.

Además, se realizó una investigación de campo, que implico la validación del sistema automatizado con tecnología GPS para el UAV. El objetivo fue automatizar las trayectorias de un dron para videovigilancia y operaciones aéreas, esto mejoró la seguridad y realización de ciertas tareas o acontecimientos dentro del recinto universitario de la UTA.

2.2.2 Recolección de información

La obtención de datos se realizó mediante la consulta de libros, revistas, recursos en línea como repositorios de tesis o artículos científicos, bibliotecas digitales y proyectos o trabajos de investigación previamente desarrollados relacionados con la temática del proyecto. Además, se tomaron en cuenta sistemas autónomos ya creados para el manejo de vuelo con trayectorias predefinidas, utilizándolos como punto de partida para su mejora. En el caso de la videovigilancia, se buscó implementar estos sistemas

con algoritmos de aprendizaje profundo. Asimismo, se especificaron algunos detalles adicionales sobre otros recursos que se utilizaron para la recolección de información del proyecto.

Tabla 14. Recolección de Información

Criterio	Descripción
¿Para qué?	Al identificar el método más idóneo, se garantizó que el sistema sea eficiente, rentable e interoperable con la infraestructura preexistente, lo cual promovió una implantación exitosa y un fortalecimiento de la seguridad y operaciones aéreas o realización de tareas con el UAV dentro del campus.
¿De qué personas u objetos?	De personas aptas con el conocimiento del UAV pertenecientes al campus universitario, y agentes que brinden seguridad como guardias dentro del campus universitario.
¿Cómo?	Mediante un sistema automatizado de vuelo para un UAV.
¿Con qué?	Con un sistema de control de vuelo, por tecnologías relacionadas a Georreferencia.
¿Cuándo?	En el periodo Septiembre 2024 - Febrero 2025.
¿Dónde?	En los parqueaderos y los talleres tecnológicos de la FISEI del campus Huachi de la UTA.

2.2.3 Procesamiento y análisis de datos

Durante el procesamiento y análisis de datos, se llevó a cabo las siguientes etapas:

- Recolección de la información.
- Revisión de todas las fuentes de información relacionadas con el empleo de sistemas automatizados con drones.
- Evaluación de las distintas propuestas presentadas para la elección de implementación de algoritmos de aprendizaje profundo o sistemas electrónicos en el uso de sistemas automatizados en drones.
- Pruebas de funcionamiento del sistema.

Verificación de los datos obtenidos a través de las pruebas realizadas para identificar y corregir posibles errores en el sistema implementado del dron.

CAPÍTULO III. RESULTADOS Y DISCUSIÓN

3.1 Desarrollo de la propuesta

El desarrollo del proyecto se llevó a cabo en las instalaciones de la Universidad Técnica de Ambato, campus Huachi, específicamente en los talleres tecnológicos de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial (FISEI) y en otro espacio dentro del campus en el estacionamiento de vehículos entre el estadio de la universidad. Este entorno proporcionará un espacio controlado para la experimentación y pruebas del UAV, donde se tuvo en cuenta obstáculos y trayectorias predeterminadas para evaluar el sistema de evasión de obstáculos automatizado. Las limitaciones geográficas y de espacio se ajustarán al área de los talleres, asegurando que el dron opere en un entorno controlado y seguro. Además, se detallará la elección de los elementos del sistema de acuerdo con ciertas características específicas escogidas acorde con modelos comerciales de los sensores, cámaras, equipos, placas programables y tecnologías específicas del sistema autónomo del UAV completo.

De acuerdo a las tecnologías existentes actualmente en sistemas de navegación autómatas para drones, que se explican algunas de ellas en el apartado 1.3.7 se adapta dependiendo del uso de cierta placa de control de vuelo que se utiliza en el UAV, además de los alcances y diferentes equipaciones que se pueda añadir al dron, esto siendo más robusto o con más posible añadidura de herramientas con otras placas embebidas con tecnología actual, siendo la tecnología más concurrida los sistemas GNSS que ayudan en las trayectorias autómatas y posibles modos de vuelo con el UAV a utilizar.

Para la selección de la tecnología a utilizar se escoge de acuerdo al análisis de entre diferentes proyectos de investigación realizados precisamente para drones con navegación autónoma aplicados en diferentes campos como se visualiza en el Anexo H, se comparó entre sistemas de control con modelos de IA (Inteligencia Artificial) y sistema de control tradicional electrónico, los parámetros más importantes a tener en cuenta son el tipo de dron utilizado, la placa de control usada y la eficiencia del sistema, entonces de esta forma se escogió el sistema de control tradicional electrónico

por una eficiencia alta del 92% ante los otros sistemas, esto mediante tecnología GPS con el protocolo MAVLink para el control del UAV el mismo provee herramientas para implementar un sistema de georreferencia óptimo esto una vez revisado referencias de investigaciones previas.

3.1.1 Delimitación del terreno

Se llevo a cabo en un terreno delimitado en el área de interés, resaltada como en la Figura 21, abarca un polígono de forma rectangular que contiene una estructura central rodeada por calles y edificios. Las coordenadas geográficas proporcionadas por la herramienta de Google Earth aseguran la precisión en la delimitación del perímetro, que será esencial para definir la trayectoria de vuelo del dron y los puntos clave para la cobertura de videovigilancia. Este terreno representa la zona de operaciones donde el sistema autónomo se desplego para realizar tareas de monitoreo y seguridad, para tener un ambiente con un entorno controlado se tomará en cuenta los alrededores de los talleres de la FISEI ubicado en el campus de la Universidad Técnica de Ambato.



Figura 21. Vista en 3D de Google Earth - Talleres FISEI

Como segundo lugar se tuvo en cuenta para las pruebas del modelo el estacionamiento de la Universidad Técnica de Ambato Figura 22. Vista en 3D de Google Earth – Parqueaderos Universidad Técnica de AmbatoFigura 22. El área se destaca por su uso principal como estacionamiento, lo que crea tráfico de vehículos y dinámicas humanas que el dron puede monitorear. Las coordenadas exactas y el perímetro son claros, lo

que facilita la planificación de rutas de vuelo y lugar para monitoreo o vigilancia completa. La ubicación del sitio en un entorno universitario a estudiar también añade consideraciones especiales de seguridad y privacidad que deben tenerse en cuenta al diseñar este sistema.



Figura 22. Vista en 3D de Google Earth – Parqueaderos Universidad Técnica de Ambato

3.1.2 Obstáculos dentro del terreno

En la delimitación del primer terreno de la Figura 21, los principales obstáculos incluyen las estructuras arquitectónicas, como postes específicamente en las calles de la universidad, que representan peligros de colisión debido a sus diferentes alturas. Además, si hay vehículos o personas circulando en la zona, esto añade complejidad al sistema autónomo, pues deben ser evitados en tiempo real esto teniendo en cuenta una altura mínima establecida para evadir los obstáculos antes mencionados. La vegetación presente, como árboles o arbustos como se indica en la Figura 23, también podría bloquear la visión o el vuelo del dron, generando puntos ciegos en la videovigilancia.



Figura 23. Objetos como árboles y dentro del campus farolas

En cuanto a las delimitaciones del segundo terreno los parqueaderos de la Universidad Técnica de Ambato de la Figura 22, los obstáculos más evidentes son los vehículos estacionados y en movimiento, que cambian constantemente y deben ser evitados por el dron. También, personas transitando en el área representan riesgos móviles. Otros obstáculos fijos incluyen farolas, postes de luz y señales de tráfico como en la Figura 24, que deben ser tomados en cuenta al planificar la ruta de vuelo. El sistema también se consideró la posible interferencia de señales electromagnéticas y condiciones climáticas que puedan afectar el control del dron.



Figura 24. Objetos como farolas y señales de tránsito.

Se deberá tener en cuenta cada altura de los diferentes objetos u obstáculo posibles antes mencionados para la planificación de misiones específicamente el dato necesario es la altura de estos ya que se pudo configurar el sistema de navegación del UAV a

ciertas alturas de vuelo, para aquello se detalla en la Tabla 15, las alturas de acuerdo con los estudios pertinentes.

Tabla 15. Alturas de los diferentes objetos u obstáculos [53] [54] [55] [56] [57]

Objeto u obstáculo	**Altura (m)**
Árbol de Pino	12 m (De acuerdo con la edad puesta en el estudio)
Palmera pequeña	8 m (Detallado como Washingtonia)
Palmera grande	12 m (Detallado como Washingtonia)
Postes de luz	8 – 15 m

3.1.3 Trayectorias del UAV dentro del terreno delimitado

Las trayectorias de los vehículos aéreos no tripulados en los sistemas autónomos de videovigilancia se diseñaron para cubrir eficazmente áreas limitadas y evitar obstáculos estacionarios y en movimiento. Normalmente, los drones pueden seguir un camino en forma de cuadrícula o de barrido, comenzando desde un punto de partida y cubriendo el terreno con rutas paralelas que cubren todo el perímetro. Este tipo de trayectoria proporciona una cobertura uniforme, lo que permite al dron capturar imágenes desde diversos ángulos.

Para evitar colisiones, el dron mantendrá una altitud segura y superará estructuras más altas, como edificios o postes de servicios públicos, mientras ajusta su velocidad o trayectoria si detecta vehículos o peatones. En áreas con estacionamientos o áreas urbanas más dinámicas, las trayectorias pueden seguir un patrón en zigzag, además de algunas rutas más adicionales como patrones rectangulares cortos o extensos, circular, rectangular, cuadrada o triangular, esto permite al dron evitar obstáculos como vehículos estacionados o en movimiento, se plantean diferentes posibles rutas que se programara para grabar en tiempo real en base al sector antes mencionado donde se realizara las pruebas como antes se menciona estas se podrán visualizar en la Figura 25.

De acuerdo con las operaciones aéreas para videovigilancia dispuestas para drones que se ha mencionado anteriormente de forma que es necesario un número de rutas mayor esto en cuanto a la planeación de misiones, así se tiene en cuenta también el tiempo de autonomía de vuelo del dron para volar y saber si se puede cumplir la misión con la ruta escogida.

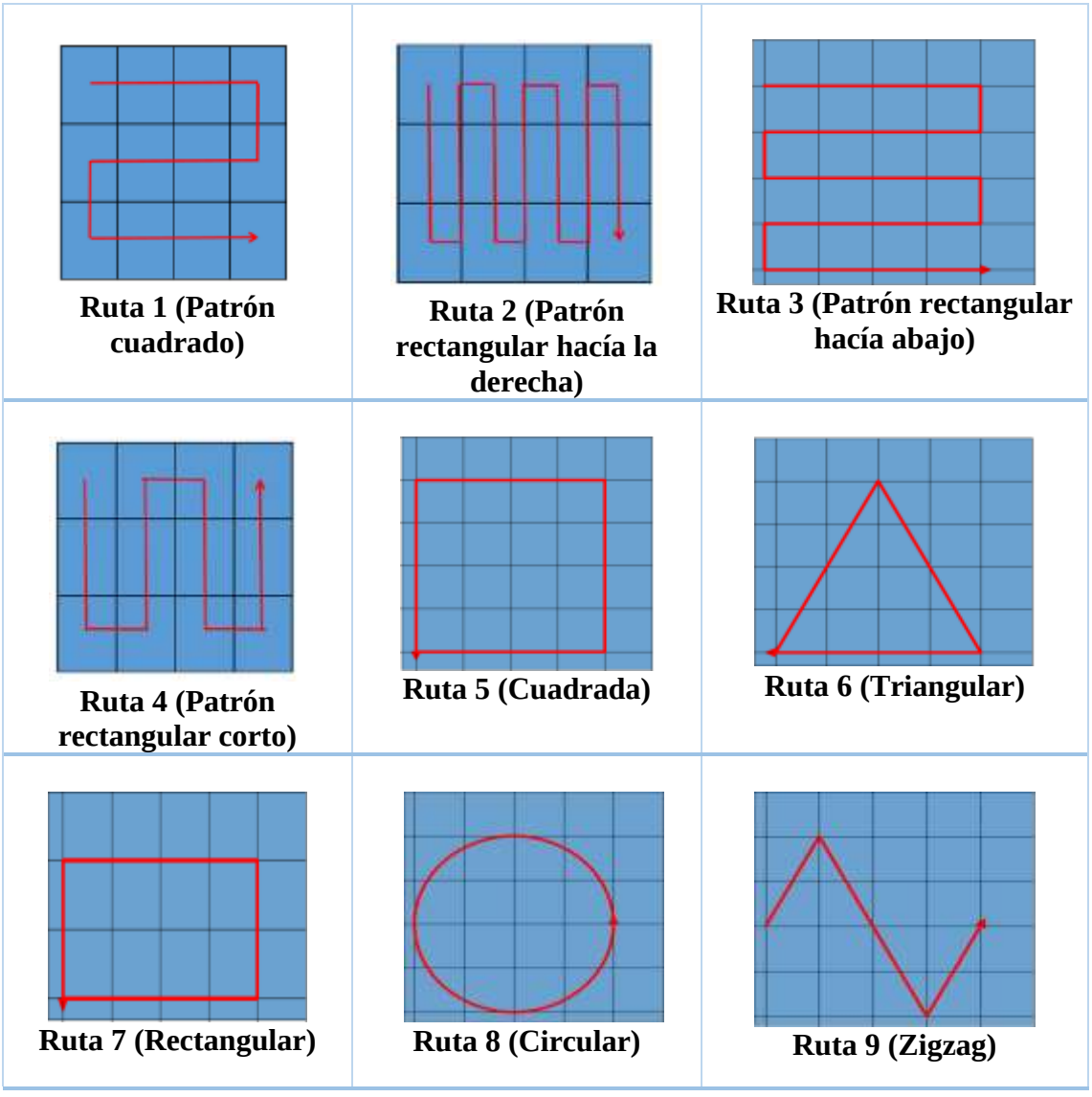


Figura 25. Tipos de ruta

3.1.4 Esquema general del sistema

En el diseño del sistema de navegación autónoma para un UAV va estructurado en tres etapas las cuales la etapa 1, se encarga de la recepción de información y estabilización

por parte del gimbal que está conectado hacia una cámara GoPro que se encarga de grabar audio y video para su posterior análisis en la estación base, también se encuentra la cámara FPV la cual visualiza en tiempo real video hacia la placa de video transmisión, en la misma etapa se encuentra una antena GPS NEO-M8N que recibe información satelital para cumplir las trayectorias, mientras tanto la etapa 2 realiza el análisis de la información mediante la placa controladora PIXHAWK 2.4.8 la cual se encarga del control de los motores del gimbal para la estabilización y movimiento de la cámara GoPro, esta también se encarga de procesar la información recibida a través de la antena GPS finalmente en esta etapa en el dispositivo aéreo se encuentra montado un video transmisor hacia el monitor FPV, todas estas acciones serán implementadas en el UAV y la última etapa será para visualización de datos y estación terrena de control y monitoreo implementados en un ordenador, como primera etapa la captura de datos por GPS, captura de video por FPV y por una cámara adicional con gimbal integrado para videovigilancia, esto basado en una comunicación serial UART/I2C además del uso de los pines auxiliares PWM en la tarjeta controladora de vuelo para el control del gimbal, como segunda etapa es el control de motores y comunicación con la estación base, en esta parte se tiene en cuenta los datos recolectados del GPS para la georreferencia específicamente el posicionamiento del dron, también la comunicación inalámbrica utilizando telemetría para la conexión con la estación base, entonces como última etapa será la estación base donde se visualizaran los datos y se configuró las rutas puestas para el dron mediante el software Mission Planner y el sistema de georreferencia así también para el control y monitoreo a partir de una interfaz gráfica en Python, la conexión será mediante comunicación inalámbrica utilizando telemetría antes mencionado, se visualizara el video en tiempo real en un monitor FPV. En la Figura 26 se presentará el esquema general del sistema dividido en etapas, destacando las conexiones y comunicación entre los diferentes elementos de cada bloque.

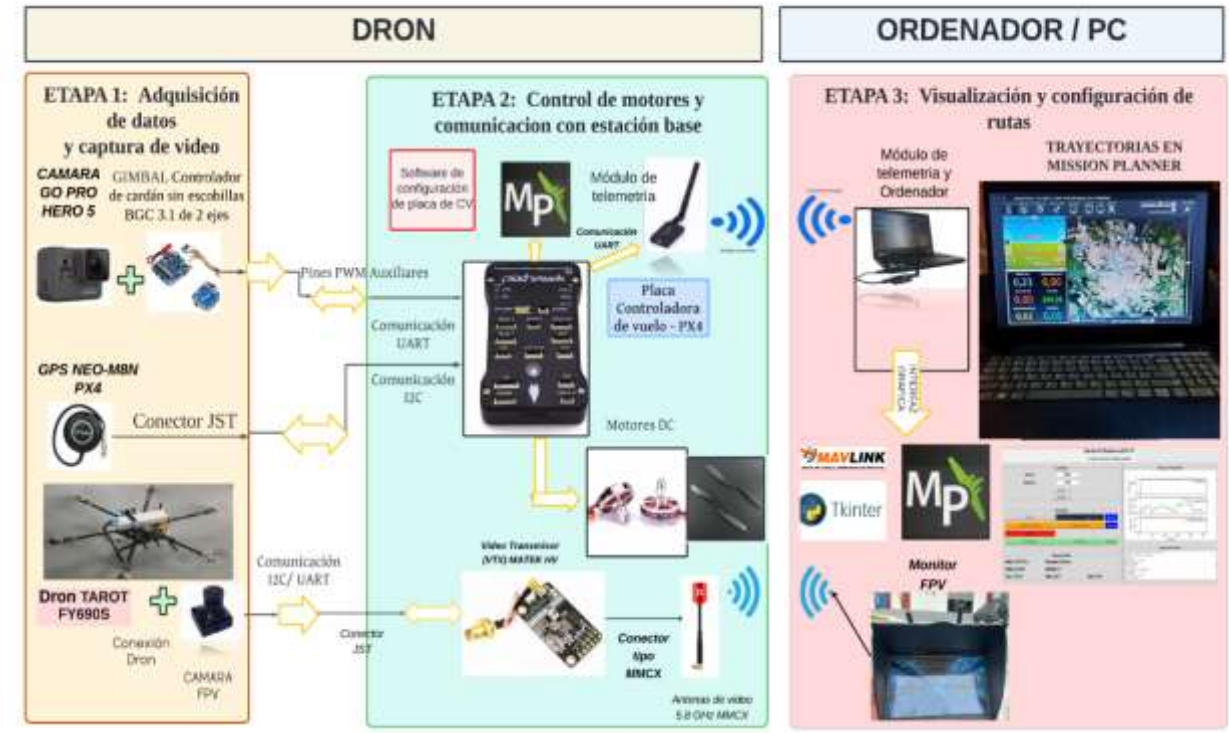


Figura 26. Arquitectura del Sistema

a. Etapa 1: Adquisición de datos y captura de video

La etapa de adquisición de datos y captura de video trata de la obtención de datos y captura de video en primera persona utilizando un sistema FPV, se utilizó un módulo GPS NEO-M8N PX4 utilizado como sistema de navegación por satélite (GNSS) esté conectado por el puerto GPS la comunicación serial se estableció mediante el protocolo UART y conexión por el protocolo I2C el cuál viene incorporada un compás esto conectado a la tarjeta controladora de vuelo PIXHAWK 2.4.8, en la funcionalidad del módulo GPS utilizado principalmente para el posicionamiento del UAV para cumplir trayectorias configuradas. La parte de captura de video será recolectada por una cámara FPV está conectada a un módulo Video Transmisor (VTX) MATEK HV teniendo comunicación serial UART a la cámara, esta viene acoplada a una antena de video 5.8 GHz MMCX para transmitir información por radiofrecuencia, la utilidad sirve para tomar en tiempo real el video y se pueda recolectar información para videovigilancia. Además de tener un gimbal control simple de dos ejes BGC de 32 bits integrado al UAV, controlado por el radio control conectados por las salidas auxiliares de la placa controladora de vuelo PIXHAWK 2.4.8, la comunicación fue por salidas PWM utilizando conexión serial para tener control de los servos motores a partir de

las perillas del radio control Radiomaster TX-16S, esto específicamente para tomas de video o fotografía utilizando una cámara GoPro Hero 5.

b. Etapa 2: Control de motores y comunicación con la estación base

La etapa de control de motores y comunicación con la estación base consiste en una tarjeta controladora de vuelo PIXHAWK 2.4.8, como medio principal para el control de los motores a partir del módulo GPS integrado, será la tarjeta controladora de vuelo la cuál tenga el control de velocidad de los mismos, la configuración es realizada en el software Mission Planner o la interfaz de control y monitoreo realizada en Python de acuerdo a configuraciones preestablecidas y cargadas en la placa se ha dado cierta velocidad a los motores para poder tomar direcciones entre derecha, izquierda, arriba, abajo, atrás o adelante. Para la conexión con la estación base se utilizó el puerto de telemetría 1 de la tarjeta controladora de vuelo PIXHAWK 2.4.8 la cual tiene una comunicación UART (serial), utilizando un módulo Holybro Sik de radio telemetría para la comunicación de forma inalámbrica hacía el ordenador o estación base que tendrá el mismo módulo pero con un puerto USB para la conexión al ordenador, será una comunicación full dúplex, estos módulos de telemetría utilizarán el protocolo de comunicación MAVLink, la parte de transmisión de video en tiempo real se encargará el módulo de Video Transmisor (VTX) MATEK HV, el cual estará conectado a la cámara FPV y se envió la información mediante radiofrecuencia utilizando el protocolo SmartAudio o IRC Tramp para transmisión y control de parámetros a través de radiofrecuencia (RF) en la banda de frecuencia de 5.8 GHz, se realiza la transmisión de video en tiempo real.

c. Etapa 3: Visualización y configuración de rutas

La etapa de visualización y configuración de rutas consiste en una estación base en donde se utilizó un ordenador para el uso y configuración del software Mission Planner, que sirve para la configuración de parámetros completos de la tarjeta controladora de vuelo PIXHAWK 2.4.8 además de tener un sistema de control y monitoreo en Python, la comunicación inalámbrica es mediante el módulo Holybro Sik de radio telemetría que se utiliza para la conexión con la tarjeta controladora de vuelo del UAV, que se requiere específicamente para configurar las rutas o trayectorias del dron mediante planeación de vuelo en la parte de la barra de herramientas que nos

concede el software, además se utilizó el sistema de georreferencia realizado en Python para recolección de datos precisos y poder cargar a la tarjeta controladora mediante el software Mission Planner. Así, al ajustar el modo de vuelo AUTO el UAV, tomará las rutas cargadas mediante el módulo de GPS tomando puntos georreferenciados para tener en consideración la ubicación de este. La visualización de video y en primera persona se realiza con un Monitor FPV, el cuál viene integrado con una pantalla LCD de 7 pulgadas, lo que proporciona una pantalla grande y clara para visualizar la transmisión de video en tiempo real, la comunicación se realiza mediante radiofrecuencia a la banda de frecuencia de 5.8 GHz en el canal 4 de los 8 posibles.

3.1.5 Selección de los elementos de hardware para el sistema de vuelo autónomo de un UAV

Se llevó a cabo un análisis de los distintos elementos de hardware que integran el sistema de vuelo autónomo de un UAV. Para lo cual, se elaboraron tablas comparativas que permitieron evaluar las características técnicas de cada elemento o dispositivo. Utilizando este método ayudo a la selección de los componentes que ofrecen los mejores parámetros técnicos para un rendimiento óptimo y eficiente en el sistema realizado.

a. UAV (Dron)

Para la selección del UAV dentro de los talleres tecnológicos de la FISEI, existe una cantidad variada de drones en las instalaciones, esto diferenciando la tecnología y usos los cuales han sido aplicados como para la agricultura de precisión, seguridad y desarrollo investigativo por parte de todo el personal educativo. Las configuraciones de cada uno los vehículos aéreos no tripulados (UAV’s) engloban muchas configuraciones y alcances de acuerdo con los módulos, sensores y las tarjetas controladoras de vuelo que estas disponen que es la parte primordial para la investigación. Entonces, en la Tabla 1 de algunos de los tipos de UAV se presentan las características técnicas de cada uno de los modelos de UAV disponibles, se escogerá el modelo Hexacóptero Tarot FY690S por sus características técnicas más sofisticadas que ayuda a la implementación de diferentes tipos de sensores y placas de control para adaptarlo al sistema autónomo de vuelo para videovigilancia.

b. Cámara

La cámara es un dispositivo o aparato que permite captar, tomar y reproducir imágenes. Esto para captar el video necesario para videovigilancia con ciertos parámetros como nitidez, brillo, contraste y resolución apta para fotogramas exactos tomados por el video en primera persona. En la Tabla 10 se muestra una comparación de tres distintas cámaras para captura de video con una resolución apta para visualizar lo que contenga el video en una calidad apta.

Para capturar el video en tiempo real se utilizará una cámara FPV apta para el módulo de Video Transmisor (VTX) MATEK HV, esto por el tipo de conector BJT para él envió de datos, se utilizará la comunicación serial utilizando el protocolo UART para transmisión de video y audio, además el montaje de todo el sistema FPV es de un tamaño reducido el cuál es óptimo para no aumentar peso en el UAV previamente escogido. Se escoge está cámara por ser útil para utilizar radiofrecuencia para el envío de video y la visualización en un monitor FPV específicamente para todo este sistema.

c. Placa controladora de vuelo

La placa controladora de vuelo es la placa central de control, configuración, procesamiento y conexión de algunos de los sistemas de drones para UAV´s los cuales son necesarios para incorporar sensores, motores, GPS, barómetro, pantalla OLED para visualización de parámetros, módulo de telemetría, y otras incorporaciones de control y visualización de datos. Para ello, se muestra la Tabla 4 una comparativa de entre características técnicas importantes para la configuración ya sea por software o por programación.

De acuerdo al análisis de los diferentes parámetros técnicos de las placas controladoras para el UAV, se escogerá la placa PIXHAWK 2.4.8 sus características son adecuadas en cuanto a la estructura del dron ReadyTosky ZD550, esto por el tamaño o dimensiones 82,2 x 51,8 x 16,5mm adecuadas en cuanto al peso 38g siendo adaptable para la estructura completa del dron, pero la mayor utilidad en cuanto a diferentes sensores, módulos y conectividad por diferentes formas, el uso preferencial en cuanto de datos de georreferencias dando la posibilidad de ubicar un GPS con una precisión alta para captar datos completos para la configuración de trayectorias. El placa

controladora de vuelo tiene la posibilidad de conexión inalámbrica usando telemetría esto para la conexión terrena que esencial para el sistema a incorporar ya que se necesita el monitoreo en una estación terrena, el procesador es de suma importancia en cuanto a la respuesta para el control de motores que ayudarán a cambios bruscos de la trayectoria configurada anticipadamente, además de la conexión y configuración de la placa puede ser por comunicación serial por USB mini, el números de salidas auxiliares utilizando señales PWM es adecuada para la conexión de la configuración y control de los servo motores del gimbal, teniendo una parte del pitch y otra del roll, también poder conectar un número considerable de sensores en cuanto al puerto serial UART y I2C, esto ayuda al control del sistema para el vuelo autónomo del UAV.

d. Gimbals

El gimbal es un dispositivo estabilizador integrado con servo motores controlados por una placa de control, conectada con sensores, acelerómetros y compás magnético, el uso específicamente por la estabilización mecánica ante perturbaciones del movimiento del UAV. Para ello, se muestra la Tabla 3 una comparativa de entre diferentes gimbals el cuál posee cardanes y un número de ejes mayor para una estabilidad óptima.

Una vez comparado los diferentes parámetros de las placas de diferentes gimbal, de acuerdo con el peso de la cámara escogida tanto de la cámara GoPro Hero 5 y la cámara FPV, tienen un peso aproximado de 145,34 gr de forma que el peso no afecta considerablemente al UAV y es adecuado para la estabilidad del UAV, se escoge Controlador de cardán BGC 3.1 de 2 ejes por el número de servo motores el cuál viene implementado en la estructura del gimbal siendo eficiente para grabar el video.

e. Módulo de telemetría

La telemetría es un sistema de comunicación inalámbrica trabaja a una frecuencia de 915 MHz trabaja con el protocolo de comunicación MAVLink (UART) específicamente para drones, a una distancia considerable que ayuda a recoger, procesar y transmitir información de un dispositivo electrónico que en este caso será los sensores o módulos incorporados en el UAV, además que él envió de datos para monitoreo en la estación terrena que será el ordenador. Se recolecta información la

cual se muestra en la Tabla 12, se realizó una tabla comparativa de las características técnicas de cada uno de los diferentes módulos de telemetría.

De esta manera se compara los parámetros técnicos de los módulos de telemetría, las características esenciales son la frecuencia de operación que se configura, el protocolo de comunicación que sea compatible con para la configuración de la tarjeta controladora de vuelo PIXHAWK 2.4.8, una de las características más esenciales será el rango de alcance del módulo de recepción en la estación terrena y el módulo de transmisión en el UAV, para ello se escoge el módulo HolyBro SiK de radio telemetría que se adecua de forma correcta a las características necesarias del sistema.

f. Módulos de Video Transmisión

La transmisión de video es él envió en tiempo real de un flujo de datos de video o audio para ser transmitida por una canal de comunicación a una frecuencia de operación, utilizando radiofrecuencia como tecnología de comunicación. Los módulos en circuitos electrónicos son un sistema flexible y modular, que viene integrado con electrónica específica para aplicaciones en concreto. De forma que, es necesario un sistema FPV para captar el video en tiempo real y poder transmitirlo a un monitor FPV. Para ellos, en la Tabla 2 se realizó una comparativa entre características técnicas para la elección de cierto módulo de transmisión de video para el sistema de videovigilancia en tiempo real.

Una vez comparada las diferentes características de los módulos se escoge el Módulo Video Transmisor (VTX) MATEK HV 5.8GHz; debido a la frecuencia de operación ajustable de entre 5,8 GHz los valores son ajustables debido a un rango de frecuencias, el protocolo de comunicación que utiliza es el analog video transmisión (AV), y el formato de video NTSC y PAL al cuál envía el flujo de datos de forma que no tenga perdida de señal al momento de la transmisión de video para el sistema de videovigilancia, además de tener 40 canales siendo eficiente en la prueba de cada uno de ellos en el caso de fallo de alguno cabe recalcar que el nivel de potencia de transmisión es eficiente para no tener ruido en la comunicación ya que se tomara en cuenta una distancia de entre 100 a 1000 m entre el dron y el monitor FPV.

g. GPS

El sistema de posicionamiento global (GPS) es un sistema de navegación que usa satélites, un receptor y algoritmos para sincronizar datos de localización, la selección de un módulo GPS es necesario para la configuración de rutas mediante georreferencia configurado mediante el software Mission Planner y la interfaz de control y monitoreo en python, para aquello se realizó la Tabla 5 para tener una comparativa entre características técnicas deseadas a módulos de GPS en concreto.

Una vez revisado alguno de los módulos GPS, se tiene en cuenta como principales características para el diseño de rutas como la frecuencia de operación de 1575.42 MHz y la tecnología soportada GNSS siendo el detalle para la configuración y captación de un número determinado de satélites de entre 7 a 14 para la georreferencia del UAV, además de la presión bordeando un valor de 1 a 2,5 m del mismo teniendo en cuenta las condiciones con las que trabaja el módulo, para funcionamiento correcto una vez ya cargadas las trayectorias deseadas mediante el software Mission Planner o por la interfaz de control y monitoreo en Python.

h. Fuente de Alimentación del UAV

Para la alimentación completa del sistema y del UAV, se utilizarán dos baterías Lipo de acuerdo a los requerimientos de voltaje y corriente de los motores los cuales se concentran un mayor nivel voltaje se utiliza una batería Lipo de 6S con un voltaje de 22.2 V y de corriente 6000 mAh, entonces en base a los elementos completos del UAV se separará otra batería Lipo de menor voltaje y corriente para la alimentación de los servos motores del gimbal los cual se utiliza una batería de 3S con un voltaje de 11.1 V y de corriente 1500 mAh; siendo eficiente y óptimo para realizar vuelos y planificación de rutas de forma óptima, se presenta en la Tabla 16 el tipo de baterías encontradas y utilizadas en el UAV.

Tabla 16. Baterias Lipo [58] [59]

Baterias de Alimentación	Caracteristicas
	Batería de Polímero de Litio (LiPo) Material: Li-polímero; Celdas: 6 S; Voltaje: 22.2 V; Capacidad: 6000 mAh; Descarga: 50C; Enchufe de descarga: EC5 Plug
	Batería Lipo Turnigy Capacidad: 1500 mAh. Configuración: 3S1P / 11.1 V / 3Cell. Descarga constante: 20C. Pico de descarga (10seg): 30C. Tamaño del paquete: 80 x 34 x 28 mm. Conector de carga: JST-XH. Conector de descarga: XT60.

3.1.6 Software para el sistema de vuelo autónomo de un UAV

El software elegido de acuerdo con la configuración del sistema completo de navegación autónoma de un UAV para videovigilancia. Se desarrollo los diagramas electrónicos de los módulos completos del sistema de control de motores, sensorización y del sistema FPV para videovigilancia, programación y configuración de dispositivos de comunicación, configuración y planeamiento de rutas por software, además de programar la parte de georreferencia y simulación de trayectorias por programación en dos lenguajes distintos.

a. Ardupilot / Mission Planner

El software Ardupilot / Mission Planner es la base para la configuración de la tarjeta controladora de vuelo PIXHAWK 2.4.8 que ha permitido la configuración de sensores, GPS, gimbal y control de motores para el vuelo, este software a su vez ha permitido la planeación de trayectorias o rutas de forma que el UAV cumpla las trayectorias cargadas una vez configurado, además de ellos sirve como estación terrena de monitoreo completo del UAV.

b. Python – Visual Studio Code

El software de edición de código se ha utilizado para programar el lenguaje de programación en Python es utilizado para la simulación con todos los parámetros necesarios del UAV utilizando librerías específicamente para el modelado y diseño de trayectorias esto además de parámetros técnicos y graficas para el análisis del comportamiento del UAV, además de utilizar scripts para georreferencia en la que se usó la tecnología GPS para el vuelo autónomo.

c. Fusion 360

El software Fusion 360 se ha utilizado para el diseño de la estructura del UAV, para cálculos de dimensiones respecto al modelo completo del modelado en 3D que se utiliza en la simulación y análisis del comportamiento del UAV, además de la fabricación de piezas con modelos y ensamblajes complejos para estructuras o partes de este.

d. Matlab

El software Matlab se conoce por su nivel de cálculos en cuanto a ser un laboratorio matricial de forma que se puede simular las trayectorias y rutas de acuerdo con modelos matemáticos para tener un ambiente simulado de las posibles rutas y trayectorias, la programación es basta en cuanto cantidad de librerías para el uso de drones.

3.1.7 Diseño de la estructura del UAV HEXACOPTER TAROT 680

La estructura del dron se la realizo con el propósito de conocer el objeto aéreo y con esto entender de mejor manera todo lo relacionado al añadir el sistema FPV y sistema de control del gimbal para la video vigilancia del espacio y operaciones en el espacio aéreo.

Mediante el software AUTODESK FUSION 360 se desarrolló el diseño 3D para cada elemento que conforma dicho UAV, en este caso el marco los brazos, hélices, motores, etc. Para que una vez acoplados fueran unidos y formen el ensamblaje del dron.

a. Marco principal del dron

En la Figura 27, se presenta una estructura tridimensional del sostén principal del dron donde consta de unas dimensiones que en dicha base están diseñados los espacios correspondientes para cada componente a añadir. Cada una de estas características permiten una mejor visualización del espacio comprendido para el ensamblaje de los componentes.

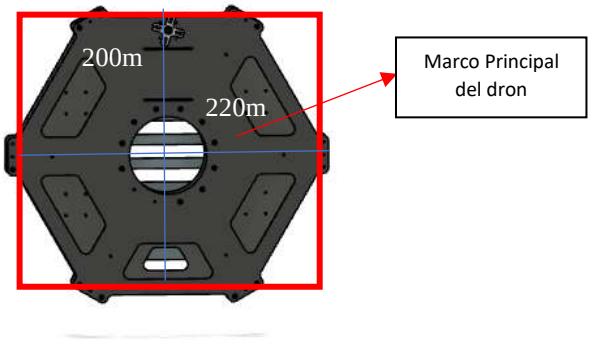


Figura 27. Marco Principal del dispositivo

b. Brazos exteriores

Los brazos exteriores hechos en fibra de carbono tienes una distancia de 330 mm, cada uno de estos elementos son esenciales en el UAV debido en ellos se ensamblan los motores, los cuales integrados con las hélices se encargan del vuelo del dispositivo como se puede observar en la Figura 28.

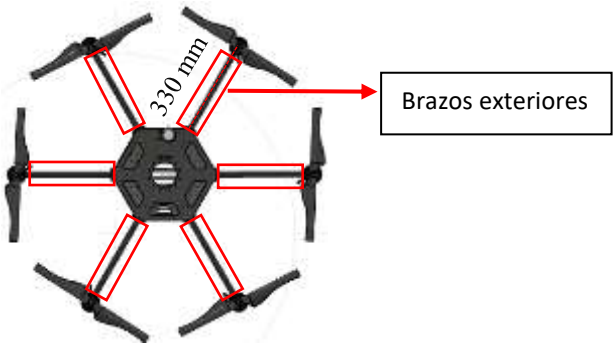


Figura 28. Diseño brazos exteriores.

c. Motores

Motores sin escobillas que generan una potencia de 750 W y con un tamaño de 50 mm de diámetro los cuales están encargados que generar el movimiento de las hélices del UAV justo aquí es donde se genera la fuerza de empuje para el UAV, como se puede visualizar en la Figura 29, junto con las hélices permiten que la masa de la unidad de vuelo se desplace y realice las trayectorias previamente analizadas antes del vuelo.

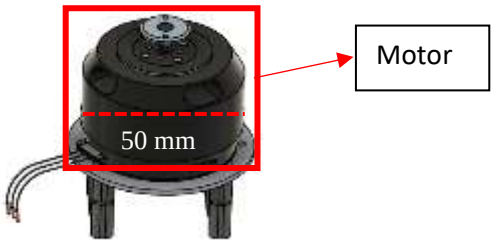


Figura 29. Diseño de motores.

d. Hélices

Estos elementos son encargadas de darle fuerza de empuje, velocidad, propulsión y estabilidad al dispositivo, las cuales forman la parte fundamental para realizar vuelos, que se visualiza en la Figura 30, se puede observar las medidas e inclinación del diseño en este caso son: 12 pulgadas para cada brazo con un grado de inclinación de 45°, de las cuales en el centro aparece un orificio de diámetro de 6.5mm, mismo que permite fijar cada hélice al motor y poder tener su fuerza de empuje generados.



Figura 30. Diseño de Hélices.

e. Antena GPS

Esta antena GPS NEO-M8N utiliza tecnología GNSS (Global Navigation Satellite System) para recibir señales de sistemas como GPS, GLONASS y Galileo, es conocido por su precisión de hasta 2.5 m con una eficiencia en consumo de energía, haciéndolo ideal para proyectos de navegación, geolocalización y drones.

La antena GPS que se visualiza en la Figura 31, se encarga de recibir las señales satelitales las cuales cumplen una función esencial para la navegación, posicionamiento y control de vuelo.

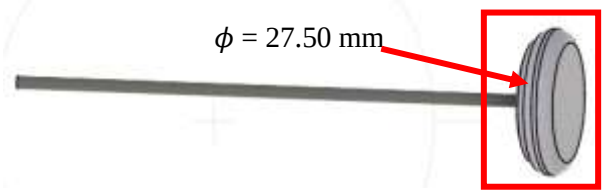


Figura 31. Diseño de antena Satelital.

f. Antena RX de radiofrecuencia

La antena RX que se visualiza en la Figura 32, trabaja en una banda de 2.4 GHz que es la más utilizada para sistemas de radio control por su estabilidad, alcance confiable y resistencia a interferencias la cual se encarga de receptar cada una de las señales que son transmitidas por el Radio control Radiomaster TX16S.

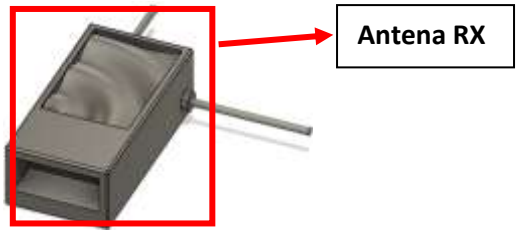


Figura 32. Diseño de Antena Rx.

g. Diseño renderizado de todos los componentes del dron

Fusion 360 es una plataforma CAD/CAM que permite el diseño, simulación y fabricación de componentes con un alto nivel de precisión. En este proyecto, se empleó para desarrollar el diseño del Hexacoptero Tarot 680, lo que permitió generar un modelo detallado y funcional, optimizando la evaluación de su estructura y rendimiento. Como se representa en la Figura 33, la renderización del armazón del dron junto con todos sus elementos físicos, este diseño se realizó acorde al modelo del UAV, todo esto para poder conocer sus componentes ya sean mecánicos y electrónicos.



Figura 33. Diseño renderizado del UAV.

3.1.8 Implementación electrónica del sistema de navegación autónoma del UAV



Figura 34. Conexión Electrónica del Sistema completo del UAV

La conexión electrónica del sistema representa la configuración de un sistema avanzado de control diseñado para un hexacóptero, como se puede visualizar en la Figura 34, cuya estructura está basada en el controlador de vuelo Pixhawk. Este sistema integra seis motores brushless dispuestos en un patrón hexagonal, proporcionando mayor estabilidad y capacidad de carga. Adicionalmente, incluye

elementos de estabilización como un módulo GPS y un gimbal con cámara, una pantalla OLED para visualización de datos y un módulo de seguridad. El diseño está orientado tanto para la operación autónoma como manual, utilizando un radiocontrol Radio Master TX16S. Además, cuenta con una interfaz en el software Mission Planner así también como una interfaz en Python, que facilita la programación de trayectorias y el monitoreo en tiempo real de los parámetros operativos del dron.

a. Análisis detallado por componentes y conexiones.

El sistema del dron integra múltiples componentes para garantizar un vuelo estable, seguro y autónomo, destacando la placa controladora de vuelo Pixhawk, que actúa como núcleo central, gestionando las conexiones con los motores, sensores, GPS y telemetría, permitiendo tanto navegación manual como autónoma. La configuración del hexacoptero incluye seis motores sin escobillas distribuidos en un patrón alterno (tres motores giran en sentido horario y tres en antihorario), controlados por un ESC (Controlador de velocidad electrónico) individuales que regulan la potencia de la batería principal de litio-polímero (LiPo), encargada de suministrar energía a todo el sistema. Además, incluye un gimbal de 2 ejes con un cardan que es la estructura que sostiene a la placa de control que estabiliza la cámara durante el vuelo, mientras el módulo GPS proporciona posicionamiento preciso en tiempo real. Una pantalla OLED conectada al Pixhawk muestra la información clave como altitud, estado del dron (Armado o desarmado) y el estado de la batería, y un botón de seguridad asegura el sistema antes del despegue. El control remoto Radiomaster TX16S permite alternar entre modos manual (Stabilize) y autónomo (Auto), mientras que el módulo de telemetría envía datos a la estación terrena que tendrá el software Mission Planner en un ordenador además de la interfaz de control y monitoreo realizado en python, facilitando la programación de rutas y el monitoreo completo del dron durante la operación, toda su estructura se puede observar en la Figura 34.

b. Análisis y conexiones de Video vigilancia con Gimbal

Este sistema utiliza motores sin escobillas (brushless) y sensores inerciales que, combinados con un controlador que mantienen la cámara nivelada mediante una placa de control. Está conectado al Pixhawk para sincronizarse con el sistema de vuelo, pero

también permite ajustes manuales desde el control remoto Radiomaster TX16S, con el que el piloto puede manipular la inclinación (pitch) y la panorámica (yaw) de la cámara para tareas específicas. En cuanto a su alimentación, el gimbal utiliza una batería independiente LiPo 3S de 1.3 amperios, lo que asegura un suministro estable de energía y permite operar de manera autónoma sin afectar el rendimiento energético del sistema principal del dron.

c. Funcionamiento del UAV

Una vez que se ha montado la parte del hardware del dron verificando esté completo, se debió conectar la batería principal de LiPo al conector de la placa de distribución de energía (PDB) vinculada al Pixhawk, que será la fuente de alimentación para todo el UAV. Fue esencial la verificación de que los cables estén conectados correctamente, respetando la polaridad de los terminales para evitar cortocircuitos. Además, se debió conectar la batería secundaria que alimentará el gimbal, generalmente una batería LiPo de 3 celdas con una capacidad de 1.5 Ah.

d. Activar el Botón de Seguridad

Se debe activar el botón de seguridad del Pixhawk, este botón es una medida de seguridad que impide que el dron despegue de manera accidental. Para activarlo, se debió presionar el botón de seguridad en la controladora de vuelo antes de encender el sistema. Como método de seguridad para realizar las configuraciones o calibración del mismo antes del vuelo.

e. Calibración del Dron

Antes de realizar el vuelo, se requirio calibrar los sensores de la controladora de vuelo Pixhawk. Esto se logra conectando el Pixhawk a la computadora utilizando el software Mission Planner o QGroundControl. Una vez conectado, se procedió a calibrar los sensores esenciales: acelerómetro, giroscopio y compás. El proceso de calibración debió llevarse a cabo en una superficie plana para el acelerómetro y el giroscopio. Se calibro el compás, moviendo el dron en todas las direcciones mientras el software se tuvo como guía al usuario.

f. Conexión con Mission Planner

Luego de completar la calibración, fue necesario establecer la conexión con Mission Planner a través de la telemetría o por medio de un cable USB. En el software, se debió verificar que la placa principal esté correctamente conectada y que los sensores estén detectados adecuadamente. Se debió revisar que los datos en tiempo real (como altitud, ubicación GPS, número de satélites receptados y estado de la batería) se visualicen de manera precisa en la interfaz. Para pasar a la planeación de rutas y establecer la misión al dron.

g. Cargar la Trayectoria de Vuelo

En Mission Planner, se debió definir los puntos de referencia o waypoints que el dron requirió seguir. Estos waypoints o ciertos comandos establecidos en la configuración se pudo cargar manualmente o a través de un archivo preconfigurado. Fue esencial asegurarse de que los puntos estén dentro del rango de vuelo del dron. Una vez definidos los waypoints, se debió guardar las configuraciones y a ver verificado que el dron haya recibido correctamente la información de la trayectoria.

h. Configuración del Gimbal

Para configurar el gimbal, se debió conectar este componente a la placa controladora de vuelo Pixhawk, asegurándose de que esté alimentado por la batería LiPo de 3 celdas. Desde el software Simple BGC GUI v2.2 fue utilizado para la configuración del gimbal, donde se debió calibrar su movimiento, además de a ver configurado la placa Pixhawk en el software Mission Planner para poder controlarlo mediante el radio control.

i. Puesta a Vuelo del Dron

Con todos los sistemas correctamente configurados, se pudo poner a prueba el vuelo. El dron se encendió y se verifico que todos los componentes, como los motores, el gimbal, y los sensores, estén funcionando correctamente. Se inició Mission Planner ó la Interfaz de control de monitoreo y control en Python para seleccionar el modo de vuelo adecuado. Como primera prueba de verificación correcta del sistema se utilizó los controles del radio control, se comenzó en modo manual antes de a ver pasado al modo autónomo (Auto), donde el dron siguió los waypoints previamente cargados o

el comando establecido a realizar de acuerdo con la planeación de rutas. Es importante el reconocimiento del terreno, asegurándose de a ver monitoreado constantemente el estado del dron a través de la telemetría y manteniendo el control visual del mismo. Si se puso todo en un estado de correcta funcionalidad, el dron podrá tener un vuelo preciso.

3.1.9 Configuración completa del sistema de navegación autónoma del UAV

Para la configuración y planificación de misiones del UAV antes mencionado se utiliza el software Mission Planner, se necesitará algunos pasos para empezar con las pruebas de vuelo desde la calibración de los sensores como barómetro, compas y acelerómetro, que son partes fundamentales para realizar pruebas de vuelo del UAV, de forma que para aquello se especifica algunas de las partes y funciones del software antes mencionado para tener consideraciones de los datos mostrados en el mismo estas funciones se mencionan a continuación:

- En el apartado de “PLAN” para agregar puntos del plan de vuelo se tendrá Point-and-click para la entrada waypoint, esto utiliza Google Satelital Maps o en caso de requerir otro mapa que se necesario visualizar en el software.

- En el apartado de “DATA” de vuelo se tiene la Estación terrena completa para el monitoreo del UAV y de las misiones y el envío de comandos durante el vuelo.

- Dentro de “DATA” se podrá verificar parámetros y datos a la salida del sensor y rendimiento del piloto de pruebas.

- Dentro de “PLAN” para descargar archivos de registro de la misión configurada o planteada y poder analizarlos.

- Configurar las opciones de la tarjeta controladora de vuelo de la PIXHAWK 2.4.8 para su fuselaje.

- Visualización de la salida de la terminal de serie de la PIXHAWK 2.4.8.

Algunas de las partes de monitoreo en la estación terrena se especifican en la Figura 35 las partes del Heads-up display (HUD) que ayudo a monitorear el UAV en el momento de las pruebas del vuelo ya sea a través de la conexión USB o por telemetría.

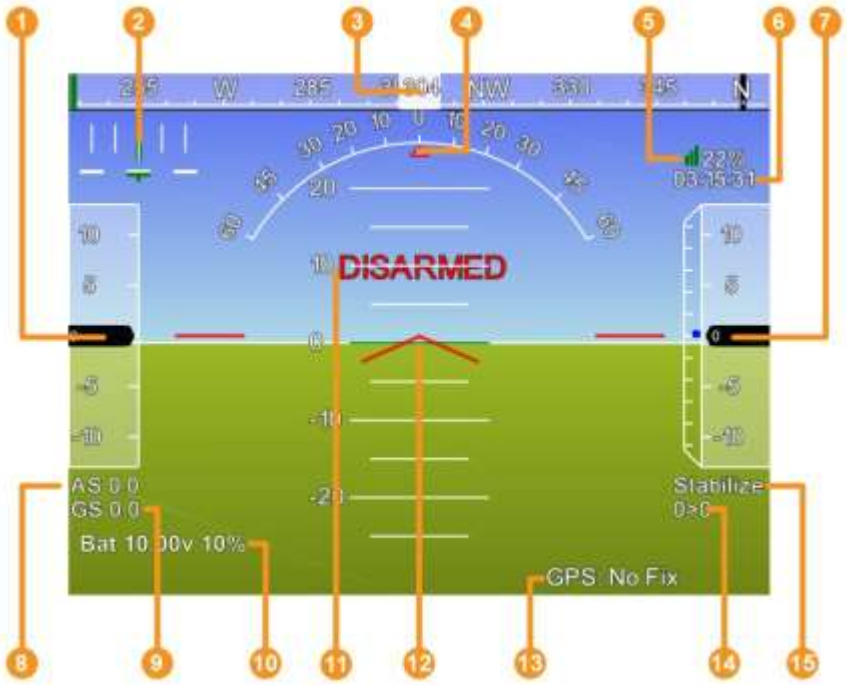


Figura 35. Partes del Heads-up display (HUD)

1. Velocidad del aire (velocidad de avance si no hay sensor de velocidad está equipado)

2. Error de Rumbo y velocidad de giro (T)

3. Dirección Heading

4. Ángulo De Inclinación

5. Conexión de telemetría inalámbrica (% paquetes erróneos)

6. Tiempo GPS

7. Altitud (barra azul es velocidad de subida)

8. Velocidad del aire

9. La velocidad de avance

10. Estado de la batería

11. Artificial Horizon

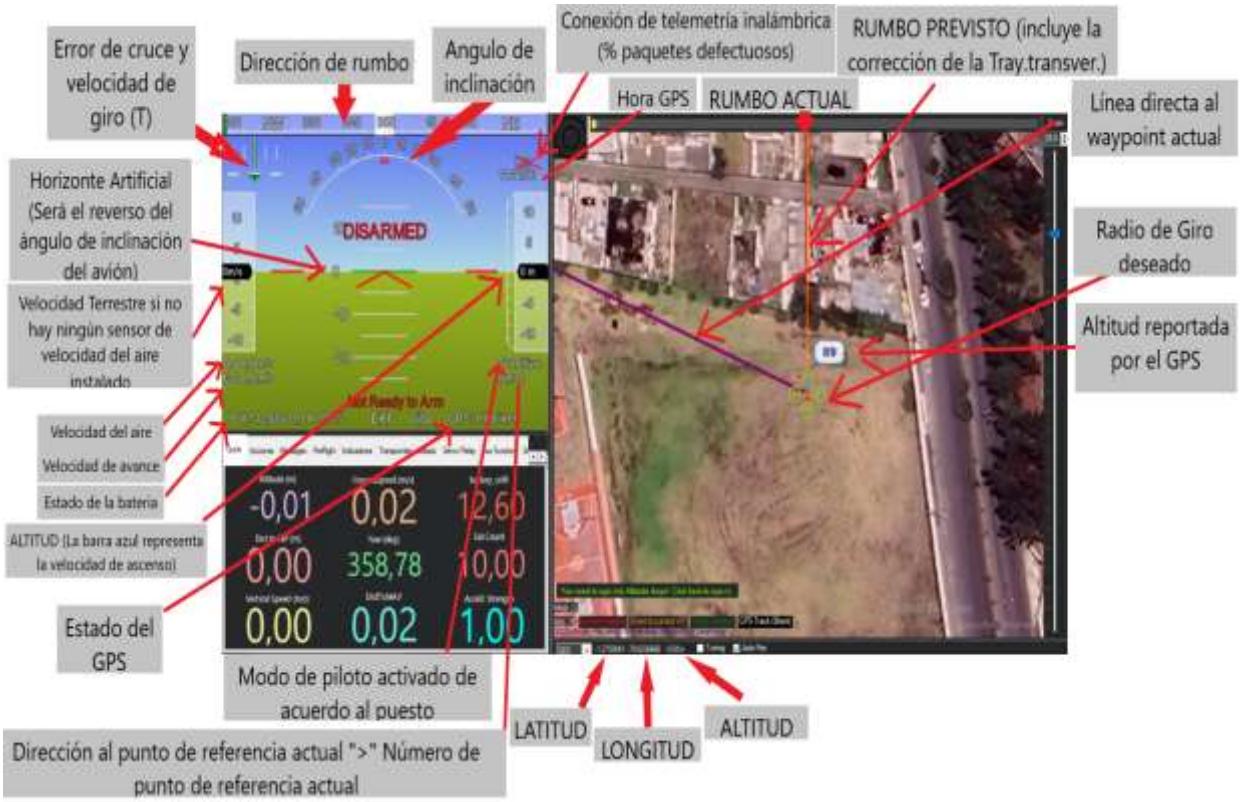


Figura 36. Partes detalladas de Flight Data del software Mission Planner

En la Figura 36, se visualiza la estación terrena de la planificación de Misiones en el software Mission Planner, como primero se mostrará en la parte superior izquierda el heads-up display (HUD) antes explicado, para después tener en la parte derecha el mapa satelital proporcionado por Google Satelital Maps y en la parte inferior izquierda se tendrá los valores recogidos de los diferentes sensores del UAV, además de otras posibles funciones dentro del software, se menciona además algunas de características de las herramientas antes mencionadas del software.

a. Configuración del módulo de telemetría con la Pixhawk 2.4.8

Se inicio por montar el módulo de telemetría específicamente para evitar la conexión con el puerto micro USB que tiene la placa de control PIXHAWK 2.4.8, el principio de funcionamiento del módulo de telemetría viene explicado en el diagrama de flujo que se visualiza en la Figura 37.

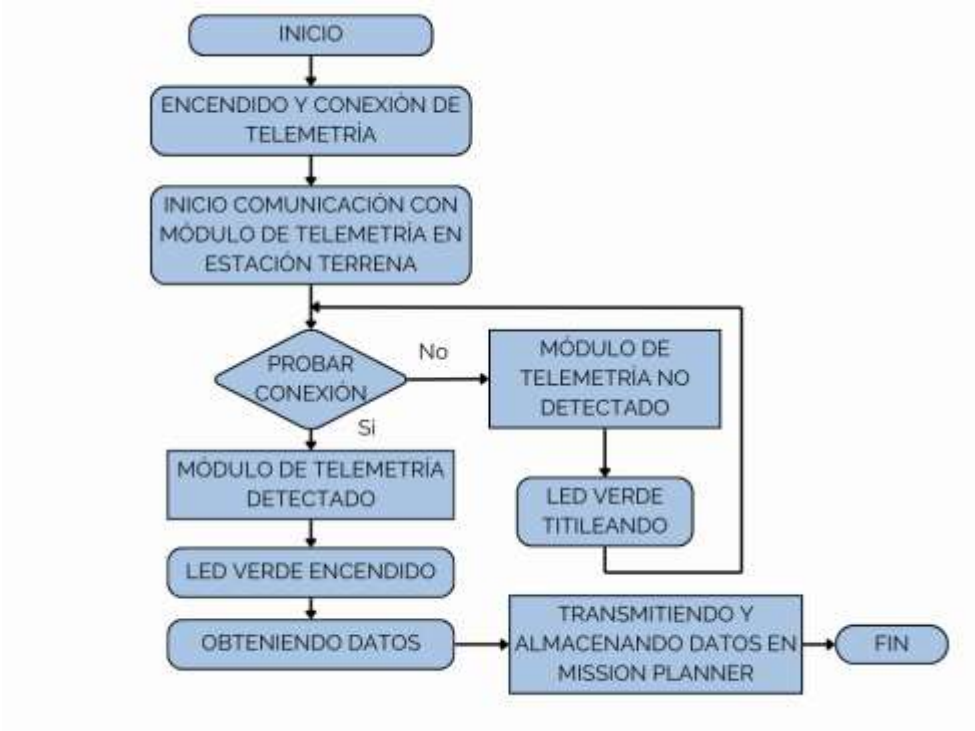


Figura 37. Diagrama de flujo de funcionamiento de Telemetría

Configuración del hardware y montaje del sistema de telemetría

Se monto el módulo de telemetría Holybro Ski que tiene como base una conexión por un puerto BJT de 4 pines establecidos para la comunicación UART del puerto de telemetría (TELEM1) el cual tiene una entrada igualmente de puerto BJT hembra de 6 pines, y en la estación terrena el mismo módulo Holybro Ski pero con una conexión micro USB a el puerto USB del ordenador esto como se visualiza en el esquema electrónico de la Figura 38.

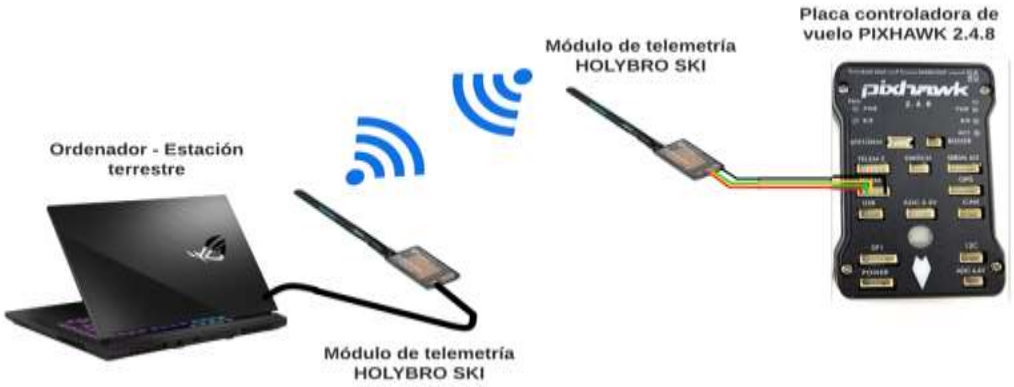


Figura 38. Esquema electrónico del sistema de telemetría del dron con la estación terrena

También se podrá visualizar en la Figura 39 como estuvo montado el módulo de telemetría tanto en la estación terrena como en el UAV.



Figura 39. Módulo de telemetría montado en el UAV y en la estación terrena

Configuración del software del sistema de telemetría con la placa de vuelo Pixhwak en el software en Mission Planner

Para la configuración de la placa Pixhawk 2.4.8 en el caso de no tener configurado o reseteado la placa desde cero, se necesitó cargar algunos parámetros para habilitar la comunicación UART además de los baudios necesarios para la comunicación correcta de datos, estos parámetros se muestran en la Tabla 17.

Tabla 17. Parámetros ajustar para activar el puerto de Telemetría

Parámetro	Valor
SERIAL1_PROTOCOL	2 (Protocolo MAVLink)
SERIAL1_BAUD	57 (57600 Baudios)

Se cargo los parámetros dirigiéndose al apartado de Config > Full parameters List para editar los parámetros antes mencionado estos pasos se visualizan en la Figura 40.

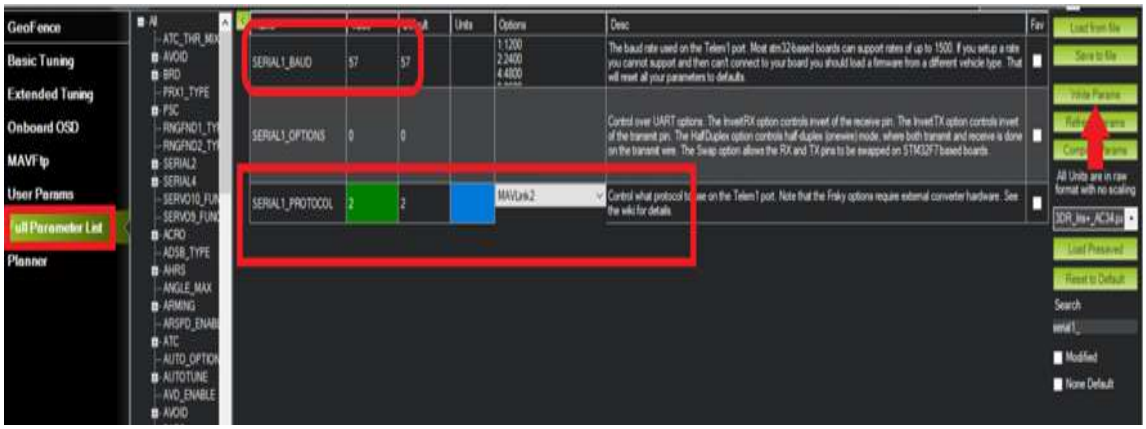


Figura 40. Configuración de parámetros para el módulo de telemetría dentro del software Mission Planner

b. Calibración completa del UAV con el software Mission Planner para la puesta en marcha

Para la calibración se empezó con la conexión de dos formas ya sea por telemetría o por conexión serial, en el sistema se incorporó un módulo de telemetría mencionado en la selección del hardware para el sistema, es de parte fundamental para la velocidad de transmisión de datos la selección correcta de los baudios, se debió escoger 57600 Baudios para el módulo de telemetría como se visualiza en las Figura 41, y se escogerá 115200 Baudios para la conexión por micro USB por conexión serial.



Figura 41. Desconexión y conexión mediante el software Mission Planner

Calibración del acelerómetro del UAV

La configuración del acelerómetro en el UAV una vez se realizó la conexión del sistema completo se debió calibrar la parte del acelerómetro el cuál utiliza un IMU (Unidad de medidas inerciales) este se define como un dispositivo capaz de medir y reportar la actitud, la velocidad, los cambios de altitud y las fuerzas gravitatorias que actúan sobre el UAV. Este a su vez sumara las fuerzas de inercia dependiendo del número de motores que incluya el UAV. En un sistema de navegación inercial, los datos recopilados e informados por una IMU se procesan a través de la estación terrena que será un ordenador para calcular la posición actual en función de la velocidad y el tiempo [60].

Para iniciar con la calibración de la conexión a la placa y la estación terrena debió realizarse mediante telemetría o comunicación serial mediante USB para establecer conexión con el software Mission Planner se necesitó entrar al apartado de Setup o configuraciones, para después redirigirse a Mandatory Hardware > Accel Calibration; luego en el apartado de modo de vuelo 4 dependiendo en el modo de vuelo que este el dron para calibrar es necesario hacerlo en modo estabilizado (Stabilize), y clickear en "Calibrate Accel" como se visualiza en la Figura 42.



Figura 42. Calibración Acelerómetro Inicio

Una vez presionado el “calibrate Accel”; se debió calibrar de acuerdo con la posición del dron teniendo en cuenta que la placa tendrá una señal en la carcasa la cual se debe apuntar en dirección de la cámara de FPV, se empezó calibrando con diferentes posiciones, la posición inicial es la de LEVEL (Nivelado), para pasar a la posición de LEFT (Izquierda), luego la posición de RIGTH (Derecha), luego la posición NOSEDOWN (De frente), para después la posición NOSEUP (Atrás) y acabar con la posición BACK (Boca Abajo), cada una de las posiciones de este método se visualizan en la Figura 43.



Figura 43. Posiciones de calibración del dron

Al finalizar la calibración del Acelerómetro, se tuvo un pitido específico del buzzer de la placa controladora de vuelo Pixhawk y se pudo visualizar un parpadeo verde del LED indicador de la placa, además se mostró un mensaje de correcta calibración en el

software de forma que sirvió para comprobar si se calibro de manera eficiente como se indica en la Figura 44.



Figura 44. Mensaje en el software finalización correcta del Acelerómetro

Calibración del compás del UAV

El compás en el dron será una parte fundamental para la prueba de vuelo, considerando que se define como un dispositivo electrónico que permite al cerebro del dron comprender su orientación o rumbo en relación con el campo magnético de la Tierra y, por lo tanto, hacia dónde viaja. Una consideración importante, es la conexión que se tiene con la placa de control de vuelo PIXHAWK 2.4.8 ya que se utiliza un GPS que viene incorporado un compás el cuál se configura de acuerdo por donde se enviara los datos esto utilizando el puerto I2C que utiliza este mismo protocolo de comunicación.

Para la calibración se debió ir al apartado de Setup > Mandatory Hardware > Compass, para después iniciar a calibrar cliqueando en “START” como se visualiza en la Figura 45.

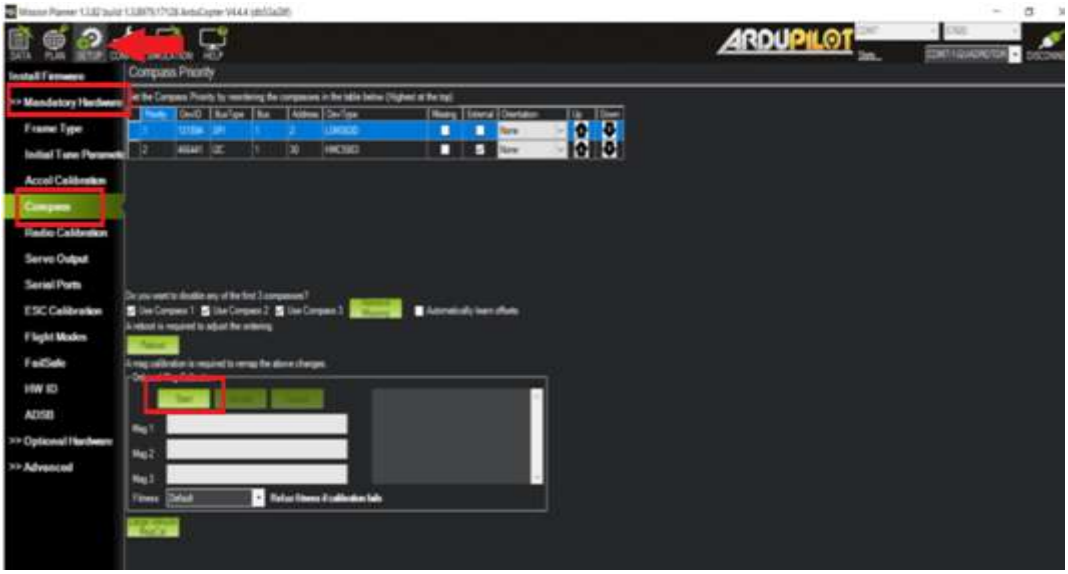


Figura 45. Calibración Compass iniciación en el software Mission Planner

De forma que se debió mover el dron en diferentes posiciones, girarlo en su propio eje hasta a ver visualizado que este cargada la barra de “Mag 1” y “Mag 2” se muestra cada uno de los apartados y como se inició la calibración en la Figura 45.

Para después se realizó el movimiento del dron continuamente en todas las direcciones girándolo en su propio eje esto sin descenderlo al suelo como se visualiza en la Figura 46, esto hasta que se haya completado las dos barras que se muestran en el software como se indica en la Figura 47.

Así, entonces sonó un pitido como alarma en el dron, se requirió reiniciar el UAV presionando en “Reboot” y se obtuvo un reinició en la placa, para después se visualizó el indicador LED de color verde en la placa Pixhawk 2.4.8.

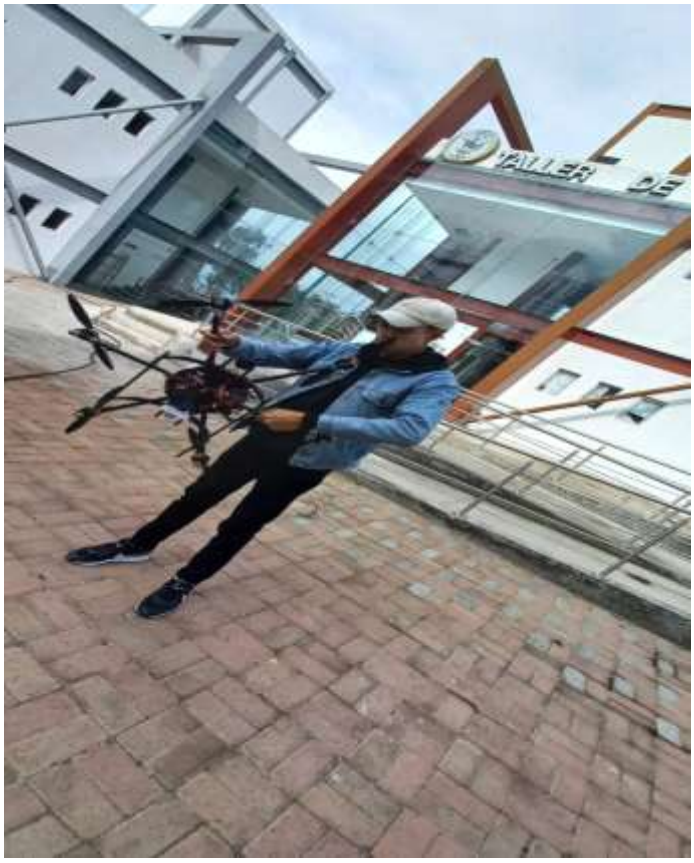


Figura 46. Movimiento del dron para calibración del Compas en físico



Figura 47. Calibración completada del Compas satisfactoriamente

Calibración de los controles de velocidad electrónicos (ESC)

Para la calibración del ESC, hay que tener en cuenta que estos elementos van conectados a los motores y son fundamentales en cuanto el control de los motores del dron, se puede describir al ESC (Control Electrónico de Velocidad), también conocido como controlador, es un dispositivo que recibe señales de la placa de controladora de vuelo y envía la potencia adecuada a los motores.

De esta manera, para calibrar los ESC se debió ir a “Setup” al apartado de Mandatory Hardware > ESC Calibration, y se presionó en “Calibrate ESCs” para iniciar con la calibración, se tiene algunas consideraciones entre la selección del tipo de ESC que en este caso será normal, las salidas de PWM tanto Mínimo y Máximo de acuerdo a los valores de recepción, además del parámetro “Spin” cuando se arma el dron así también valores mínimos y máximos de “Spin” que es la velocidad de los motores fueron esto fue necesario para el vuelo estable del dron.

Las consideraciones o pasos a tener en cuenta para calibrar el ESC se visualizaran en la Figura 48, una vez que se presionó el botón “Calibrate ESCs”, se desconectó el USB del módulo de telemetría o la conexión USB que tenga hacía la placa, después se desconectó la alimentación del dron y se volvió a conectar, como paso siguiente se presionó el switch o botón de seguridad que está conectado a la placa Pixhawk 2.4.8 hasta que suene un pitido constante y flashee el led indicador de la placa, así se quitó el modo de seguridad del dron después se escuchó un tono en específico y se completó la calibración del ESC, es necesario a ver reiniciado el radio control y reiniciado la placa Pixhawk 2.4.8.

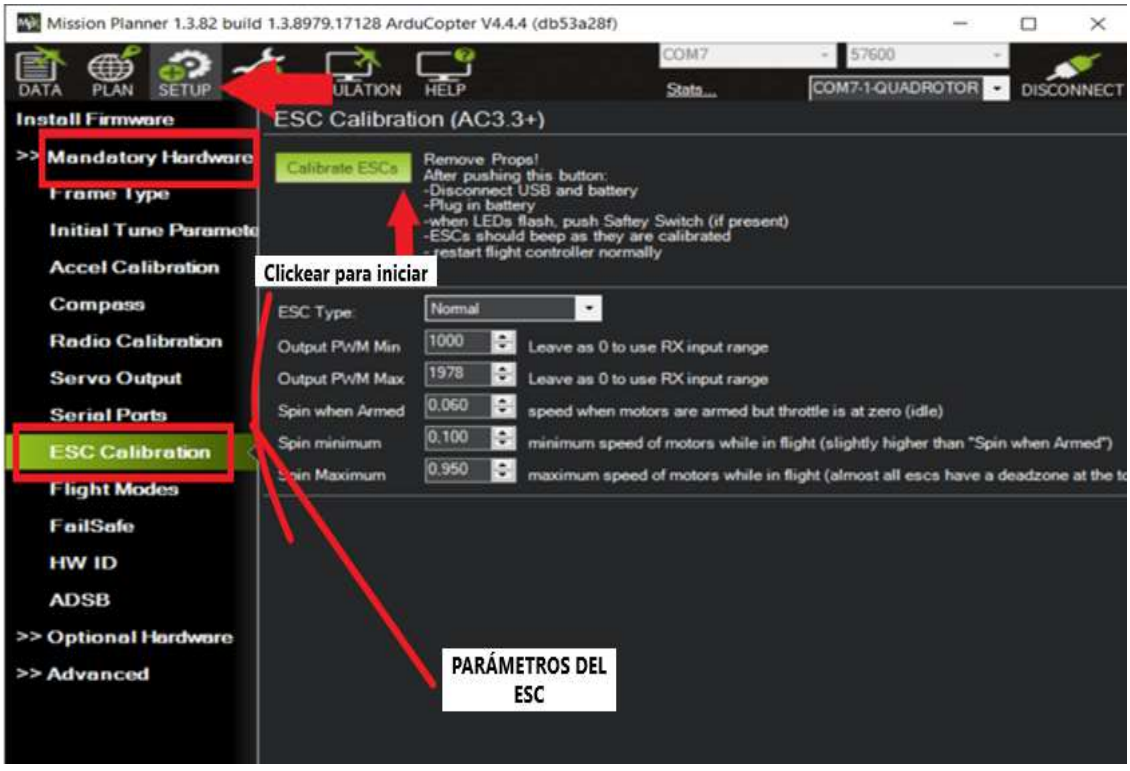


Figura 48. Calibración del ESC iniciación en el software Mission Planner

Calibración del radio control TX16S Radio Máster

El radio control es utilizado como el control de movimiento y orientación del dron, esto además del control del acelerador, cabeceo, balanceo y guiñada además del control de señales son mapeadas o transmitidas por algún stick/switch(s) y, a su vez, la habilitación de algunos de los canales para modos de vuelo y armado, para tener un control que será necesario en el modo AUTO.

Algunas de las especificaciones técnicas que se consideraron, son la frecuencia de operación del radio receptor FrSky R-XSR de 2,4 GHz la modulación utilizada es DSSS (Direct Sequence Spread Spectrum) y FHSS (Frequency Hopping Spread Spectrum), esto respecto al protocolo SBUS (Serial Bus) que permitió la configuración de 16 canales y el radio control Radiomaster TX-16 que trabajo a una frecuencia de 2,4 GHz, la compatibilidad al tener un sistema multiprotocolo RF interno con los módulos RF Chip CC2500 (FrSky), NRF24L01, A7105 (Flysky) y CYRF6936 (ASSAN) ayudo a la conexión con el módulo radio receptor FrSky R-XSR.

Para la calibración de cada uno de los controles/canales del transmisor, se debió mover cada uno de los joysticks/interruptores habilitados a lo largo de su rango completo y se registró las posiciones máximas y mínimas.

Para empezar a calibrar, se debió tener en cuenta la conexión correcta desde el software Mission Planner con la placa Pixhawk 2.4.8, para después a ver dirigido a “SETUP” a la parte de Mandatory Hardware > Radio Calibration, en este menú se pudo visualizar indicadores los cuales muestran los canales de radio habilitados y los joysticks para el control de movimiento del dron estos pasos se visualizan en la Figura 49.

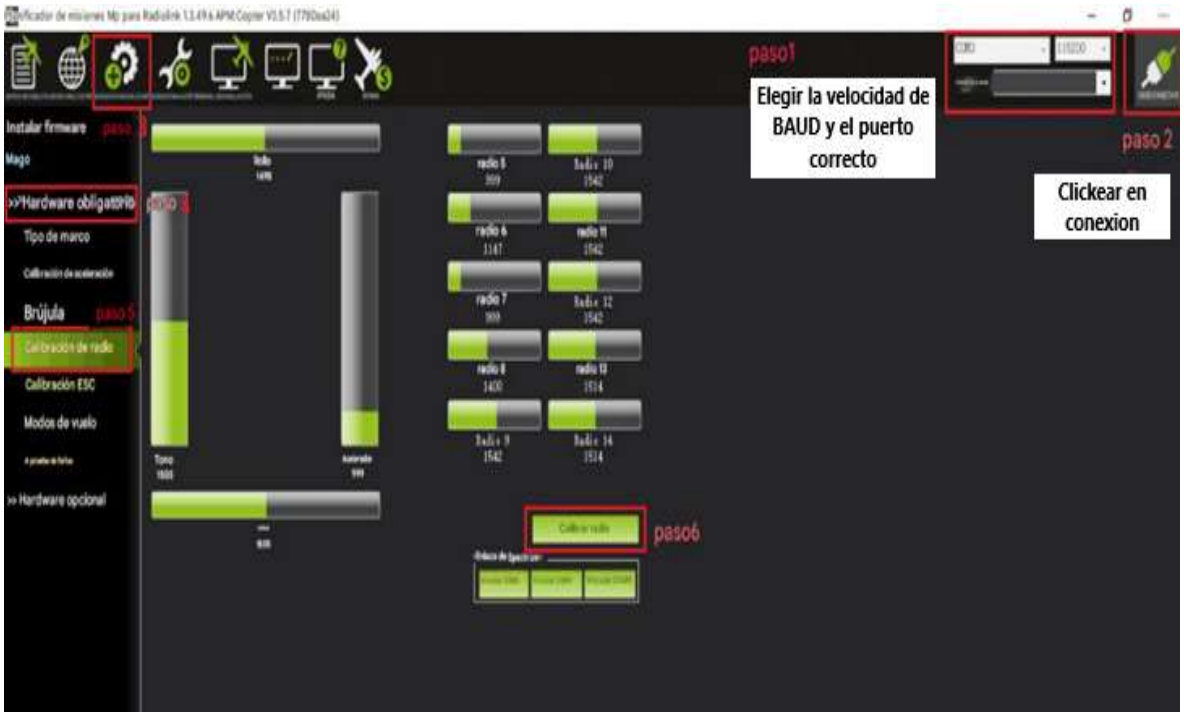


Figura 49. Calibración del Radio Control iniciación en el software Mission Planner

Se comprobó si cada barra roja como se visualiza en la Figura 50, en cada indicador correspondiente para cada canal funciona se muestra los controles principales de los mandos del radio control como se muestra a continuación:

- CH1: posición baja = balanceo (hacia la izquierda), posición arriba = balanceo (hacia la derecha).
- CH2: posición baja = cabeceo (hacia adelante), posición arriba = cabeceo (hacia atrás).
- CH3: posición baja = velocidad reducida, posición arriba = aceleración.
- CH4: posición baja = guiñada (hacia la izquierda), posición arriba = guiñada (hacia la derecha).

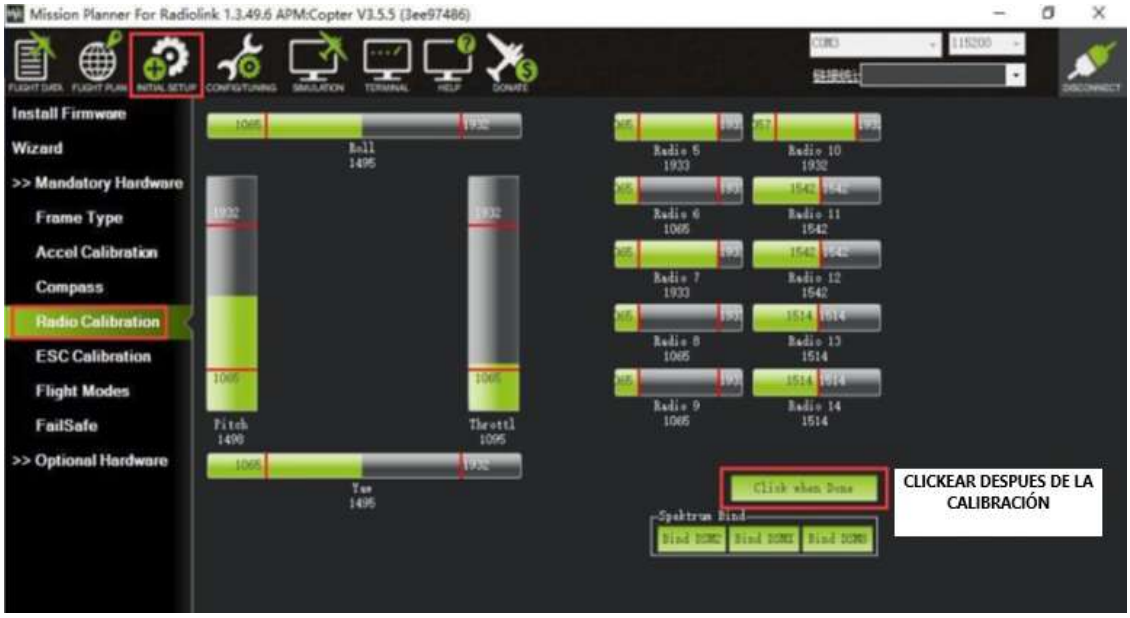


Figura 50. Calibración del Radio control.

Si las barras rojas no cambian cuando se mueven los joysticks, se comprobó si el receptor se ha conectado correctamente o no, se debió asegurar de que el receptor el módulo Radioreceptor FrSky R-XSR emita una señal SBUS (el LED azul indico que funciona como señal SBUS) como se visualiza en la Figura 51.

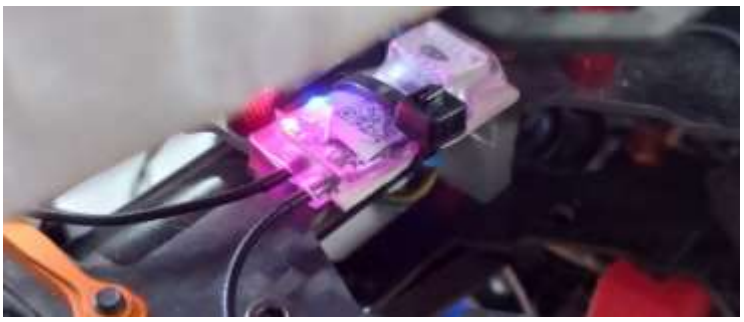


Figura 51. Módulo Radioreceptor FrSky R-XSR encendido

Se presionó “OK” una vez comprobado y calibrado de forma correcta guardando los datos de calibración, y se abrió una ventana emergente indicando los canales el radio control como se visualiza en la Figura 52.

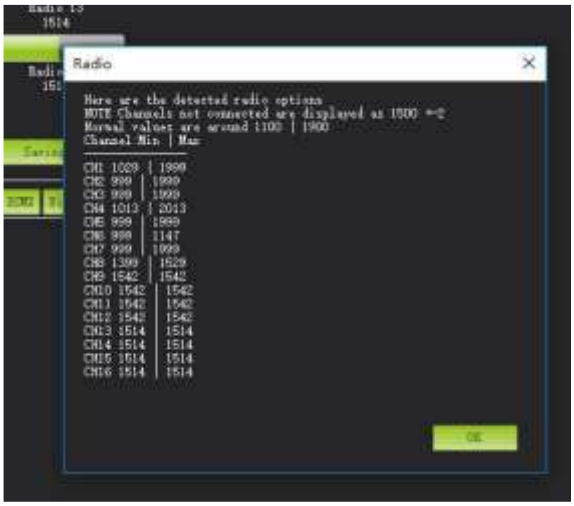


Figura 52. Confirmación para la calibración del radio control

Configuración de modos de vuelo en el software Mission Planner

La placa PIXHAWK 2.4.8 tiene la posibilidad de 20 modos de vuelo integrados, 10 de los cuales se utilizan habitualmente. Hay varios modos que admiten diferentes niveles/tipos de estabilidad de vuelo, pilotos automáticos complejos, sistemas de seguimiento, etc. El modo de vuelo se controló por radio (a través del contacto del transmisor), comandos de misión o comandos de aeronave mediante la estación terrestre (GCS) o computadora adicional.

Para empezar con la configuración de modos de vuelo algunos de los pasos se muestran en la Figura 53, para ello se debió dirigir en “SETUP” hacía Mandatory Hardware > Flight Modes que es el menú de modos de vuelo, cada modo de vuelo se activó a un intervalo numérico PWM de los tres modos de vuelo que se ha configurado. En base a investigaciones previas, el primer modo de vuelo se configuró en “Stabilize”, para los demás modos adecuar de acuerdo con las necesidades de vuelo requeridas, que en este caso fue necesario establecer un modo de vuelo “Auto” y un modo de vuelo de fotografía conocido como “Loiter”.



Figura 53. Configuración de Modos de vuelo dentro del software Mission Planner

Se cargó algunos parámetros necesarios para habilitar el cambio de modo de vuelo los parámetros o modos de vuelo se puede cambiar como antes se menciono, lo que es necesario es indicar por cuál canal de RC del radio control estara enviando esa información, para aquello se cambio el parámetro FTLMODE_CH al valor 10 ya que en las opciones nos indica el número de canales como se observa en la Figura 54.

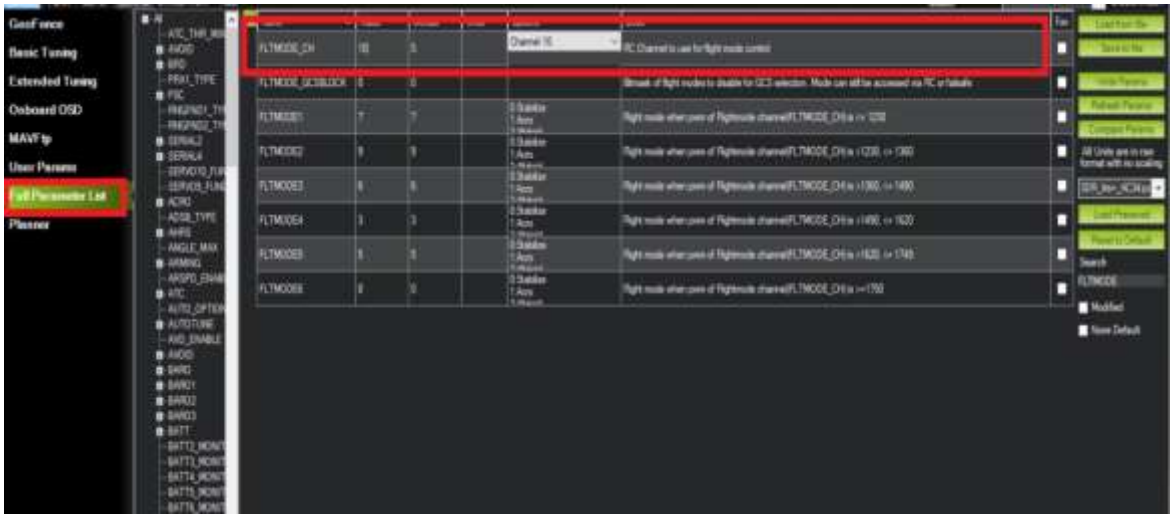


Figura 54. Cargar parámetros para el canal de RC para cambios de Modo de vuelo dentro del software Mission Planner

Después se debió configurar el modo de vuelo en el transmisor para aquello se estableció conexión RC hacía la placa Pixhawk con el Radio control. Para a ver redirigido a la configuración dentro del radio control como se visualiza en la Figura 55, para habilitar el canal RC de acuerdo al Switch/perilla que se requiera los cambio de modos de vuelo fue necesario configurar el switch SD del CH10 como se observa en la Figura 56, se tiene en cuenta el valor de PWM antes mencionado para configurar que se haya ajustado a un valor en específico para el cambio de modos de vuelo, así se pudo comprobar con un pitido de alarma el cambio respectivo de cada modo de vuelo por parte de la placa Pixhawk y visualizado en el software como en la pantalla OLED integrada, los cambios de modo de vuelo.

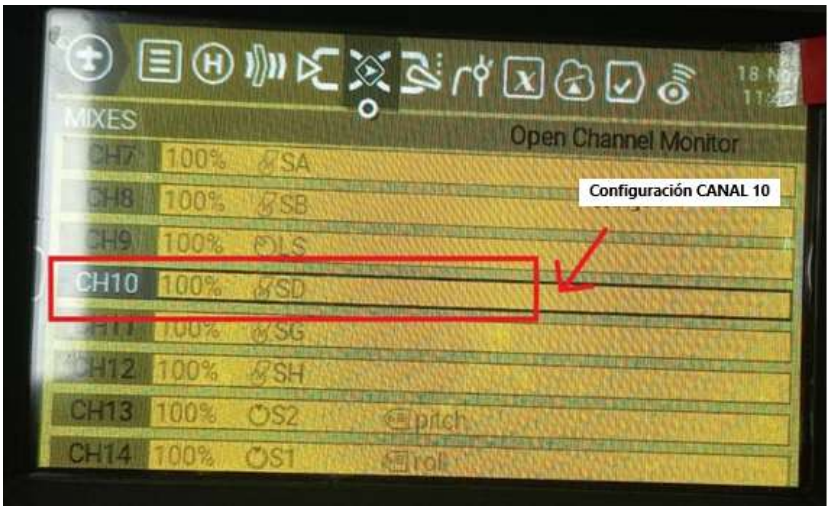


Figura 55. Configuración de Canal en la interfaz del radio control TX16 RadioMaster



Figura 56. Configuración del interruptor para los tres modos de vuelo en la interfaz del radio control TX16 RadioMaster

Modo auto (modo de vuelo autónomo)

En el modo automático, el dron seguirá los pasos programados en la misión preprogramada anteriormente y almacenado en el piloto automático que se compone de comandos de navegación (es decir, puntos de referencia de la ruta establecida) y comandos "DO" (es decir, comandos que no afectan la ubicación del helicóptero u otra acción en específico programada) estos comandos fueron programados anteriormente en el software Mission Planner, configurados en el apartado de “PLAN” como se puede visualizar en la Figura 57,



Figura 57. Apartado de Plan (Planeo de misiones) dentro del software Mission Planner

También se pudo configurar por la interfaz de monitoreo y control en Python mediante simulaciones de georreferencia, este es la base de la configuración para vuelo automático de trayectorias con el UAV, se tuvo como un inicio para poder cumplir cierta misión de acuerdo a la imagen que se visualiza en la Figura 58.

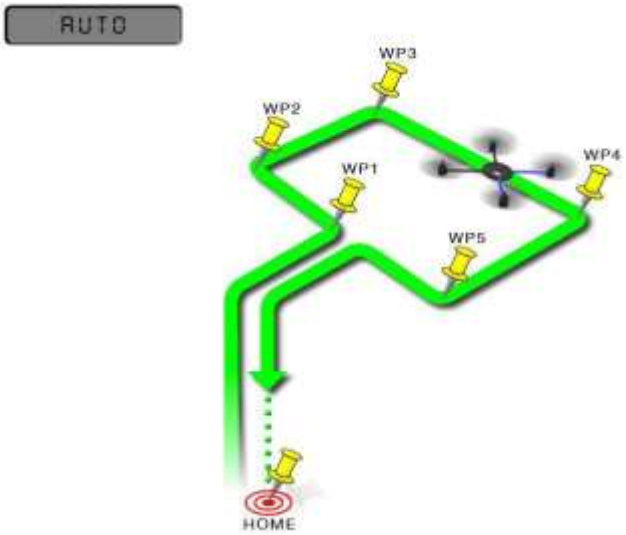


Figura 58. Diagrama con waypoints de una misión para el UAV

Respecto al modo AUTO se tuvo algunas funciones específicas para el funcionamiento correcto, este se incorporó para el control de altitud del modo AltHold y el control de posición del modo Loiter. Se aplicó los mismos requisitos antes mencionados para un vuelo estable, incluidos el a ver asegurado los niveles de vibración y de interferencia de la brújula en un rango aceptable y que el GPS tenga correcto funcionamiento, es decir una recepción de satélites de entre 8 a 12 esto de acuerdo con el módulo GPS MN8 PX4 incorporado en el dron.

Se configuró anteriormente el modo AUTO como uno de los modos de vuelo en el interruptor de modo de vuelo. Para comprobar el funcionamiento estable se verificó que el GPS haya estado posicionado para aquello se tiene algunas consideraciones siguientes:

El LED de la PIXHAWK está verde que será de armado como se observa en la Figura 59.



Figura 59. Armado del UAV (Led verde)

El LED indicador del módulo GPS y brújula está parpadeando para verificar que no tiene errores con los mismos, este es de color azul como se observa en la Figura 60.



Figura 60. Indicador GPS/BRUJULA del UAV (Led azul)

c. Sistema Electrónico FPV para videovigilancia y operaciones en el espacio aéreo.

El sistema FPV (First Person View) se compone principalmente de una cámara instalada en el dron, un transmisor de video que envió la señal, una antena adecuada para maximizar el rango y estabilidad, un receptor que se conectó a un monitor FPV, para el sistema de alimentación se utiliza una placa de distribución de energía FCHUB-A, se realizó una conexión con la batería que alimenta al dron, la estructura completa del esquema electrónico se visualiza en la Figura 61 .

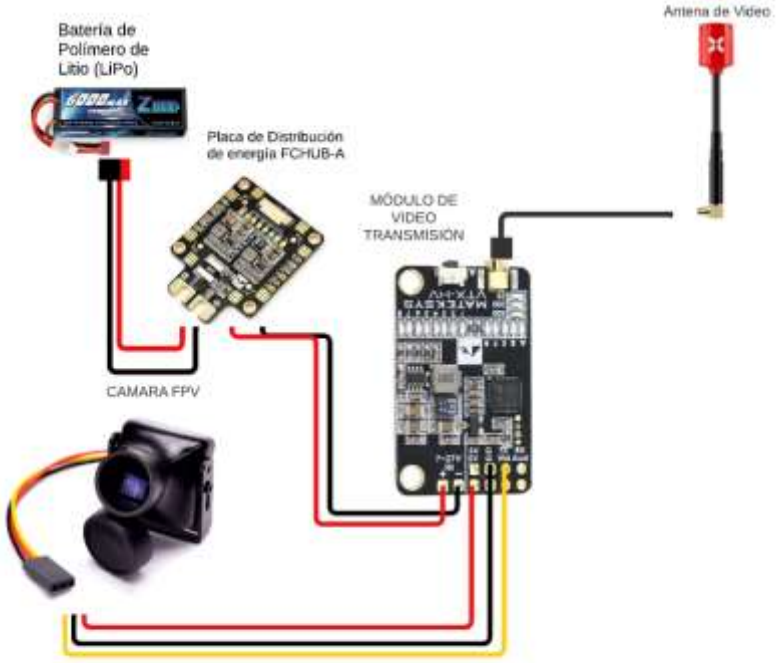


Figura 61. Esquema electrónico del Sistema FPV

Configuración del sistema FPV

Para proceder a la configuración se debió asegurar que la cámara esté bien montada y orientada, se seleccionó las frecuencias compatibles entre transmisor y receptor, y ajusto la calidad de transmisión según las necesidades (resolución, rango o latencia). Además, se verifico que el sistema esté correctamente alimentado y se realizaron pruebas de interferencias una vez conectada al monitor FPV para garantizar una transmisión estable durante el vuelo. En la Figura 62, se muestra el proceso realizado para la configuración y puesta en marcha del sistema completo FPV.

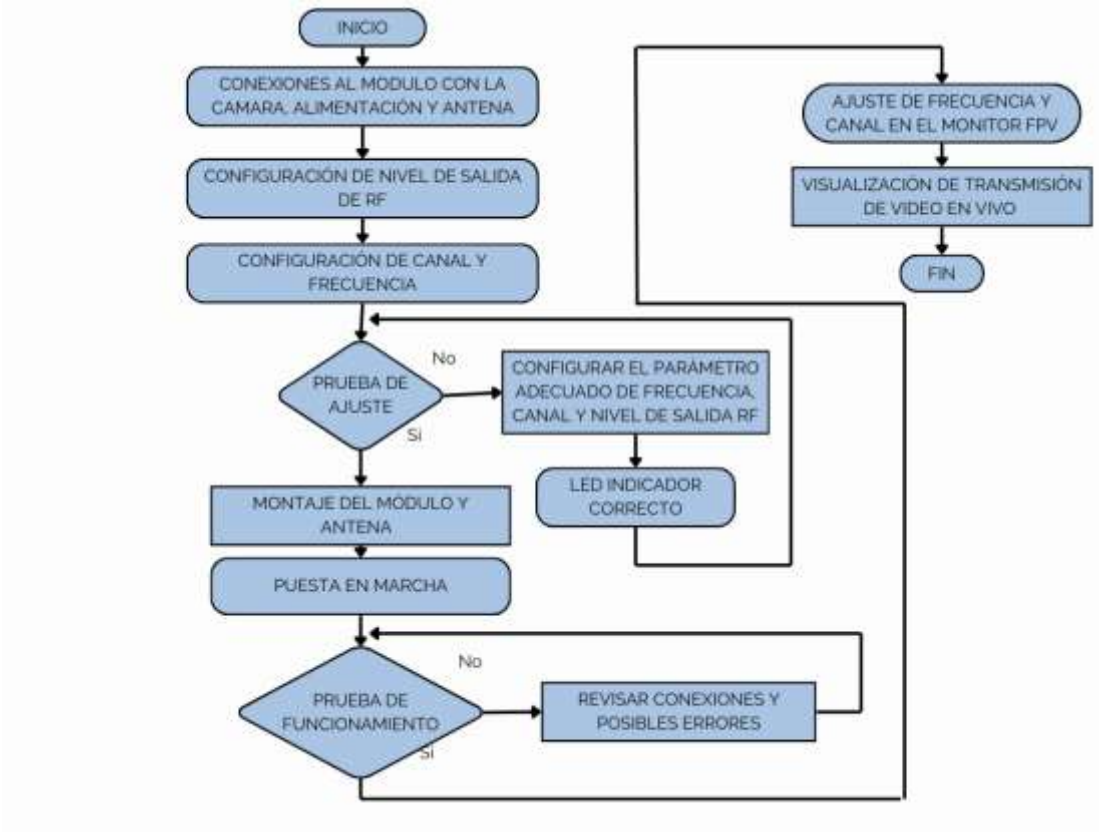


Figura 62. Diagrama de flujo de la configuración y montaje del sistema FPV

Se presionó el botón A como se visualiza en la Figura 63 para seleccionar los niveles de salida de Radio frecuencia, esto como el nivel 0 (LED azul) que es el modo pit y su valor corresponde a 0,1 mw, el nivel 1 que corresponde al valor de 25 mw, el nivel 2 que corresponde al valor de 200 mw y el nivel 3 que corresponde al valor de 800 mw.

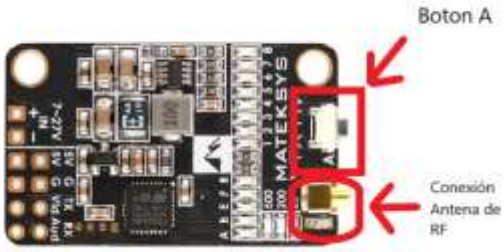


Figura 63. Botón A de la placa MATEK HV 5.8 Ghz

Se mantiene presionado el botón B que se visualiza en la Figura 64, durante 2 segundos para la configuración posterior.



Figura 64. Botón B de la placa MATEK HV 5.8 Ghz

Para la configuración de frecuencia se debió tener en cuenta la Tabla 18 para los niveles de consignación de frecuencias, la frecuencia de trabajo que se selecciono es de 5,800 MHz. El LED de la frecuencia parpadeo, para después a ver presionado nuevamente el botón B para seleccionar la frecuencia.

Tabla 18. Frecuencias de operación (MHz)

BFOSD	**LED**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
BOSCAM A	**A**	5865	5845	5825	5805	5785	5765	5745	5725
BOSCAM B	**B**	5733	5752	5771	5790	5809	5828	5847	5866
BOSCAM C	**E**	5705	5685	5665	5645	5885	5905	5925	5945
FatShark	**F**	5740	5760	5780	5800	5820	5840	5860	5880
RaceBand	**R**	5658	5695	5732	5769	5806	5843	5880	5917

Cuando el LED parpadeo a una nueva frecuencia, no hay salida de RF en esta frecuencia. Se tuvo en consideración que se habilitarán nuevas frecuencias y salidas al no tener una acción ningún botón en 3 segundos y se pudo visualizar de acuerdo con los leds indicadores a que frecuencia se estableció la configuración como se muestra en la Figura 65.

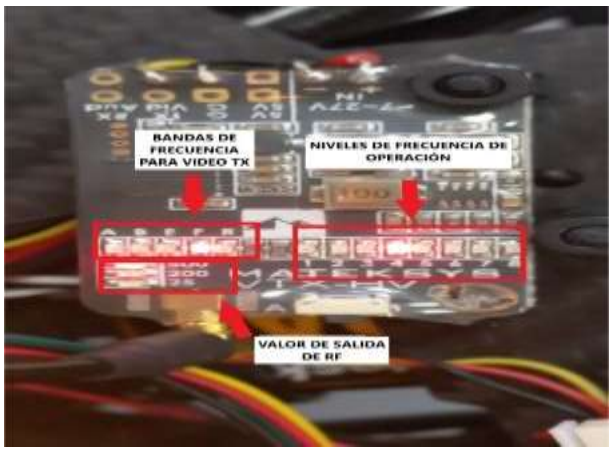


Figura 65. Visualización del módulo de video transmisión en funcionamiento

Una vez ajustado el módulo de video transmisión se debió buscar el canal y la frecuencia a la cuál esta ajustada en el monitor FPV, la pantalla FPV utiliza cierto número de canales y frecuencias de trabajó, se ajustó a la frecuencia de 5820 MHz del CH-5-D, estas se especifican en la Tabla 19, una vez se configuro se visualiza el video por el monitor como en la Figura 66.

Tabla 19. Selección de banda y canal del monitor FPV

CH GRUPO	CH1	CH2	CH3	CH4	CH5	CH6	CH7	CH8
A	5865 MHz	5845 MHz	5825 MHz	5805 MHz	5785 MHz	5765 MHz	5745 MHz	5725 MHz
B	5733 MHz	5752 MHz	5771 MHz	5790 MHz	5809 MHz	5828 MHz	5828 MHz	5866 MHz
C	5705 MHz	5685 MHz	5665 MHz	5645 MHz	5885 MHz	5905 MHz	5925 MHz	5945 MHz
D	5740 MHz	5760 MHz	5780 MHz	5800 MHz	5820 MHz	5840 MHz	5860 MHz	5880 MHz
R	5658 MHz	5695 MHz	5732 MHz	5769 MHz	5806 MHz	5843 MHz	5880 MHz	5917 MHz



Figura 66. Configuración canal y frecuencia en el monitor FPV

Para la validación de la conexión del sistema FPV, se tomó diferentes pruebas respecto a la distancia que se encuentra el monitor FPV con el UAV para aquello se tomaron algunas observaciones del video receptado a tanta distancia estos datos se observan en la Tabla 20.

Tabla 20. Distancias de conexión respecto al sistema FPV con el monitor FPV

Prueba	Distancia entre UAV y monitor FPV	Observaciones
1	100 m	Perfecta conexión sin interferencias
2	150 m	Perfecta conexión sin interferencias
3	300 m	Conexion estable
4	500 m	Conexion estable
5	650 m	Conexion regular
6	700 m	Conexion regular

d. Configuración de la placa de control del GIMBAL para montaje de la cámara GoPro-Hero 5 para videovigilancia y operaciones aéreas

Configuración de hardware y software del gimbal

Para la configuración de la placa de control del gimbal este es un dispositivo estabilizador que mantiene a la cámara de un dron nivelada durante el vuelo, compensa movimientos no deseados causados por vibraciones o cambios de orientación.

Entonces se detalla a continuación los pasos seguidos para la configuración, fue necesario una configuración tanto para la parte del hardware como del software que son necesarios para tener todo el sistema de control montado y en funcionamiento, de acuerdo con el diagrama de flujo de la Figura 67 se siguieron aquellos pasos.

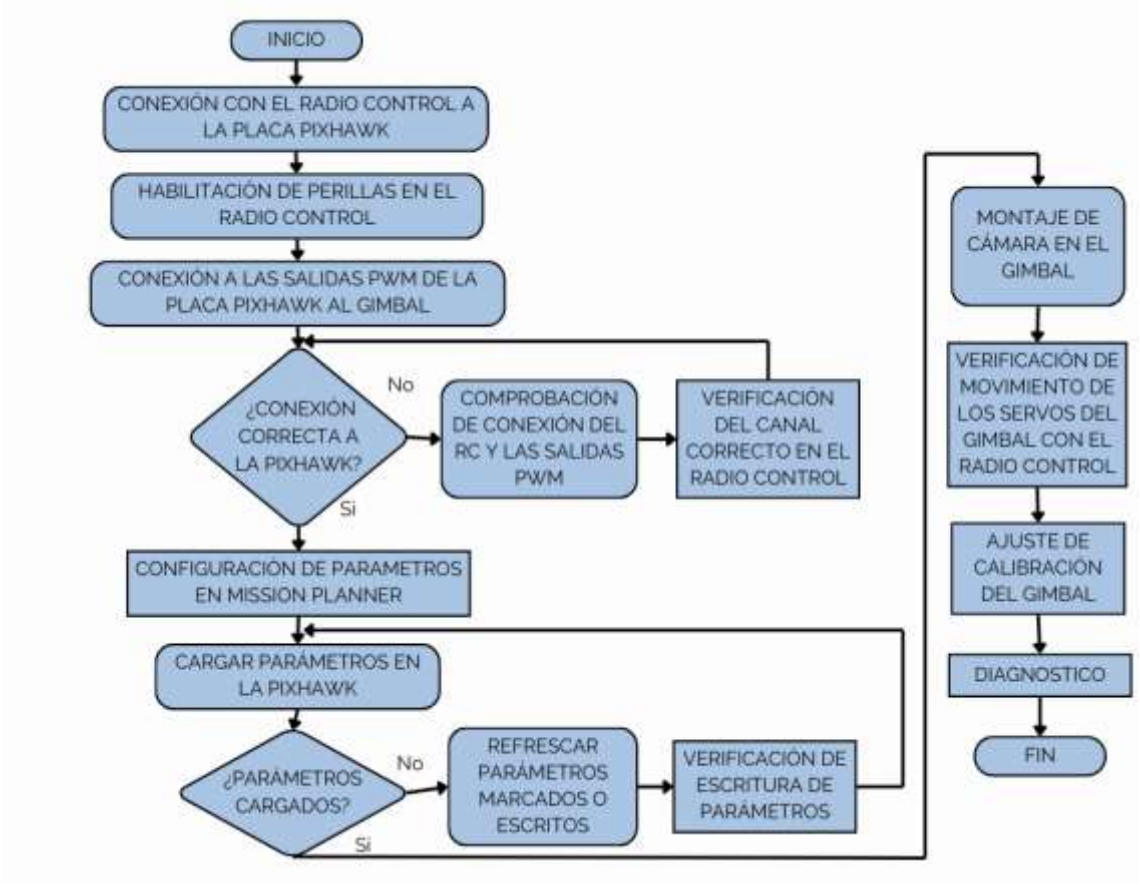


Figura 67. Diagrama de flujo de configuraciones y pruebas del gimbal

Para la configuración del Gimbal, con referencia a la placa de control PAMENET Controlador de cardán BGC 3.1 de 2 ejes que se escogió para montar el gimbal en el dron, se configuró primeramente la calibración de acuerdo con ajustes de parámetros iniciales de PID de entre P = 15, I= 0,05 y D = 5 para el comportamiento de perturbaciones del gimbal esto específicamente para tomas de video sin vibraciones para tener como resultado un video con el menor ruido posible, este vino conectado a dos servo motores en base a dos ejes o una estructura para ubicar la cámara, el acelerómetro fue montado debajo de la placa de sostén de la cámara y esté conectado a la placa de control de señales las cuales se enviaran a la placa principal Pixhawk que tomo las señales PWM de pitch y roll que irán conectados a la misma, la alimentación se realiza con una batería Lipo de 3S de 11.1V toda la conexión se muestra en el diagrama de la Figura 68.

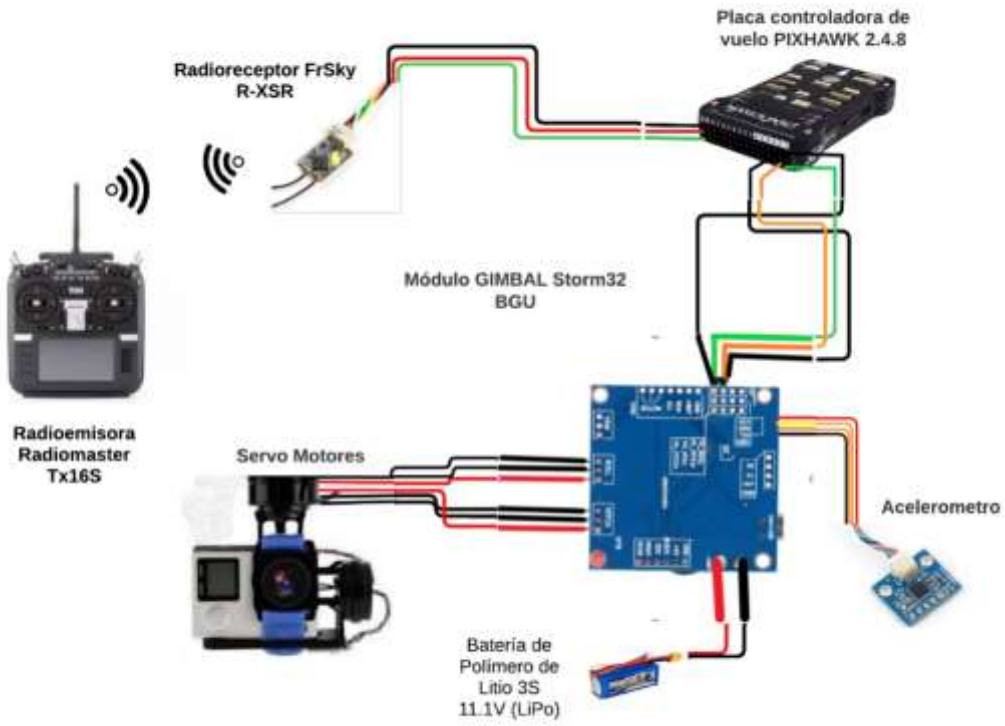


Figura 68. Esquema electrónico del sistema del gimbal

Configuración del firmware de la placa de control PAMENET Controlador de cardán BGC 3.1del gimbal.

Se empezó por cargar el firmware dentro de la placa de control de forma que se pueda programar y calibrar solo con el software libre creado específicamente para configurar y calibrar para placa de control del gimbal, se empezó descargando el archivo "null.hex" esto con la utilidad de un software crackeado que utiliza la placa de control.

Se necesito un programador AVR, estas son herramientas para cargar programas (firmware) antes mencionado para microcontroladores AVR, se utilizó un módulo USBtinyisp para esta tarea en específico. Para poder manejar el módulo se usó el software AVRdude esta es una herramienta de línea de comandos para tener conexión con el módulo antes mencionado permitiendo la carga de programas (firmware), un paso antes de empezar a configurar se probó un correcto funcionamiento de la placa de control solo visualizando los leds indicadores de la misma como se muestra en la Figura 69.

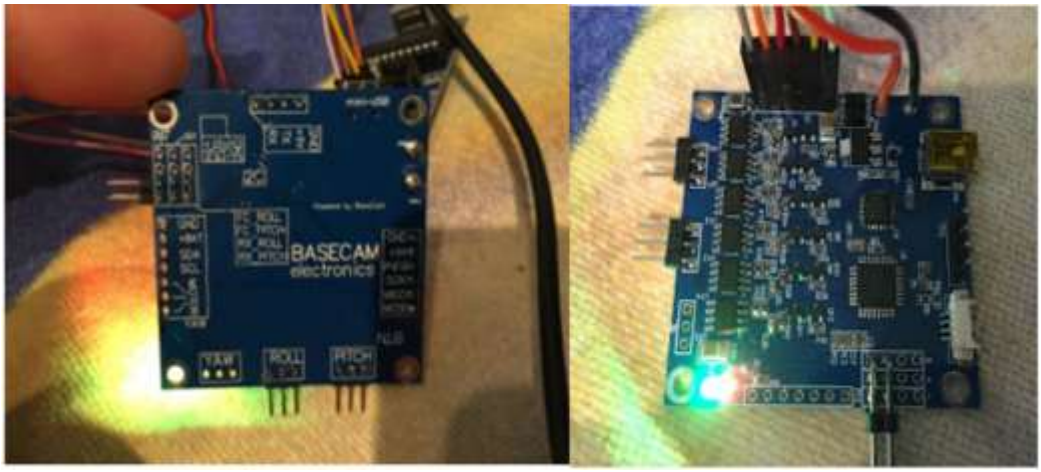


Figura 69. Indicadores Led y muestra de la placa de control del Gimbal

Posteriormente, se accedió al software AVRdude a través de la línea de comandos (CMD) en el ordenador. Una vez establecida la conexión, se observó una ventana emergente, como se muestra en la Figura 70.

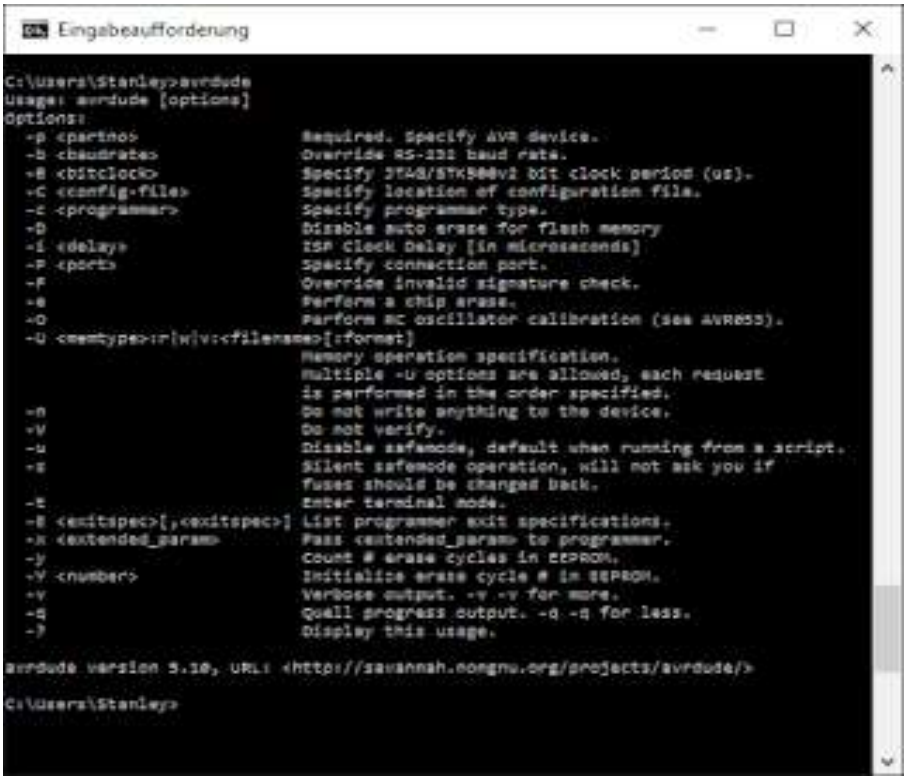


Figura 70. Ingreso por cmd al software AVRdude

A continuación, se conectó físicamente el módulo USBtinyISP a la placa de control, asegurándose de que esta no estuviera conectada a las escobillas ni a los servomotores del gimbal. Las conexiones realizadas se detallan en la Figura 71.

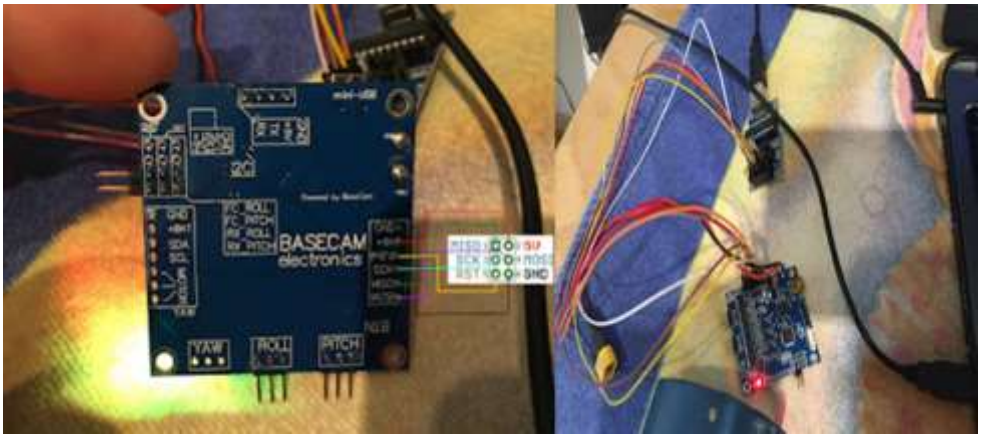


Figura 71. Conexiones de la placa de control con el módulo USBtinyisp

Después se ingresó algunos comandos necesarios para la conexión y carga del programa; se empezó ingresando en cmd del ordenador antes mencionado y se ingresó el siguiente comando:

avrdude -c usbtiny -p atmega328p

Una vez se realizó la conexión, para la comprobación se visualizó en la ventana los mensajes como se indica en la Figura 72.

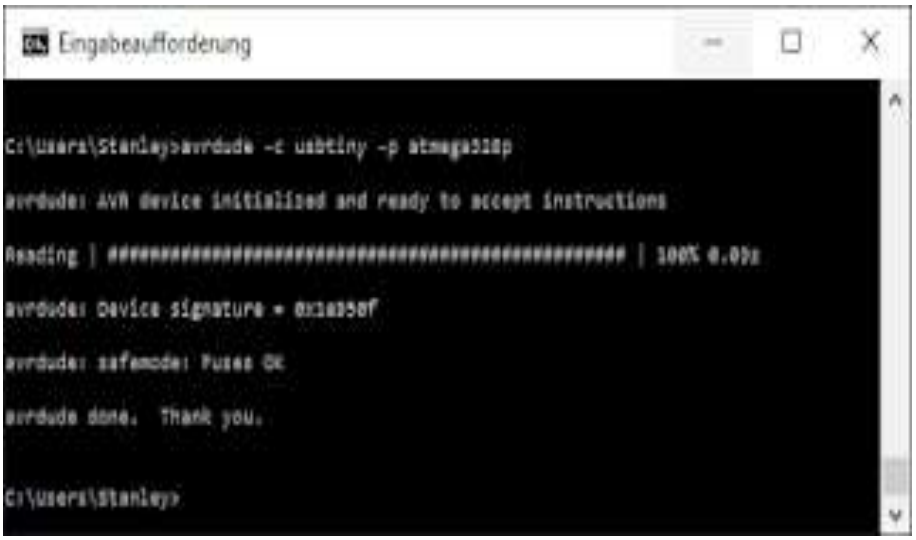


Figura 72. Ingreso de comandos por cmd en AVRdude para la configuración del firmware de la placa de control

Luego se ingresó después la confirmación y código respecto a los puertos de la placa de control para habilitar el firmware y el archivo “null.hex” descargado anteriormente, el comando siguiente ingresado fue:

avrdude avrdude.conf -c usbtiny -p atmega328p -e -u -U lock:w:0x3f:m -U efuse:w:0x05:m -U hfuse:w:0xDC:m -U lfuse:w:0xEE:m

Se visualizó la configuración al mandar el comando como se indica en la Figura 73.

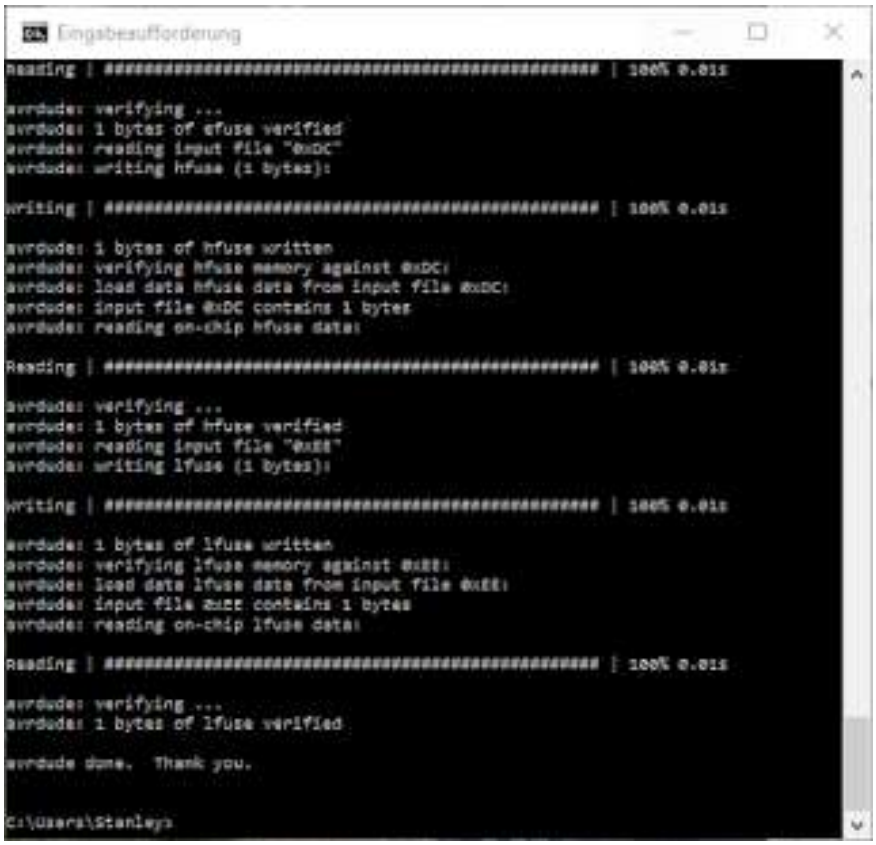


Figura 73. Ingreso de comando por cmd en AVRdude para la configuración de paquetes hexadecimal de la placa de control

Como siguiente paso se redirigió otra vez a la herramienta AVRdude por cmd, se agregó la ruta del archivo hexadecimal, el comando siguiente ingresado fue:

*avrdude avrdude.conf -c usbtiny -p atmega328p -U flash:w:"**C:\Download**\simplebgc_2_2_b2_null.hex":i -U lock:w:0x0C:m*

El cargar el firmware tomó un tiempo considerable, una vez finalizo la carga se visualizará un mensaje de gratificación como se muestra en la Figura 74.

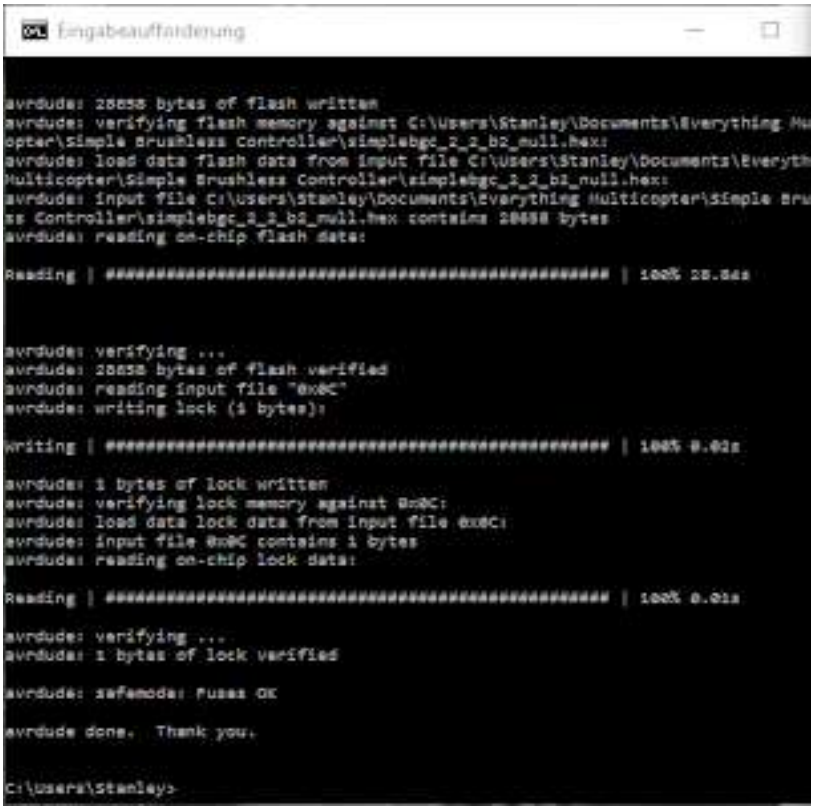


Figura 74. Ingreso de comando por CMD en AVRdude para cargar el firmware descargado del archivo hexadecimal de la placa de control

Se desconecto el módulo USBtinyisp, y se pasó a la calibración con el software libre SimpleBGC de la placa de control.

Calibración del gimbal mediante el software SimpleBGC_GUI_2.2 y montaje del gimbal al UAV.

Una vez cargado el firmware se programó la placa de control con el software libre SimpleBGC, la interfaz gráfica de usuario se muestra en la Figura 75 se tiene en cuenta algunos de los menús para las configuraciones como las básicas, avanzadas, configuraciones de RC (radio control) y graficas de comportamiento en tiempo real del gimbal.

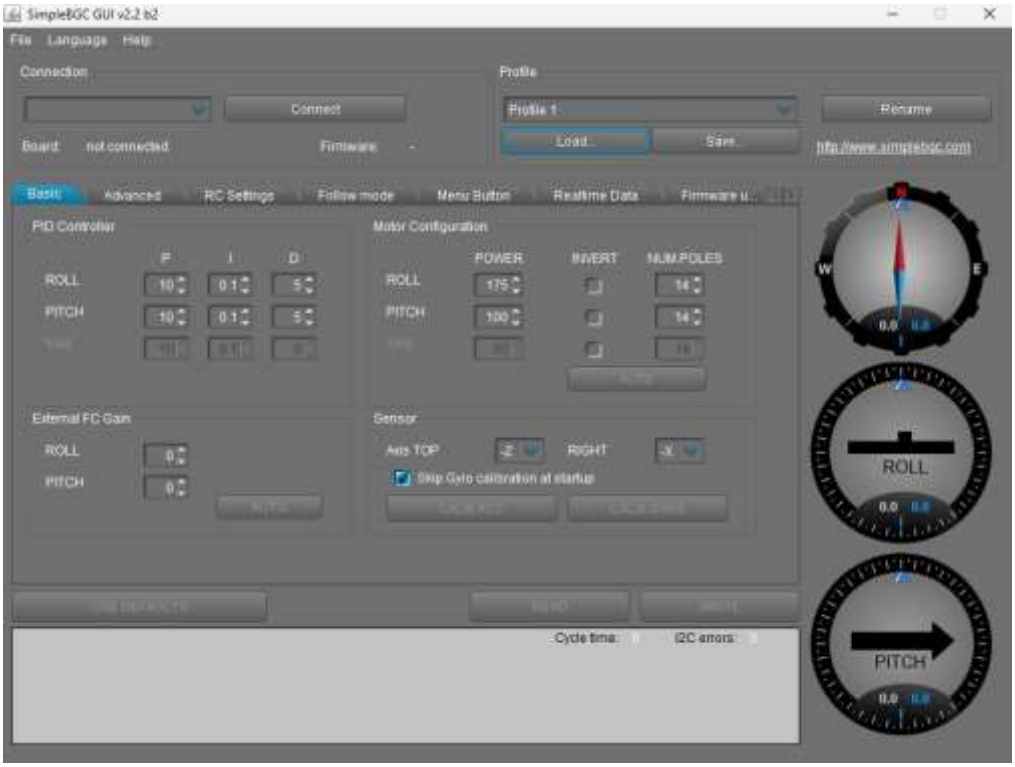


Figura 75. Interfaz Gráfica GUI para configuraciones de la placa de control del gimbal

Para iniciar se conectó por el puerto serial o USB del ordenador con la placa de control, para la calibración del acelerómetro y giroscopio fue necesario mantener un nivel estable la cámara sin la alimentación de la batería Lipo del sistema principal de la placa de control de vuelo, ya que se empezó calibrando el acelerómetro manteniendo la posición en la cual se va a grabar en base a una estabilidad óptima, para finalizar la calibración se presionó “CALIB ACC”. Por otro lado, para pasar a la calibración del giroscopio se visualizó que la brújula el puntero rojo este apuntado hacía norte y el puntero azul hacía al sur, así como paso final se presionó “CALIB GYRO” esto como se muestra en la Figura 76.

Luego, se procedió a la configuración básica necesaria para la calibración del gimbal. Durante este proceso, se utilizó el control PID, enfocado en manejar las perturbaciones, la velocidad de reacción y respuesta de acuerdo con los comandos del radiocontrol (RC). Además, se consideraron las vibraciones que surgieron tras montar el sistema y realizar pruebas de vuelo del dron. En las configuraciones básicas por defecto, se asignaron los siguientes valores: P = 10, I = 0.1 y D = 5; estos se aplicaron

tanto para los ejes de "pitch" como de "roll", que corresponden a las orientaciones controladas por los servomotores.

En la configuración de los motores, se ajustó la potencia adecuada para los servomotores. Para el eje "roll", fue necesario establecer un valor mayor de 175 W, como se muestra en la Figura 76, debido a la estructura que soportaba el peso de la cámara GoPro Hero 5. Por otro lado, para el eje "pitch", que controla el movimiento horizontal (derecha o izquierda) y el posicionamiento de la cámara, se respetaron los valores predeterminados recomendados en el manual técnico de la placa de control, tal como se detalla en el Anexo B.



Figura 76. Parámetros configurados para la calibración de la placa de control del gimbal

Una vez configurado los parámetros necesarios en la configuración básica de calibración del PID, y motor configuration fue necesario crear un perfil para guardar los datos cambiados o guardados. Se requirió guardarlo en un formato de archivo.perfil con un nombre en específico guardado como “calibración final” como se muestra en la Figura 77.

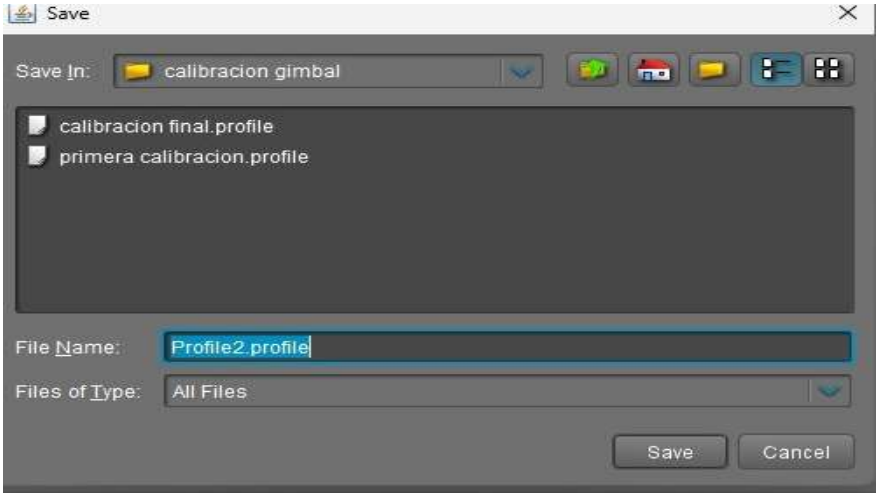


Figura 77. Guardar perfil de calibración dentro del software SimpleBGC_GUI_2.2

Se debió cargar los parámetros apretando el botón “WRITE” una vez guardado el perfil, así se obtuvo la acción de parpadeo de un led de color amarillo en la placa de control por unos segundos que indico que se cargó el perfil deseado. Luego, se alimentó la placa de control del gimbal desconectando la conexión USB de la placa de control al ordenador específicamente para realizar pruebas de funcionamiento, de esta manera se esperó 15 segundos a que se estabilizará el gimbal de forma automática.

Se montó el gimbal en el Dron, esto específicamente con cuatro soportes ya que se tuvo una estructura de soporte tanto para la placa, como para los servos motores que es todo lo que se compone el gimbal y en donde se montó la cámara como se muestra en la Figura 78 y la Figura 79.

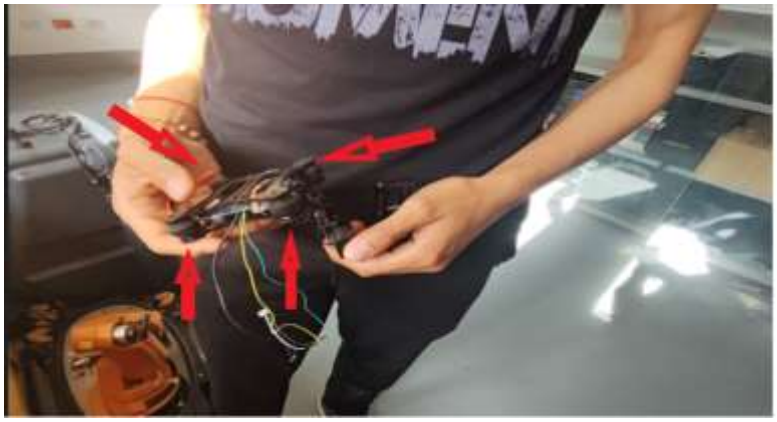


Figura 78. Gimbal en forma física desarmado

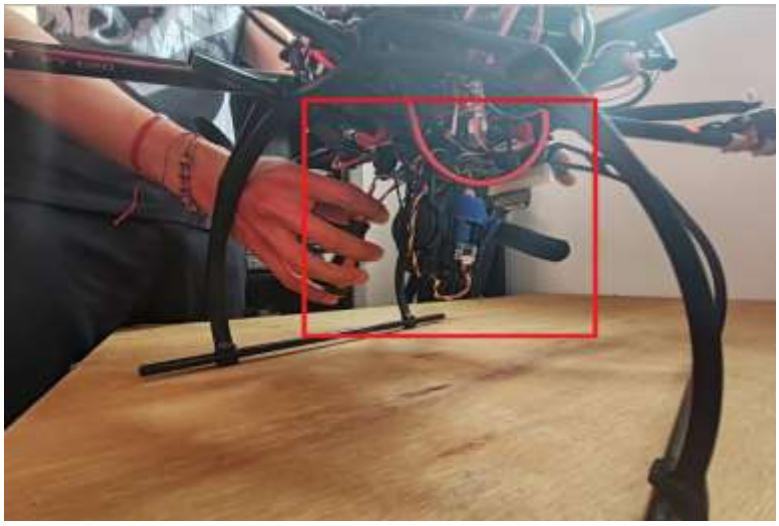


Figura 79. Montaje del gimbal en el dron

Una vez se ha montado el gimbal con la cámara GoPro Hero 5, el peso del gimbal es de 400 gr y la camara es de 117 gr entonces el peso aumentó 517 gr, desajustando el acelerómetro. Fue necesario la calibración de la placa de control y el gimbal con la cámara montada. Se reajusto la potencia del motor del “pitch” a 100 W como se indica en la interfaz del software SimpleBGC de la Figura 80, reduciendo vibraciones al encender el dron.



Figura 80. Cambio de parámetros de potencia dentro del software SimpleBGC_GUI_2.2

La gráfica de la Figura 81 muestra que, al conectar la placa sin calibrar se han obtenido altas perturbaciones en los servomotores y el acelerómetro. Fue necesario calibrar

completamente el gimbal para obtener grabaciones óptimas durante las pruebas de vuelo.



Figura 81. Grafica de perturbaciones en tiempo real de datos recogidos por el software SimpleBGU_GUI_2.2

Configuración de parámetros en el software Mission Planner para control de motores del gimbal por RC

Se configuró la placa controladora de vuelo para el control de movimiento de los servomotores siendo esta la placa principal dentro de la arquitectura del sistema, ya que es la encargada del envió de señales PWM por las salidas auxiliares de la misma. Se basa la configuración utilizando el software Mission Planner.

Primero, se configuraron los canales específicos dentro de la interfaz del radiocontrol TX16S RadioMaster para habilitar las perillas destinadas al movimiento de los servos, correspondientes a los ejes “pitch” y “roll”. Para ello, se accedió a la configuración de telemetría del control, específicamente al apartado de entradas (“Inputs”), donde se asignó la entrada cinco a “pitch” (S2) y la entrada seis a “roll” (S1), como se muestra en la Figura 82. Posteriormente, se verificó en la pantalla principal que los indicadores respondieran correctamente al movimiento de las perillas S1 y S2.

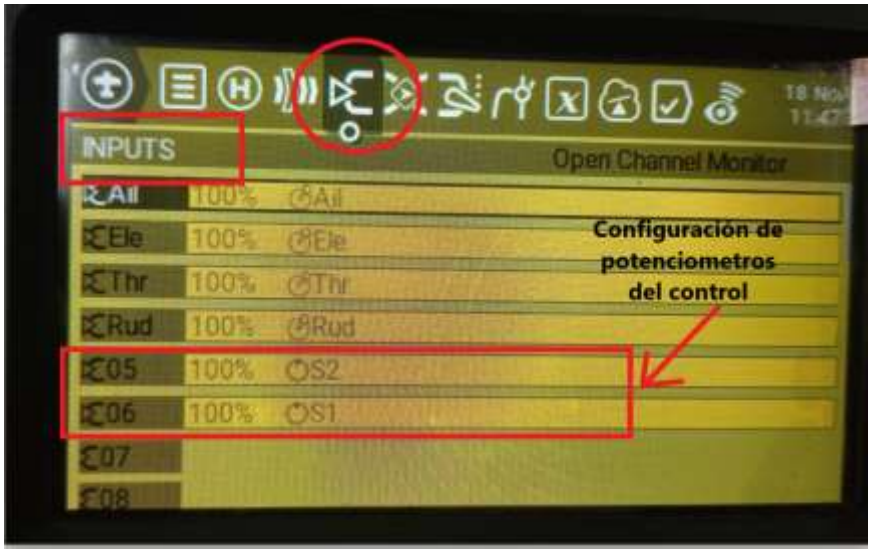


Figura 82. Configuración en la interfaz del radio control apartado de entradas en configuraciones de telemetría

Fue necesario además asignar canales para el control mediante el radio control de cada uno de los motores esto teniendo en cuenta que cada canal este libre, por lo tanto, para poder asignar cada canal para las perillas S1 y S2 para el movimiento de los servos motores. Después, se configuró el Canal 13 para “pitch” y el canal 14 para “roll” como se visualiza en la Figura 83.



Figura 83. Configuración de canales en mixes dentro de la interfaz del radio control

Además de tener en cuenta cuál servo es pitch y cuál es roll esto se ajusta respecto a las partes del gimbal como se muestra en la Figura 84 y poder también cargar los parámetros necesarios en la placa controladora de vuelo Pixhawk 2.4.8 para no tener

errores y dificultades al momento de asignar las señales de RC de acuerdo con el canal asignado.



Figura 84. Partes del gimbal

De acuerdo con el esquema electrónico del gimbal a la placa Pixhawk, y además de las conexiones físicas que se realizaron con la placa y el gimbal se tomó las salidas auxiliares para señales PWM el número del puerto es el puerto 1 y el puerto 2, una vez se realizó las conexiones se procede a la configuración por el software Mission Planner.

Para aquello, se redirigió hacía Config > Full Parameter List es necesario buscar los parámetros de estas salidas de acuerdo a la página oficial de ardupilot los parámetros y sus valores se especifican en la Tabla 21, asimismo a estos valores se le asigna 63 que es RCIN13 y 64 que es RCIN14 que son los canales antes configurados en el radio control TX16S RadioMaster, esto específicamente para el control de los servo motores de “pitch” y “roll”, se actualizo aquellos valores como se visualiza en la Figura 85.

Tabla 21. Parámetros ajustar para el control de servo motores

Parámetros	**Valor**
SERVO10_FUNCTION	63 (RCIN13)
SERVO9_FUNCTION	64 (RCIN14)



Figura 85. Valores para configuración dentro de los parámetros SERVO10_FUNCTION y SERVO9_FUNCTION en Mission Planner

Se cargó los valores dentro del software primero se cambia en la parte de Value los valores antes especificados para después escribir los parámetros cambiados este botón se indica en la parte derecha del menú CONFIG en el botón “Write Params” como se visualiza en la Figura 86. Para después se requirió del reinicio del software y del dron, se quitó la alimentación y de nuevo se volvió alimentar el dron y se comprobó el movimiento correcto de los servos motores con el radio control con las perillas S1 Y S2 como se visualiza en la Figura 87.



Figura 86. Configuración de parámetros para el gimbal dentro del software Mission Planner



Figura 87. Perillas S1 y S2 en el radio control

e. Configuración de rutas autónomas para la planificación de misiones para el UAV en el software Mission Planner

La configuración de rutas autónomas es basada en comandos, específicamente comandos MAVLink los cuales vienen preestablecidos dentro del software Mission Planner, con el apoyo de ArduPilot estos mismos se pueden ejecutar como parte de

una misión, es decir en el modo de vuelo AUTO mencionando detalladamente que el dron va a tomar la configuración cargada de la ruta y de acuerdo con el sistema de la placa controladora de vuelo deberá cumplir la misión cargada. Entonces definiendo como misión a la función, encargo, o propósito que deberá cumplir una persona, pero en este caso se le encarga a una maquina la cuál es el UAV se detalló algunos pasos a seguir para iniciar una misión previamente establecida, esto se explica en el diagrama de flujo de la Figura 88.

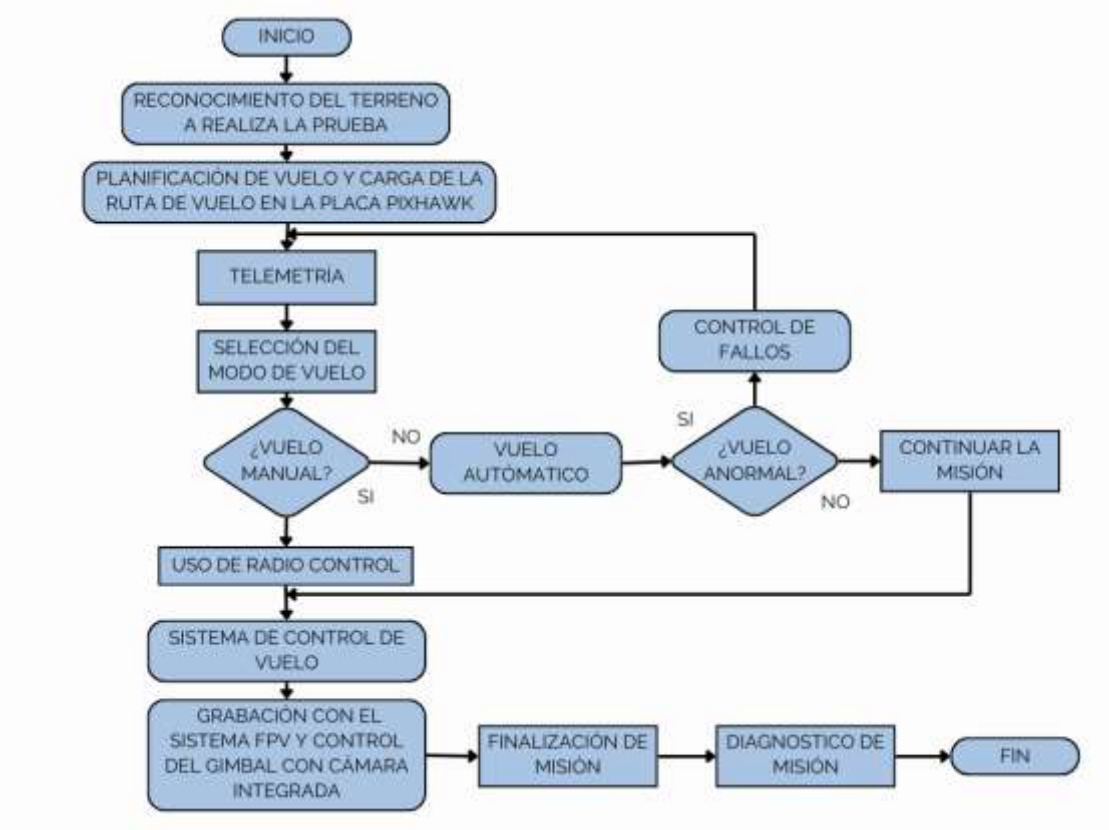


Figura 88. Diagrama de flujo de la misión de vuelo del hexacóptero

Acorde con uno de los recuadros del diagrama anterior que se especifica como “telemetría” entonces se explica el diagrama de flujo del funcionamiento del módulo de telemetría esto se visualizara en la Figura 89.

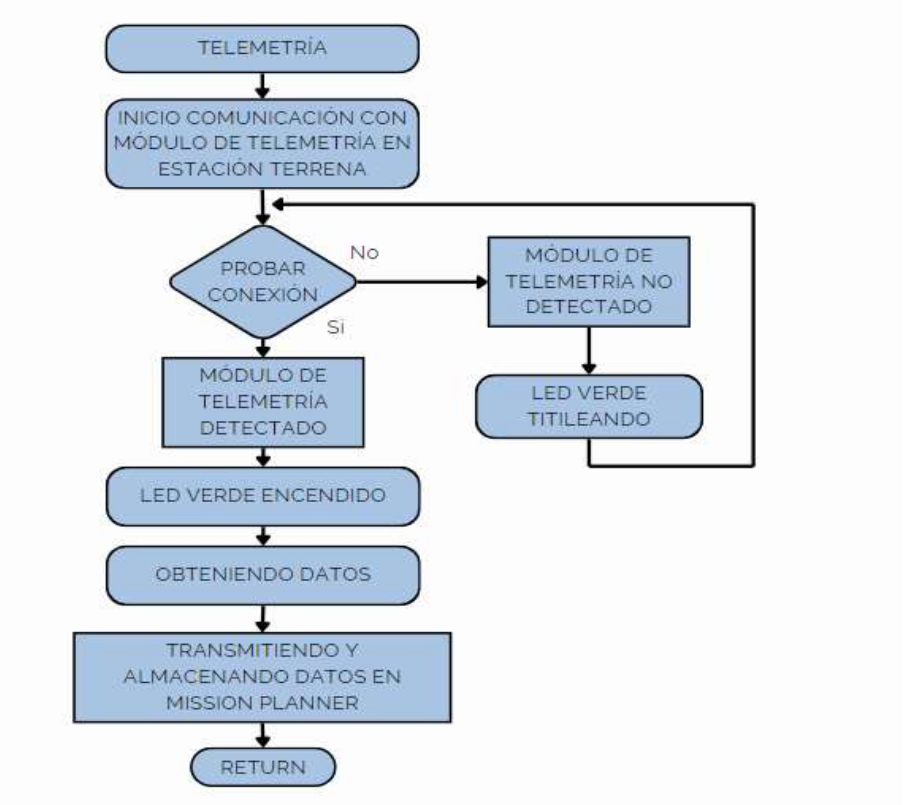


Figura 89. Diagrama de flujo de conexión y funcionamiento de telemetría

Una vez se especificó que se debió cargar comandos MAVLink que son compatibles con la placa controladora de Vuelo PixHawk, para cada uno de estos comandos es un comando de "navegación" o de "ejecución". Los comandos de navegación (así como "Takeoff" y "Waypoint") afectarán la posición del dron, mientras que los comandos "Do" (así como "Do-Set-Servo" y "Do-Cam-Trigg-Dist") se utilizan para funciones auxiliares y no afectan de forma directa a la ubicación actual del dron. Durante la ejecución de la tarea se pueden ejecutar como máximo un comando "Navegación" y un comando "No" al mismo tiempo. Después de ejecutar el comando de navegación anterior, los comandos "hacer" se ejecutarán secuencialmente. Para aquello se detalla algunos de los comandos de acuerdo con su funcionalidad y los que son más usados en la investigación estos se muestran en la Tabla 22. [61]

Tabla 22. Comandos para navegación autónoma en una misión [62]

Comando	Función
MAV_CMD_NAV_TAKEOFF	Despegue del dron en la ubicación actual.

Comando	Función
MAV_CMD_NAV_WAYPOINT	Navegación a una posición específica.
MAV_CMD_NAV_SPLINE_WAYPOINT	El dron volará a una ubicación de destino usando una ruta Spline, comandos Spline toman los mismos argumentos que son puntos de ruta regulares (lat, lon, alt, delay).
MAV_CMD_NAV_LOITER_TIME	El Dron vuela hacia el punto establecido y luego esperará en la ubicación especificada por el número de segundos especificado.
MAV_CMD_NAV_RETURN_TO_LAUNCH	Regrese a la ubicación de inicio o al punto de reunión más cercano, si está más cerca.
MAV_CMD_NAV_LAND	El Dron aterrizará en su ubicación actual o avanzará a la altitud actual hasta las coordenadas de latitud/longitudes proporcionadas (si son distintas de cero) y aterrizará.
MAV_CMD_NAV_DELAY	Al alcanzar un punto de referencia, retrasará la ejecución del siguiente comando tipo misión hasta que haya transcurrido un tiempo en específico.
MAV_CMD_CONDITION_DISTANCE	Retrasará el inicio del próximo comando “DO” hasta que el dron esté dentro de una distancia (m) del próximo punto de referencia.
MAV_CMD_MISSION_START	Utilizado para comenzar la misión una vez el dron esta armado, esto es utilizado por la estación terrena.
MAV_CMD_COMPONENT_ARM_DISARM	Armado o desarmado de motores
MAV_CMD_DO_CHANGE_SPEED	Cambiará la velocidad horizontal de destino o punto siguiente (en metros / seg) del dron.

Se detallará a continuación con una mejor explicación cada uno de los comandos mostrados en la Tabla 22, esto específicamente con los parámetros que se necesita configurar los comandos de acuerdo con el orden en que se ingresa y la utilidad de estos teniendo en cuenta la misión a cargar en el dron, además se indica una

herramienta del software Mission Planner nombrado como “PLAN” la cual tiene las configuraciones especificadas para la planeación de rutas asimismo se puede visualizar algunas de sus partes y herramientas que tiene el apartado de “PLAN” en la Figura 90.



Figura 90. Partes del apartado de planificación de misiones en el software Mission Planner

Comando de Despegue - TAKEOFF

Este comando realiza la acción para ascender al dron desde la posición actual hasta la altitud especificada (en metros). Este debería ser el primer comando para casi cualquier operación aérea o misión programada. Si la misión se inicia cuando el dron ya está volando, el dron ascenderá hasta la altitud especificada. Si el dron ya está por encima de la altitud especificada, el comando de inicio se ignorará y la misión pasa inmediatamente al siguiente comando de acuerdo con la misión programada los parámetros necesarios para este comando se muestran en la Tabla 23, está configuración se visualiza en la Figura 91 [62].

Tabla 23. Parámetros de comando TAKEOFF

Campo de commando	**Campo en el software**	**Descripción**
Param7	Alt	Altitud (m)



Figura 91. Comando TAKEOFF dentro del software Mission Planner

Comando de Waypoint

Este comando realiza la acción para que el dron vuele en línea recta a una ubicación especifica con los parámetros establecidos como latitud, longitud y la altitud (en metros) que se especifican en la Tabla 24, para el parámetro de **"Delay"** se establecerá cuánto tiempo de espera en un punto de referencia antes de pasar al siguiente comando (en segundos) la intención es mantener la distancia (en metros) desde el punto de referencia. Para ello, se utilizó el parámetro WPNAV_RADIUS. Incluso WPNAV_RADIUS solo tiene waypoints con un retardo en específico. Si no se especifica ningún retraso, el punto de ruta se considerará completo cuando el punto virtual que el dron está persiguiendo llegue al punto de ruta. Esta posición puede estar 10 metros (o más) por delante del dron, por lo que el vehículo girará hacia el siguiente punto de referencia antes de llegar al destino, los parámetros a configurar se visualizan en la Figura 92 [62].

El punto de referencia actual de guiñada "**Ang"** este no es compatible, su propósito es mantener el ángulo angular final en grados (0 = Norte, 90 = Este). Se utilizará la directiva CONDITION_YAW en su lugar. Para "**Lat", "Lon"** los objetivos de Medidas de ancho y largo. Si se establece en cero, conservará la posición actual y para

"**Alt"** la altura del objetivo sobre el objetivo o sitio en metros. Si se establece en cero, mantendrá la altura actual.

Tabla 24. Parámetros de comando WAYPOINT

Campo de commando	Campo en el software	Descripción
param1	Delay	Tiempo de espera en el punto de referencia de la misión en segundos enteros con valor MÁXIMO de 65535 segundos.
param5	Lat	Latitud objetivo, si es cero el dron se mantendrá en la latitud actual.
param6	Lon	Longitud objetivo, si es cero el dron se mantendrá en la longitud actual.
param7	Alt	Altitud objetivo, si es cero el dron se mantendrá en la altitud actual.

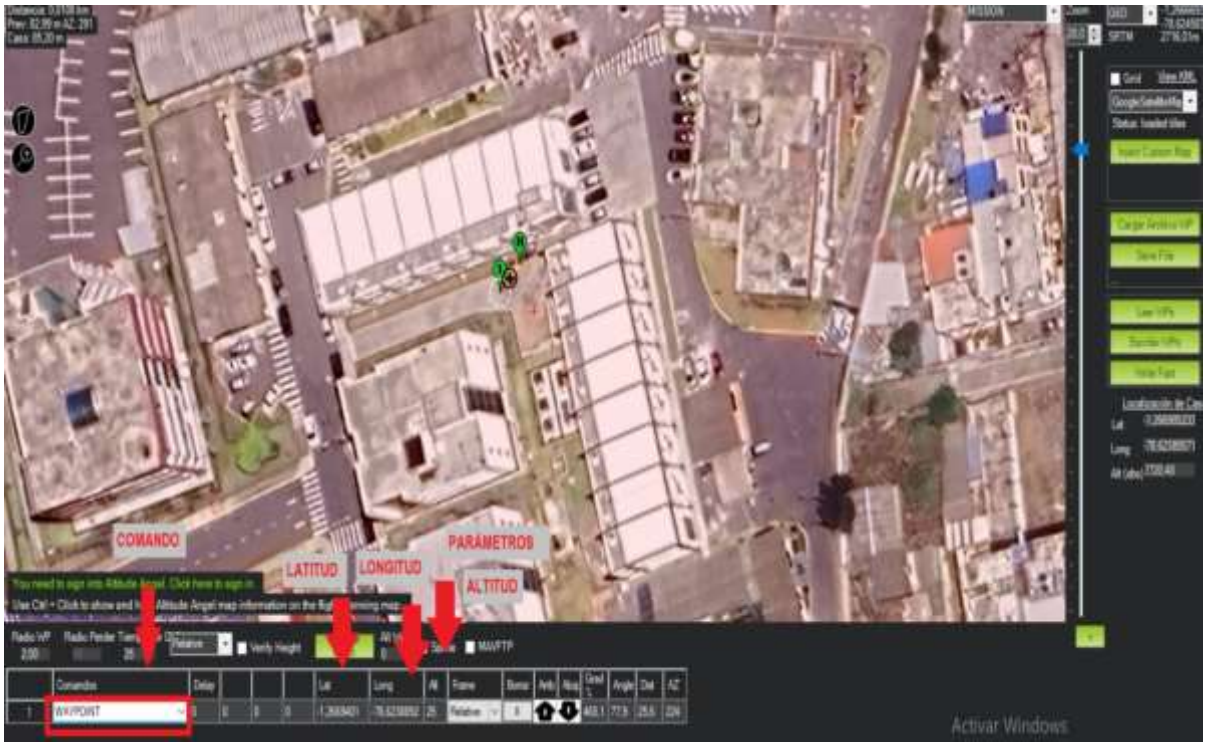


Figura 92. Comando WAYPOINT dentro del software Mission Planner

Comando de Spline Waypoint

Los comandos Spline toman todos los mismos argumentos este comando son waypoints regulares (lat, lon, alt, retardo) estos parámetros se indican en la Tabla 25, pero cuando se ejecuta el vehículo volará caminos lisos (tanto vertical como

horizontal) en lugar de líneas rectas. Waypoints Spline se pueden mezclar con los waypoints de líneas rectas regulares como se muestra en la Figura 93. [62]

Tabla 25. Parámetros de comando SPLINE WAYPOINT [62]

Campo de comando	Campo en el software	Descripción
param1	Delay	Hold time at target, in decimal seconds. Tiempo de retención en el objetivo, en segundos decimales.
param5	Lat	Latitude/X de la meta
param6	Long	Longitude/Y de la meta
param7	Alt	Altitude/Z de la meta



Figura 93. Comando SPLINE WAYPOINT dentro del software Mission Planner

Comando de Loiter Time

Este comando realiza la acción para volar/conducir hasta un lugar determinado y luego flotar durante una cierta cantidad de segundos, lo que significa "esperar en el lugar" (en lugar de "círculo"). El cronómetro comienza cuando se alcanza el waypoint, cuando se detiene, el waypoint se completa. Si el parámetro de latitud/longitud/altitud, se utilizará el valor de ubicación actual para este parámetro. Los parámetros necesarios

para la configuración se muestran en la Tabla 26, además se visualiza la configuración en el software en la Figura 94 [62]

Tabla 26. Parámetros de comando LOITER TIME [62]

Campo de comando	Campo en el software	Descripción
param1	Time (s)	Tiempo de permanencia en el punto de referencia (segundos - decimal)
param5	Lat	Latitud objetivo, si es cero el dron se mantendrá en la latitud actual.
param6	Lon	Longitud objetivo, si es cero el dron se mantendrá en la longitud actual.
param7	Alt	Altitud objetivo, si es cero el dron se mantendrá en la altitud actual.



Figura 94. Comando LOITER TIME dentro del software Mission Planner

Comando de Return to Launch

Este comando es equivalente al modo de vuelo RTL al activarlo en la misión. De acuerdo, al modo de vuelo RTL, el dron primero ascenderá a la altitud especificada por el parámetro RTL_ALT (15 m por defecto) antes de regresar a casa. La posición inicial es donde se montó el vehículo por última vez. Este comando no requiere

parámetros y normalmente debería ser el último comando de una tarea como se muestra en la Figura 95 [62].



Figura 95. Comando RETURN TO LAUCHN dentro del software Mission Planner

Comando de Land

Este comando realiza la acción para que el dron aterrice en su ubicación actual o en el lat / lon coordenadas proporcionadas. Este es la equivalente misión del modo de vuelo Land. Para los parámetros **Lat, Lon** que se muestra en la Tabla 27, los objetivos de latitud y longitud como se muestra en la Figura 94. Si se deja en cero el dron aterrizará en la ubicación actual. [62]

Tabla 27. Parámetros de comando LAND [62]

Campo de comando	Campo en el software	Descripción
param5	Lat	Latitud objetivo, si es cero el dron se mantendrá en la latitud actual.
param6	Lon	Longitud de la ubicación actual



Figura 96. Comando LAND dentro del software Mission Planner

Comando condición Delay

Retrasa el inicio del siguiente comando "No" durante el número de segundos especificado, los parámetros del comando se especifican en la Tabla 28. Por ejemplo, si se utiliza el comando Loiter time se retrasa para que no se inicie hasta 5 segundos después de que el dron haya pasado el punto de la ruta siguiente o el comando que este. Se deberá considerar que este comando no detiene el dron. También se tendrá en cuenta que el retraso de tiempo solo se aplica hasta que se alcanza el siguiente punto de ruta o waypoint, por lo que, si no se inicia el comando “No”, nunca se iniciará. Los valores de los parámetros de este comando se muestran en la Figura 97 [62].

Tabla 28. Parámetros de comando DELAY [62]

Campo de comando	Campo en el software	Descripción
param1	Tiempo (segundos)	Retraso en segundos (entero)
param2	Tiempo en horas (1-24)	Retraso en un rango en horas
param3	Tiempo en minutos (0-59)	Retraso en un rango en minutos
param4	Tiempo en segundos (0-59)	Retraso entre ese rango en segundos



Figura 97. Comando DELAY dentro del software Mission Planner

Comando Condition-Distance

Retrasa el inicio del próximo comando "_DO_" hasta que el dron esté dentro del número especificado de metros del próximo punto de referencia, la descripción de los parámetros del comando especificado se explica en la Tabla 29. Por ejemplo, si se utiliza el comando Loiter time se retrasa para que no se inicie hasta 5 segundos después de que el vehículo haya pasado el punto de la ruta siguiente o el comando que este. También se tendrá en cuenta que el retraso de tiempo solo se aplica hasta que se alcanza el siguiente punto de ruta o waypoint, por lo que, si no se inicia el comando "No", nunca se iniciará. Como se editan los parámetros se visualiza en la Figura 98 [62].

Tabla 29. Parámetros de comando CONDITION DISTANCE [62]

Campo de comando	**Campo en el software**	**Descripción**
param1	Dist (m)	Distancia desde el siguiente punto de referencia antes de que se ejecuten los comandos DO (metros).



Figura 98. Comando CONDITION DISTANCE dentro del software Mission Planner

Comando DO CHANGE SPEED

Este comando es de una acción "DO", esto específicamente para una característica de vuelo o acción en el vuelo para cierta misión los parámetros a ajustar se muestran en la Tabla 30, entonces se establece la velocidad máxima requerida en metros por segundo únicamente. Se ignorarán tanto los ajustes del tipo de velocidad como del acelerador del dron antes configurados en el transcurso después del punto waypoint establecido como se muestra en la Figura 99.

Tabla 30. Parámetros de comando DO CHANGE SPEED [62]

Campo de comando	Campo en el software	Descripción
param1	Type	Tipo de velocidad (0,1=Velocidad terrestre, 2=Velocidad de ascenso, 3=Velocidad de descenso)
param2	Velocidad en m/s	Velocidad de objetivo (m/s)



Figura 99. Comando DO CHANGE SPEED dentro del software Mission Planner

Para los siguientes comandos se utilizó específicamente para la interfaz creada en Python el cuál interactúa con el sistema de navegación autónoma del dron, esto como monitoreo y control del dron, como se mencionó son comandos MAVLink que es el protocolo de comunicación que se utiliza y este es compatible con la programación en Python utilizando las librerias *pymavlink*, entonces se especifica algunos comandos necesarios para la interfaz para el control en el caso de una misión.

Comando MAV_CMD_MISSION_START

Este comando se puede utilizar para iniciar una misión cuando el dron está en modo automático en tierra. Si el dron ya está en el aire, la misión comenzará inmediatamente después de cambiar al modo automático (por lo que este comando no es necesario/ignorado). Esto permitirá que el GCS/ordenador auxiliar que se encuentra en la estación terrena lance misiones autónomas sin aumentar el acelerador.

En el software Mission Planner en el apartado de “DATA”, en la venta de parámetros y una lista de acciones se podrá mandar esta acción en MISSION_START, tener una consideración en cuanto tiene que armarse primero el dron para iniciar con la misión y luego se presiona en “hacer acción” como se visualiza en la Figura 100.



Figura 100. Realizar una acción en el apartado de DATA, para Mission_Start en el software Mission Planner

Comando MAV_CMD_COMPONENT_ARM_DISARM

Este comando es utilizado específicamente en tierra, de forma que se ubicó el dron en donde empezara la misión para aquello el desarmado desactiva los motores y el armado activa los motores además de poder realizarlo en vuelo (lo cual no es recomendado para no tener incidentes con la estructura del UAV).

En el software Misión planner en el apartado de "DATA", en la venta de parámetros y una lista de acciones se podrá mandar esta acción en Arm/Disarm, como se visualiza en la Figura 101.



Figura 101. Realizar una acción en el apartado de DATA, para Arm/Disarm en el software Mission Planner

f. Carga e inicio de misión con el software Mission Planner

Una vez configurado los comandos para la misión y preestableciendo una ruta en específico para la operación aérea necesaria para el dron como siguiente paso fue necesario escribir esta configuración previa en el dron, con un paso previo se necesitó leer los datos en el caso de tener una misión ya cargada en el dron esto para verificar también la localización del punto home en el caso de no tener ubicado bien el punto inicial que es de suma importancia, estos botones se visualizan en el apartado de PLAN como se indica en la Figura 102.



Figura 102. Cargar y Leer misión en el apartado de PLAN en Mission Planner

Antes de iniciar la misión, se debió asegurar que la palanca del acelerador del control remoto esté en su posición mínima. Posteriormente, se activa el modo de vuelo automático ("AUTO"), y solo después se incrementa el acelerador. Una vez que el acelerador supera el nivel cero, el dron inicia automáticamente la misión programada.

Si el dron ya está en vuelo y se activa el modo "AUTO", la misión comienza desde el primer comando de la secuencia preprogramada al cambiar al modo automático. Si el primer comando corresponde a un despegue, pero el dron ya se encuentra a una altitud superior a la indicada en dicho comando, el sistema lo marca como completado y procede al siguiente waypoint de la misión.

En cualquier momento, el piloto puede tomar el control manual del dron cambiando el modo de vuelo a una modalidad alternativa, como "Stabilize" o "Loiter". Si se retorna al modo "AUTO", la misión se reinicia desde el primer comando, independientemente de la posición actual del dron.

Durante la ejecución de la misión, las entradas de balanceo (roll), cabeceo (pitch) y aceleración (throttle) realizadas desde el control remoto son ignoradas por el piloto automático. Sin embargo, el control de guiñada (yaw) puede ser sobrescrito manualmente, permitiendo al piloto orientar el dron, por ejemplo, para apuntar el sistema del gimbal con la cámara montada. El sistema de piloto automático retomará el control de la guiñada automáticamente al alcanzar el siguiente waypoint.

Finalización de una misión

Es necesario que las misiones incluyan un comando final, como RTL para garantizar el retorno automático del dron al punto de inicio (donde fue armado), o LAND para aterrizar en una ubicación específica. Si no se establece un comando final, el dron se detendrá en el último waypoint, y el piloto deberá asumir el control manualmente. Al aterrizar, el piloto deberá desarmar los motores del dron una vez aterrizado.

3.1.10 Análisis del sistema de navegación autónoma del UAV

a. Cinemática del UAV

La cinemática de un UAV se desarrolla mediante un vector que muestra la posición y la orientación del dispositivo respecto a un sistema de referencia fijo en este caso será representado como $\{F\}$. El punto de control para la tarea está dado por la posición del centro de masa del UAV $\xi p \in \Re^3$ y $\xi\psi \in \Re$ como orientación, de tal manera que:

$$\xi(t) = \begin{bmatrix} \xi_p\ (t) \\ \xi_\psi(t) \end{bmatrix} = \begin{bmatrix} \xi_p\ (t) \\ \xi_\psi(t) \\ \xi_p\ (t) \\ \xi_\psi(t) \end{bmatrix} \quad (1)$$

Donde:

$\xi_p\ (t)$ = posición espacial en el sistema {F}.

$\xi_\psi(t)$ = orientación angular (yaw) en el sistema {F}.

$\xi(t)$= estado completo del UAV, incluyendo posición y orientación.

La velocidad en el punto de operación definido se determina mediante la derivada temporal de la Ecuación 1, de manera que $\dot{\xi}(t) = \frac{d}{dt}\xi(t)$, lo cual resulta en:

$$\dot{\xi}(t) = \left[\dot{\xi}_p^T \quad \dot{\xi}_\psi\right]^T = \left[\dot{\xi}_x\ \dot{\xi}_y\ \dot{\xi}_z\ \dot{\xi}_\psi\right]^T \quad (2)$$

$R(\phi, \theta, \psi) = R(x, \phi)R(y, \theta)R(z, \psi)$. La Figura 103 muestra la configuración cinemática del UAV en el sistema de referencia fijo {F}.

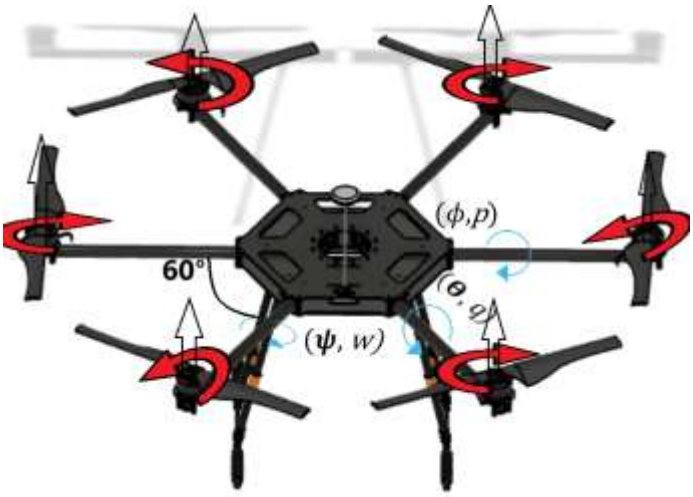


Figura 103. Configuración cinemática del UAV en el espacio [63].

b. Consumo Energético completo del UAV

El consumo eléctrico de los componentes del dron es un factor clave en el diseño y planificación de su sistema de alimentación. En particular, los motores brushless, que son responsables de la propulsión del dron, tienen un consumo significativo. Con una tensión de 22.2 V y una corriente de 5 A por motor, los seis motores en conjunto demandan una potencia total de 111 W, lo que se traduce en un consumo diario de aproximadamente 222 Wh. Esta potencia es esencial para asegurar la estabilidad y maniobrabilidad del dron durante sus vuelos. La cámara GoPro, con una potencia de 7.5 W, requiere de 15 Wh diarios. Estos dispositivos son fundamentales para la captura de imágenes y la medición de distancias, respectivamente, pero no representan un consumo tan elevado como los motores.

Por otro lado, el controlador de vuelo Pixhawk, el GPS y la telemetría consumen una cantidad relativamente baja de energía, lo que es crucial para la navegación y la comunicación del dron. El Pixhawk, con un consumo de 1.5 W, demanda 3 Wh diarios, mientras que el GPS y el sistema de telemetría, con 1.5 W y 0.5 W respectivamente, suman 3 Wh y 1 Wh al consumo total. Además, el receptor y los componentes electrónicos adicionales también requieren energía para su funcionamiento. El receptor, con un consumo de 5 W, contribuye con 10 Wh, mientras que los componentes electrónicos adicionales, con una potencia de 2.5 W, requieren 5 Wh por día. El consumo total de corriente de todos estos componentes está distribuido entre dos fuentes de alimentación (LiPo de 6 Celdas a 6000 mA y Lipo de 3 celdas a 1200 mA), y la suma de su carga diaria es crucial para determinar la capacidad de la batería necesaria para un vuelo eficiente y seguro.

Como se puede visualizar en la Tabla 31 se detalla el consumo energético del gimbal utilizado para la estabilización de la cámara GoPro. Este gimbal está alimentado por una batería independiente de 1.2 A, con un voltaje de 11,1 V compuesta por tres celdas, lo que garantiza su funcionamiento autónomo respecto al sistema principal del dron. Por otro lado, en la Tabla 32 se presenta el análisis del consumo energético total del sistema, considerando el peso y todos los componentes integrados en el dron, con excepción del gimbal.

Tabla 31. Consumo energético del gimbal del UAV

Descripción	**Tensión (V)**	**Corriente (A)**	**Potencia (W)**	**Consumo/Dia (Wh/dia)**	**Carga/Dia (Ah/dia)**
Gimbal	5 V	2 A	10 W	20 Wh/día	4.0 Ah/dia
Cámara GoPro	5 V	1.2 A	7.5 W	15 Wh/día	3.0 Ah/dia
			TOTAL	35 Wh/día	

Tabla 32. Consumo energético del sistema completo del UAV.

Descripción	**Tensión (V)**	**Corriente (A)**	**Potencia (W)**	**Consumo/Dia (Wh/dia)**	**Carga/Dia (Ah/dia)**
Motor brushless	22.2 V	5 A	111 W	222 Wh/día	40.0 Ah/día
Pixhawk	5 V	0.3 A	1.5 W	3.0 Wh/día	0.6 Ah/día
GPS	3.3 V	0.3 A	1.5 W	3 Wh/día	0.6 Ah/día
Telemetría	3.3 V	0.152 A	0.5 W	1 Wh/día	0.3 Ah/día
Receptor	5 V	1 A	5 W	10 Wh/día	2 Ah/día
Componentes Electrónicos	5 V	0.5 A	2.5 W	5 Wh/día	1 Ah/día
			TOTAL	244 Wh/dia	

c. Peso del UAV

El peso entre todos los componentes del UAV fue esencial al momento de que se realizó pruebas de vuelo esto tanto para el gasto energético de todo el sistema, como la potencia necesaria para un despegue seguro del UAV entonces para aquello se realizó respecto a parámetros técnicos de cada elemento antes mencionado en las tablas comparativas se visualiza el peso en gramos de cada componente del sistema en la Tabla 33 para tener el peso total del UAV.

Tabla 33. Distribución de Pesos de los Componentes del Dron Hexacóptero.

COMPONENTE	**PESO (gr)**
Estructura	800 gr
Batería LiPo de 6 celdas (6S)	700 gr
Batería LiPo de 3 celdas (3S)	300 gr
Motores (6 unidades)	500 gr
Hélices (6 unidades)	100 gr
Gimbal	400 gr
Pixhawk	200 gr
Cables y Conectores	100 gr
Cámara GoPro	117 gr
Caja de baterías	80 gr
TOTAL	3297 gr

Para la validación del peso correcto del dron se realizó la medida del peso con una balanza analógica de forma que el valor que se obtiene un valor de peso total en la balanza de 3300 gramos como se indica en la Figura 104.



Figura 104. Peso del dron en la balanza analógica

d. Autonomía de vuelo con relación del peso del UAV

La autonomía de funcionamiento de un UAV está directamente determinada por el peso total del dispositivo, ya que este influye en la cantidad de energía necesaria para cada vuelo. El peso total establece el tiempo que el dron podrá mantenerse en vuelo para cumplir con la misión asignada, a mayor peso, menor será el tiempo de vuelo. Cada componente debe ser optimizado para consumir solo la energía necesaria, logrando un balance entre potencia y eficiencia. Es fundamental que los sistemas de alimentación sean capaces de proporcionar la energía requerida para que los componentes operen con una mayor eficiencia energética, lo que, a su vez, garantiza un tiempo de vuelo adecuado para completar las trayectorias planificadas.

Cálculo del consumo total energético.

El consumo energético se calcula considerando el consumo de motores, y los diferentes componentes implementados en el UAV.

$$P_{total} = P_{motores} + P_{otros} \quad (3)$$

P_{total} = Potencia total.

$P_{motores}$ = Potencia consumida por los motores.

$$P_{motores} = 60\ W\ X\ 6 = 360\ W$$

P_{otros} = Potencia consumida por otros componentes electrónicos del UAV.

$$P_{total} = 360\ W + 54\ W = 414\ W$$

Capacidad energética de la batería.

Para el cálculo de la capacidad energética se determina en miliamperios/hora (Wh).

$$E_{bateria} = \frac{Capacidad\ (mAh)\ x\ Voltaje\ (V)}{1000} \qquad (4)$$

Donde:

$E_{bateria}$ = Energía que proporciona la batería.

$E_{bateria}$ = Capacidad de la batería en mA.

$Voltaje$ = Voltaje total de la batería.

Para el cálculo en nuestro sistema:

$$E_{bateria} = \frac{6000\ (mAh)\ x\ 22.2\ (V)}{1000} = 133.2\ Wh$$

Tiempo de funcionamiento del dispositivo (vuelo del UAV)

Para poder determinar el tiempo de vuelo que tiene el dispositivo se usa la formula:

$$T_{vuelo} = \frac{E_{bateria}}{P_{Total}} \qquad (5)$$

Donde:

T_{vuelo} = Tiempo total de vuelo.

$$T_{vuelo} = \frac{133.2}{414} = 0.3069\ horas\ \approx 19.30\ minutos$$

e. Condiciones y parámetros para pruebas de vuelo del UAV

En base a la implementación de la tecnología GPS para el sistema de georreferencia se consideró ciertos datos y valores que la interfaz de monitoreo provee para evitar la colisión del dron previo a que se realizó pruebas, se empezó por visualizar el número de satélites que recepta el módulo GPS entre un valor mínimo de 8 satélites y un valor máximo de 14 satélites, otro dato fue gps HDOP nos indica la precisión y el nivel de recepción de satélites que capta la antena que viene incorporada en el módulo el valor menor a 1,25 es apto para realizar vuelos, en el caso de tener un valor mayor a 1,25 no es apto para realizar pruebas de vuelo, hubo que revisar el nivel de la batería mediante el sensor integrado en el dron, se puede llegar a un nivel mínimo de voltaje de 21,9 V, para realizar pruebas y que no exista problemas de energía con el sistema.

En caso de eventos inoportunos en las pruebas de vuelo, como método de emergencia se debió tener el manejo desde el radio control para estabilizar el dron en el caso de tener errores al a ver realizado una misión, además de tener configurado previamente el modo RTL como caso de emergencia en tener problemas de recepción de satélites, de forma que en la planeación de rutas o misiones es esencial establecer el punto HOME de manera correcta verificando datos de latitud y longitud.

3.1.11 Simulación de trayectorias del UAV en el software MATLAB

En el entorno de desarrollo de Matlab, se elabora un script que permite importar el modelo 3D del UAV, previamente centrado y guardado en formato STL. Este proceso es fundamental para llevar a cabo simulaciones que ilustran tanto el posicionamiento como la trayectoria del vehículo aéreo no tripulado, facilitando así su análisis y evaluación en entornos controlados.

Como puede visualizar en la Figura 105, se realiza la preparación inicial de la simulación.

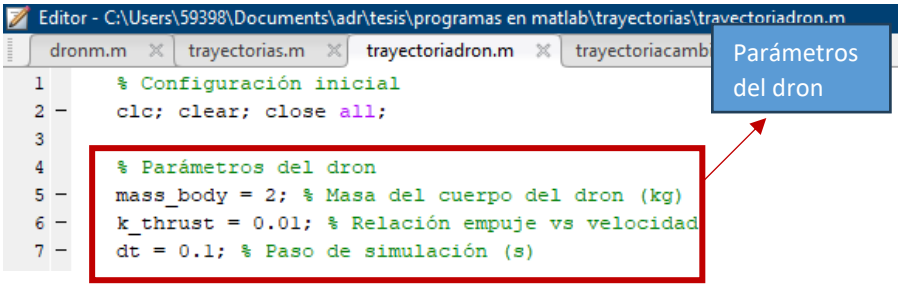


Figura 105. Parámetros del UAV.

En la siguiente parte del código, se define la trayectoria que el dron debe seguir, especificando una serie de puntos en 3D representados por las coordenadas (x, y, z). Estos puntos corresponden a ubicaciones espaciales que el dron debe alcanzar durante la simulación, representando los destinos establecidos a lo largo de su recorrido. Como se puede observar en la Figura 106, que ilustra la programación de esta sección, se detalla cómo se establecen las posiciones objetivo en el espacio tridimensional del dron y se codifican los parámetros necesarios para su ejecución.

Además, se configura un umbral de proximidad (proximity_threshold) de 0.05 m, lo que significa que el dron se considera como llegado a un punto de la trayectoria cuando la distancia entre su posición actual y el objetivo es igual o menor a 0.05 m. Este parámetro, programado en la lógica que se describe en la Figura 106, es clave para determinar cuándo el dron ha alcanzado un punto específico de su ruta, permitiendo que el proceso avance al siguiente destino o ejecute la acción correspondiente según lo previsto en el sistema de control.

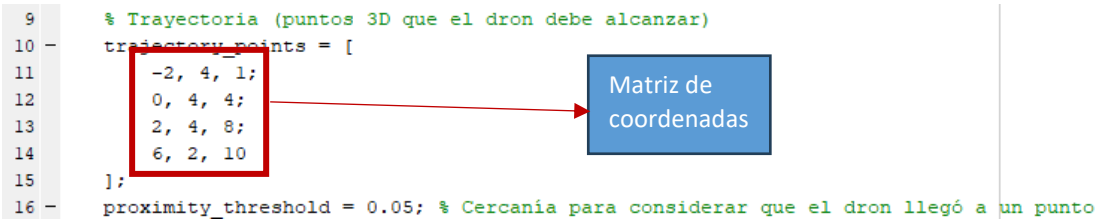


Figura 106. Ruta configurada por una matriz.

Como se ilustra en la Figura 107, se inicializan las variables necesarias para definir el estado inicial del dron al comienzo de la simulación. En este punto, se establece que la posición inicial del dron coincide con el origen del sistema de coordenadas, es decir, $x = 0, y = 0, z = 0$. Además, la velocidad inicial del dron se define como cero en las tres direcciones, lo que indica que se encuentra en reposo al inicio de la simulación.

De manera similar, la aceleración inicial se establece como cero, lo que implica que no se ejerce ninguna fuerza externa sobre el dron al inicio del proceso.

Adicionalmente, se crean tres listas vacías denominadas trajectory_x, trajectory_y y trajectory_z, que se utilizarán para registrar los valores de las coordenadas x, y y z a medida que el dron se desplaza a lo largo de su trayectoria. Como se evidencia en la programación detallada en la Figura 107, estas listas se actualizarán dinámicamente durante la simulación para almacenar y representar el recorrido del dron, proporcionando un seguimiento completo de su desplazamiento en el espacio tridimensional.

```
18      % Inicialización del estado del dron
19 -    position = [0, 0, 0]; % Posición inicial (x, y, z)
20 -    velocity = [0, 0, 0]; % Velocidad inicial
21 -    acceleration = [0, 0, 0]; % Aceleración inicial
22
23      % Registro de trayectoria
24 -    trajectory_x = [];
25 -    trajectory_y = [];
26 -    trajectory_z = [];
```

Figura 107. Posición inicial del dron.

En la siguiente sección del código, se realiza la carga del archivo STL que contiene el modelo tridimensional del dron, utilizando la función *stlread*. Esta función permite leer el archivo 'Assembly_Complete v3.stl', que incluye la información geométrica del modelo 3D del dron. Posteriormente, se ajusta el tamaño del modelo aplicando un factor de escala (scale_factor=0.01scale_factor = 0.01scale_factor=0.01), que reduce proporcionalmente las dimensiones del modelo para que se adapte adecuadamente a la simulación.

Como se puede observar en la Figura 108, los puntos que forman la malla del modelo 3D se multiplican por este factor de escala, generando una versión más compacta del modelo. Esta versión escalada se almacena en la variable *scaled_points* y se utiliza posteriormente para la visualización del dron dentro del entorno simulado, permitiendo una representación gráfica acorde con los requisitos de la simulación.

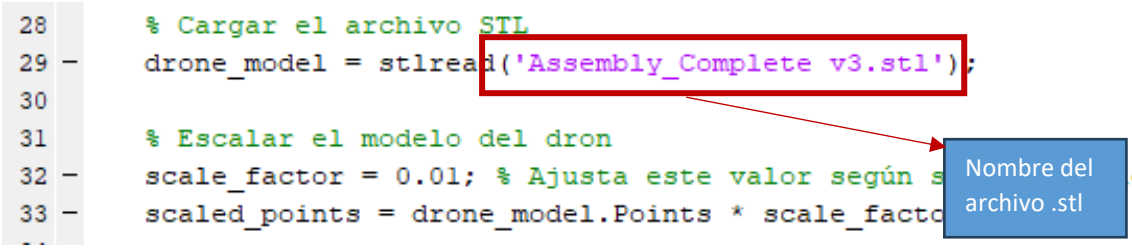


Figura 108. Cargar el archivo de diseño

Para la sección que se puede observar en la Figura 109, se configura y personaliza la visualización gráfica de la simulación. Primero, se crea una nueva figura “Figure (1)”, asegurando que se tenga un espacio de trabajo limpio. Luego, se activa el modo hold on para que se puedan agregar múltiples elementos gráficos a la misma figura sin que se borren los anteriores. Se activa la cuadrícula en el gráfico con grid on, lo que ayuda a ver mejor la orientación y el movimiento del dron. A continuación, se etiquetan los ejes X, Y y Z con las unidades correspondientes mediante xlabel, ylabel, zlabel. La vista se establece en 3D con “view(3)”, permitiendo observar el modelo desde diferentes ángulos.

Se dibujan los puntos de la trayectoria del dron utilizando scatter3, donde se muestran los puntos en el espacio 3D con las coordenadas correspondientes y se les asigna el color azul. Luego, se crea una línea que irá mostrando la trayectoria acumulada del dron a medida que avanza, usando plot3 para dibujar la trayectoria en color verde. Se llama a la función draw_drone para dibujar el modelo 3D del dron en la posición inicial, utilizando el modelo cargado anteriormente. Finalmente, se agrega una leyenda para identificar cada elemento gráfico mediante legend, lo que facilita la interpretación de la simulación

```
Configuración de la gráfica
gure(1);
ld on;
id on;
abel('X (m)');
abel('Y (m)');
abel('Z (m)');
ew(3);
atter3(trajectory_points(:,1), trajectory_points(:,2), trajectory_points(:,3), 'filled', 'b'); %
aj_line_full = plot3(0, 0, 0, 'g', 'LineWidth', 2); % Línea de trayectoria acumulada
one_plot_full = draw_drone(gca, position, drone_model, scaled_points); % Dibujo inicial del dron
gend('Trayectoria deseada', 'Trayectoria real', 'Dron');
```

Figura 109. Dibujar los puntos de trayectorias.

Para la siguiente sección, se ajustan automáticamente los límites del plano 3D en la gráfica para garantizar que toda la trayectoria del dron sea visible durante la simulación. Inicialmente, se calculan los valores mínimos y máximos de las coordenadas X, Y y Z de los puntos que conforman la trayectoria (trajectory_points), añadiendo márgenes alrededor de estos valores. Este margen asegura que tanto el dron como su trayectoria queden completamente visibles dentro del espacio representado.

Como se detalla en la Figura 110, se emplean las funciones xlim, ylim y zlim para establecer los límites de los ejes X, Y y Z en la gráfica tridimensional. Este proceso garantiza una visualización adecuada y centrada del dron y su trayectoria, permitiendo observar de manera clara el comportamiento del sistema en el entorno simulado.

```
48      % Ajustar automáticamente los límites del plano 3D
49 -    x_min = min([0; trajectory_points(:,1)]) - 10;
50 -    x_max = max([0; trajectory_points(:,1)]) + 10;
51 -    y_min = min([0; trajectory_points(:,2)]) - 10;
52 -    y_max = max([0; trajectory_points(:,2)]) + 10;
53 -    z_min = min([0; trajectory_points(:,3)]) - 5; % Menor margen para Z
54 -    z_max = max([0; trajectory_points(:,3)]) + 10;
55
56 -    xlim([x_min, x_max]);
57 -    ylim([y_min, y_max]);
58 -    zlim([z_min, z_max]);
```

Figura 110. Ajuste de Limites en el plano 3D

En esta parte del código, se realiza la simulación del movimiento del dron siguiendo una trayectoria definida. Para cada punto de la trayectoria (trajectory_points), el código realiza una iteración para mover el dron hacia ese punto.

a. Determinación del objetivo

El código establece que cada punto de la trayectoria representa un objetivo al que el dron debe dirigirse secuencialmente. Para ello, se evalúa constantemente la diferencia entre la posición actual del dron y la ubicación del siguiente objetivo. Este proceso es fundamental en la navegación de vehículos autónomos, ya que permite realizar ajustes dinámicos en el movimiento y la velocidad del dron para garantizar que siga la trayectoria planificada con precisión.

b. Controlador PD

Usando un controlador Proporcional-Derivativo (PD), calcula la aceleración necesaria para mover al dron hacia el objetivo, el cálculo de la diferencia entre la posición actual y el objetivo se realiza utilizando métricas como la distancia euclidiana, que mide la separación entre los dos puntos en un espacio tridimensional (X, Y, Z). Este cálculo permite determinar tanto la dirección en la que el dron debe moverse como la magnitud de su desplazamiento.

El código también define un umbral de proximidad para considerar que el dron ha alcanzado el objetivo. Este umbral evita que el dron realice movimientos innecesarios cuando está lo suficientemente cerca del punto deseado. Una vez que el dron está dentro de este rango, se selecciona el siguiente punto de la trayectoria como el nuevo objetivo, y el proceso se repite hasta completar toda la trayectoria.

c. Actualización de la velocidad y la posición

La actualización de la velocidad y la posición es un componente clave del modelo de simulación, ya que traduce las decisiones del controlador en un movimiento realista y permite al dron alcanzar sus objetivos de manera eficiente. Este proceso es esencial para simular con precisión el movimiento del dron y garantizar que siga la trayectoria definida. Se basa en principios fundamentales de la física del movimiento, donde la velocidad y la posición cambian con el tiempo en función de la aceleración aplicada.

d. Registro de trayectoria

El registro de trayectoria se encarga de almacenar las posiciones (X, Y, Z) del dron en cada paso del proceso de simulación. Este registro no solo permite visualizar el recorrido completo del dron en un espacio tridimensional, sino que también es una herramienta clave para analizar su desempeño. Al representar gráficamente la trayectoria seguida, es posible identificar desviaciones, verificar si el dron alcanza correctamente los objetivos definidos y evaluar su capacidad para mantenerse dentro de los parámetros de la ruta establecida. Además, este registro facilita la optimización del sistema de control, permitiendo realizar ajustes en caso de errores o inconsistencias en el movimiento.

e. Actualización de la gráfica

La actualización de la gráfica permite representar en tiempo real la trayectoria completa del dron junto con su posición actual en un espacio tridimensional. Este proceso es fundamental para visualizar de manera clara y dinámica cómo el dron sigue la trayectoria definida y cómo se desplaza en cada instante. Al mostrar la información en 3D, es posible identificar de forma precisa la posición del dron en los ejes X, Y y Z, lo que facilita el análisis de su movimiento y su capacidad para alcanzar los objetivos establecidos. Además, esta representación gráfica es una herramienta invaluable para depurar y optimizar el sistema de control, ya que permite detectar posibles errores en el seguimiento de la trayectoria y realizar los ajustes necesarios para mejorar el rendimiento del dron.

f. Control de la velocidad de simulación

El control de la velocidad de simulación es un elemento clave para garantizar que el movimiento del dron se represente de manera realista y comprensible. Al introducir una pequeña pausa en cada iteración del bucle de simulación, se logra ajustar la velocidad de ejecución del programa, evitando que los cambios en la posición y trayectoria del dron se muestren de forma abrupta o acelerada. Esto no solo mejora la experiencia visual de la simulación, sino que también facilita el análisis detallado del comportamiento del dron en cada etapa de su movimiento. Al mantener un ritmo adecuado, los usuarios pueden observar con mayor claridad cómo el dron responde al controlador y sigue la trayectoria definida, lo que resulta especialmente útil para depuración, evaluación de resultados y ajustes en el diseño del sistema.

Este proceso se repite hasta que el dron llega a cada punto de la trayectoria, como podemos observar en la Figura 111, asegurando que se mueva suavemente y con precisión hacia su destino.

```
60      % Simulación
61 -    for point = 1:size(trajectory_points, 1)
62 -        target = trajectory_points(point, :); % Siguiente punto de la trayectoria
63
64 -        while norm(position - target) > proximity_threshold
65              % Calcular la aceleración necesaria (controlador PD)
66 -            direction = target - position;
67 -            kp = 3; % Ganancia proporcional ajustada para mayor precisión
68 -            kd = 1; % Ganancia derivativa ajustada
69 -            force = kp * direction - kd * velocity;
70 -            acceleration = force / mass_body;
71
72              % Actualizar velocidad y posición
73 -            velocity = velocity + acceleration * dt;
74 -            max_velocity = 1; % Limitar la velocidad máxima para mayor precisión
75 -            if norm(velocity) > max_velocity
76 -                velocity = velocity / norm(velocity) * max_velocity;
77 -            end
78 -            position = position + velocity * dt;
79
80              % Registrar trayectoria
81 -            trajectory_x = [trajectory_x, position(1)];
82 -            trajectory_y = [trajectory_y, position(2)];
83 -            trajectory_z = [trajectory_z, position(3)];
84
85              % Actualizar la gráfica 1 (Trayectoria completa)
```

Figura 111. Configuración para simulación.

En la siguiente sección podemos observar la función draw_drone tiene como propósito dibujar el modelo 3D del dron en el espacio gráfico, como se muestra en la Figura 112. Utiliza la función patch para crear una representación visual del dron, asignándole colores a las caras y configurando la malla del modelo con los puntos escalados. Además, ajusta la posición inicial del dron en la gráfica mediante la llamada a la función update_drone, que actualiza su ubicación en función de las coordenadas actuales. Esta función permite visualizar el dron moviéndose a lo largo de su trayectoria definida en la simulación 3D.

```
98      % Función para dibujar el dron
99      function drone_plot = draw_drone(ax, position, drone_model, scaled_points)
100         % Dibujar el modelo 3D del dron
101 -       drone_plot = patch(ax, 'Faces', drone_model.ConnectivityList, 'Vertices', scaled_points, ...
102                           'FaceColor', 'red', 'EdgeColor', 'none');
103         % Ajustar la posición inicial
104 -       update_drone(drone_plot, position, scaled_points);
105 -   end
```

Figura 112. Visualización del dron en el espacio.

La función *update_drone*, como se muestra en la Figura 113, se encarga de mover el modelo 3D del dron en la simulación, garantizando que su visualización gráfica coincida con su posición en tiempo real. Para ello, ajusta la posición del modelo al sumar los datos de ubicación actuales a los puntos preescalados del dron y actualiza los vértices del modelo para reflejar la nueva posición. Este proceso es crucial para mantener la sincronización entre los datos simulados y la representación gráfica, asegurando que la trayectoria y las posiciones calculadas en la simulación se reflejen con precisión en el entorno visual.

```
107     % Función para actualizar el dron
108     function update_drone(drone_plot, position, scaled_points
109         % Actualizar la posición del modelo 3D
110 -       new_vertices = scaled_points + position;
111 -       set(drone_plot, 'Vertices', new_vertices);
112 -   end
```

Figura 113. Ajuste de la posición del dron en tiempo real

3.1.12 Análisis de Resultados de la simulación de las trayectorias del UAV en MATLAB

El código simula el comportamiento de un dron autónomo, evaluando su habilidad para seguir una trayectoria compuesta por puntos definidos en un espacio tridimensional. La simulación inicia con el dron en una posición inicial (0, 0, 0) y utiliza un controlador proporcional-derivativo (PD) para dirigirlo hacia los puntos objetivo. Durante el proceso, los parámetros de posición, velocidad y aceleración se actualizan dinámicamente en función de la dirección hacia el objetivo y las fuerzas aplicadas. Además, la simulación restringe la velocidad máxima del dron para mantener la precisión y estabilidad del sistema. Los datos de la trayectoria recorrida son registrados y visualizados en tiempo real mediante un modelo tridimensional, lo que permite analizar el desempeño del dron en cuanto a control y estabilidad mientras ajusta sus movimientos para cumplir con las restricciones establecidas y alcanzar los objetivos planteados como se puede visualizar en la Figura 114.

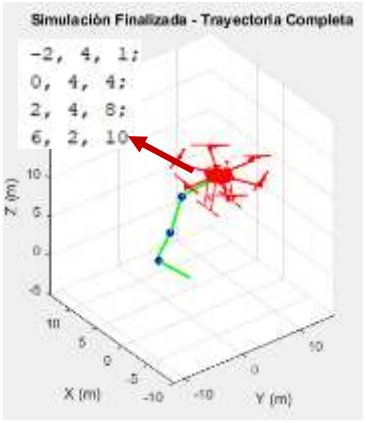


Figura 114. Trayectoria del UAV.

a. Posibles trayectorias simuladas en el software Matlab del UAV

Se ejecuta el programa según distintos puntos de destino ingresados en la matriz de trayectorias. Como se puede observar en la Figura 115, muestra los puntos deseados para generar una trayectoria, la cual manifiesta una ruta con una elevación de 45° la cual culmina en el punto final ingresado en la matriz ingresada.

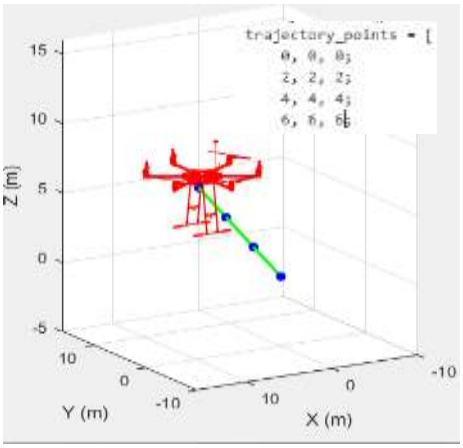


Figura 115. Trayectoria del UAV a 45°

Así mismo para diferentes rutas que se pueda realizar tenemos unas trayectorias en línea recta, como podemos observar en la Figura 116, la cual mantiene una altura estable sin cambios ni perturbaciones llegando con éxito desde su punto de origen hasta su punto de destino.

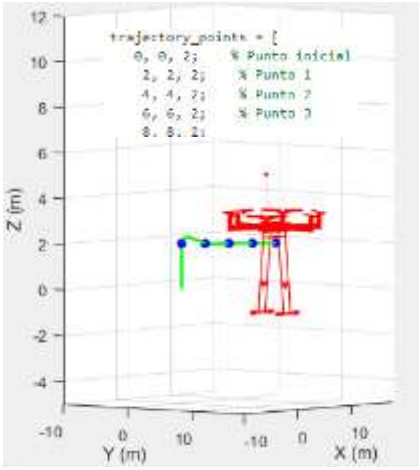


Figura 116. Trayectoria del UAV con altura constante

En la Figura 117, podemos observar que un cambio deseado en nuestra trayectoria la cual forma una circunferencia manteniendo en todo momento su altura sin presentar cambios bruscos de alguna índole.

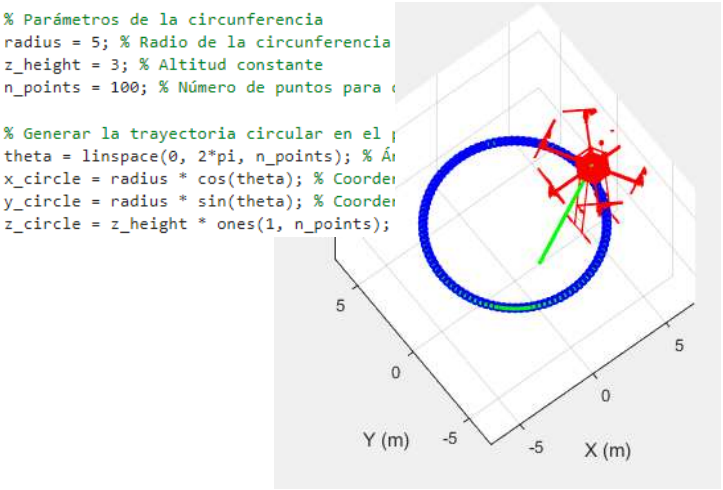


Figura 117. Trayectoria del UAV en forma circular

De igual manera, para diferentes rutas que se puedan realizar, tenemos una trayectoria en zigzag, como se puede observar en la Figura 118. En esta trayectoria, el dron mantiene una altura constante sin cambios ni perturbaciones mientras sigue un patrón de movimientos alternados hacia la izquierda y la derecha. Este diseño de trayectoria permite al dron desplazarse desde su punto de origen hasta su punto de destino de manera eficiente, asegurando que mantenga estabilidad y precisión en cada cambio de dirección.

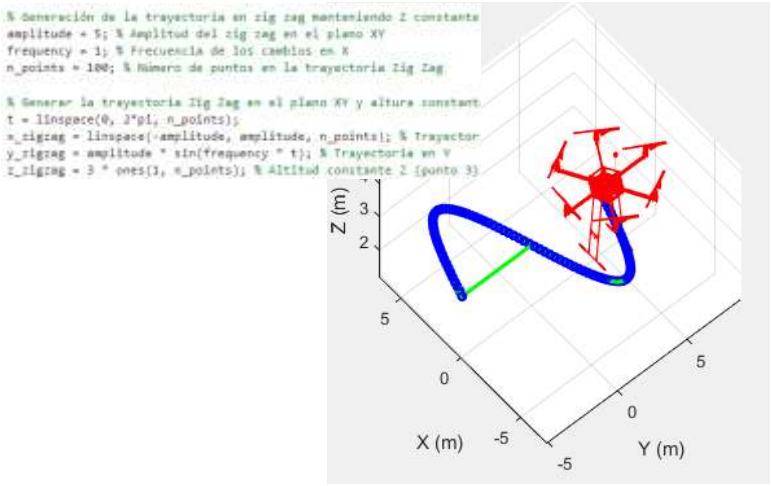


Figura 118. Trayectoria libre del UAV

3.1.13 Control PID

El controlador PID (Proporcional-Integral-Derivativo) es uno de los sistemas de control más utilizados en la industria debido a su simplicidad y eficacia. Su estructura básica lo convierte en una solución accesible para diversas aplicaciones. Su diseño permite una implementación sencilla con requisitos mínimos, lo que facilita su integración en una amplia variedad de sistemas, desde procesos industriales hasta aplicaciones en robótica y automatización. Además, su capacidad para combinar las acciones proporcional, integral y derivativa le confiere una gran versatilidad, permitiendo ajustar dinámicamente el comportamiento de un sistema y mantener un equilibrio óptimo entre precisión, estabilidad y respuesta ante perturbaciones.

3.1.14 Simulación para Control Proporcional, Integral y Derivativo (PID)

Para desarrollar la simulación del Control Proporcional, Integral y Derivativo de la trayectoria del UAV, es necesario establecer las ecuaciones de movimiento, ya que el dron se desplazará únicamente en los ejes z e y, considerando además el ángulo Ø. Teniendo en cuenta que la masa es de 3.36 Kg y el momento de inercia tiene un valor de 0,1 unidades.

$$\ddot{z} = g - u1\, m \cdot cos\,(\emptyset) \qquad (6)$$

$$\ddot{y} = u1\, m \cdot sin\,(\emptyset) \qquad (7)$$

$$\emptyset = u2\ Ixx \qquad (8)$$

$$m = 3.36\ Kg$$

$$Ixx = 0{,}1\ unidades$$

Ruta de referencia

La ruta de referencia del dron, mostrada en la Figura 119, combina movimientos sinusoidales en dos direcciones: en la vertical (z), el dron sube y baja con una amplitud de 5 unidades, y en la horizontal (y), se desplaza lateralmente con una amplitud de 0.5 unidades. Ambos movimientos, definidos por funciones sinusoidales a diferentes velocidades, conforman la ruta que el dron debe seguir en la simulación.

- $z = 5sin(t)$ $\qquad\qquad$ $y = 0.5sin(0.5t)$

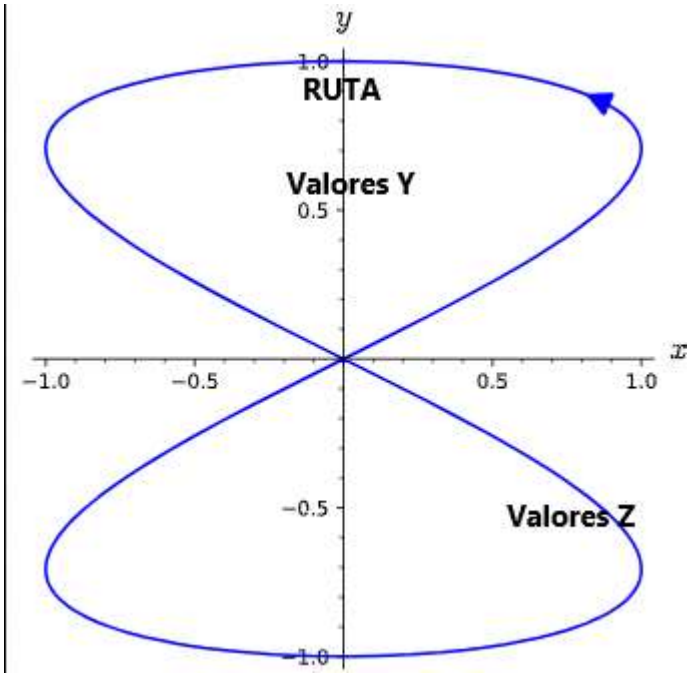


Figura 119. Trayectoria base

Implementación en el software Matlab (Simulink)

Como podemos observar en la Figura 120, tenemos el diagrama de bloques ejecutado en Simulink considerando los parámetros anteriormente definidos. La simulación del control PID aplicado a las trayectorias de un dron, empleando este software, representa

una herramienta eficiente para analizar y optimizar el desempeño del controlador en un entorno visual y gráfico.

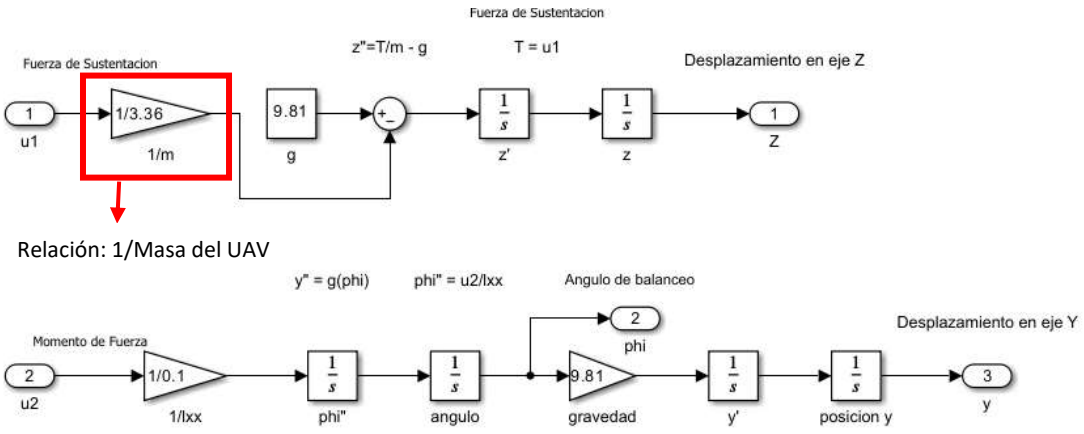


Figura 120. Subsistema para PID

Con el subsistema desarrollado, se crean bloques destinados a generar las trayectorias requeridas para el dron. Para poder lograr esto se usa funciones matemáticas, generadores de señales y la probabilidad de integrar diferentes trayectorias previamente definidas.

En la Figura 121, podemos visualizar el control PID desarrollado en Simulink donde inicia un zum de polaridad para que el fallo sea la diferencia de la entrada Z y la salida Z. con la integración de una onda que tiene un valor de 5 unidades de amplitud y una frecuencia constante de la unidad en el eje Z se demuestra el control PID para el fallo y una salida Z. Entonces para el movimiento en Y es objetiva un enlace en cascada de la onda de entrada y el control PID. La salida de Y es necesaria para volver a conectar y ajustar la utilidad del controlador. Dicho control es constante ya que no presenta una integración entonces la configuración de estos parámetros se los hace dentro de cada uno de los controladores PID. Para obtener una señal estabilizada se necesita una parte de saturación donde podamos visualizar a Phi y todo el alcance que permita.

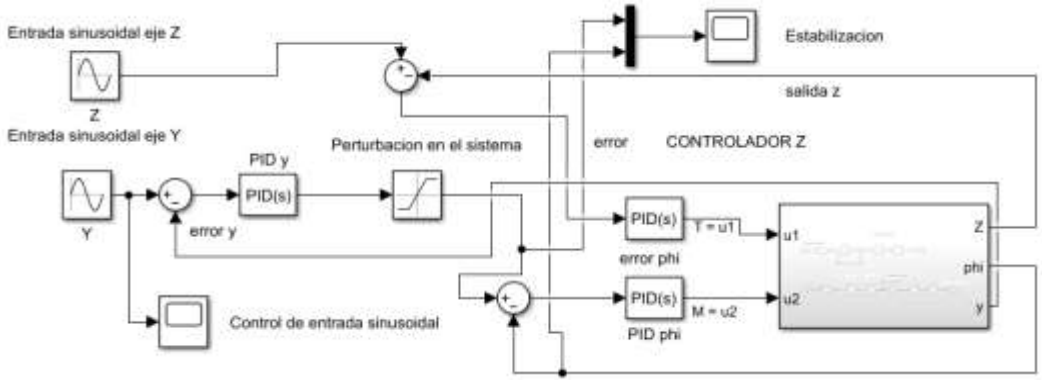


Figura 121. Sistema de control PID

Después de ejecutar el programa de control PID realizado en Simulink podemos visualizar la estabilización mostrada en la Figura 122, la cual podemos notar que se efectúa llegando a los 4.6 m de altura.

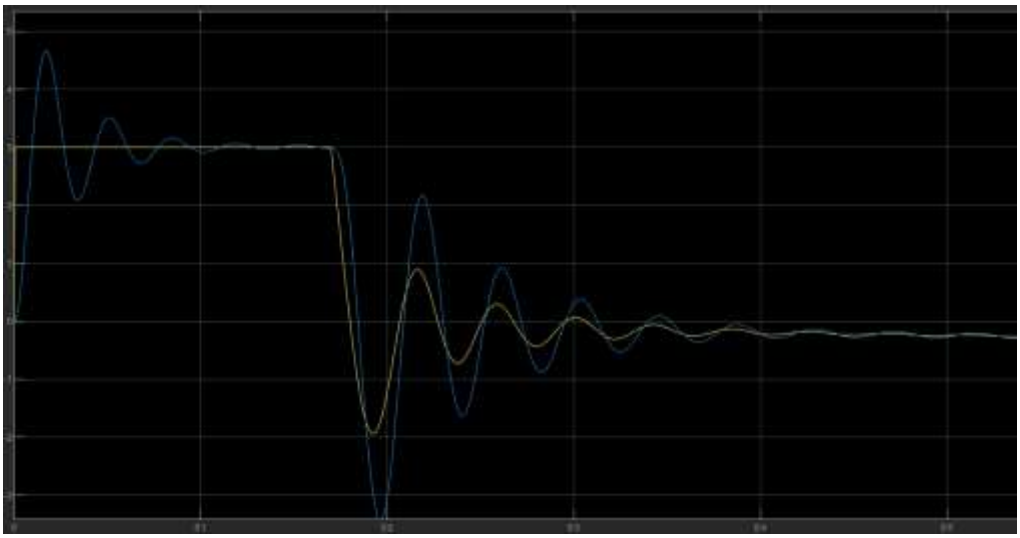


Figura 122. Altura de estabilización control PID

3.1.15 Interfaz del sistema de control y monitoreo del hexacoptero basado en Python

El control y monitoreo del UAV solo por un software libre como se realizó en Mission Planner es útil, pero en cuanto a implementar configuraciones necesarias, específicas y complementarías para poder configurarlo a base de scripts que se han realizado en Python es óptimo específicamente para usarlo desde la estación terrena permitiendo incorporar más herramientas para la configuración y uso del UAV, una de las bases de la configuración es la compatibilidad del UAV hexacóptero Tarot 860 PRO con la

configuración en base a algoritmos de programación esto arrancando por utilizar un protocolo de comunicación que acepte la placa controladora de Vuelo Pixhawk 2.4.8, y este es MAVLink como utilidad específicamente para drones se ha encontrado librerías las cuales ayudan a programar en Python utilizando la librería Pymavlink como base para el monitoreo y control del mismo, además de utilizar algunas librerías más para un óptimo desarrollo de un sistema de control y monitoreo estas se especifican en la Tabla 34.

Tabla 34. Librerías utilizadas en el programa de sistema de control y monitoreo del UAV

LIBRERIAS	**DESCRIPCIÓN**
pymavlink	Uso para recibir mensajes MAVLink, además de mandar comandos, rutinas o configuraciones a la placa Pixhawk.
matplotlib	Uso en la creación de gráficos en dos dimensiones, específicamente en diagramas de líneas.
matplotlib.backends.backend_tkagg	Uso en simulación de trayectorias por frames en gráficas 2D.
numpy	Uso en la estructura de datos tipo array de un mismo tipo organizada en forma de tabla o cuadrícula de distintas dimensiones específicamente en el cálculo numérico y el análisis de datos.
xml.etree.ElementTree	Uso con la llamada de una API sencilla y eficaz para analizar y generar datos en formato XML.
subprocess	Uso en lanzar nuevos procesos, en correr otro script a partir de la interfaz principal.
threading	Uso en el módulo que construye interfaces de hilado de alto nivel sobre otro para la recolección de parámetros del UAV.
time	Uso para trabajar con funciones relacionadas con el tiempo. Permite medir, manipular y formatear el tiempo, además de realizar pausas en la ejecución del programa.
Tkinter	Uso para la creación de interfaces gráficas basada específicamente en un escenario en Python.

a. Interfaz gráfica de control y monitoreo del UAV (Hexacóptero)

Se utilizo las librerías de Tkinter para la creación de una interfaz gráfica basada en Python esta se visualiza en la Figura 123, esta se dividió en bloques empezando desde el primer bloque de conexión donde se tuvo dos apartados para ingresar por teclado el puerto y los baudios también dos botones para la desconexión y conexión del dron, segundo bloque de controles este fue configurado con botones para armado, desarmado, cambios de modo Estabilizado, Loiter y Auto, además de botones para cargar misión, cargar archivo KML e iniciar misión, para la planificación de misiones se añadió dos botones de simulación de sistema de georreferencia y trayectorias, tercer bloque de Datos del UAV, cuarto bloque de Graficas en tiempo real y por último mensajes de estado del dron de todos los controles antes mencionados para monitoreo o acciones que se enviaron al dron.

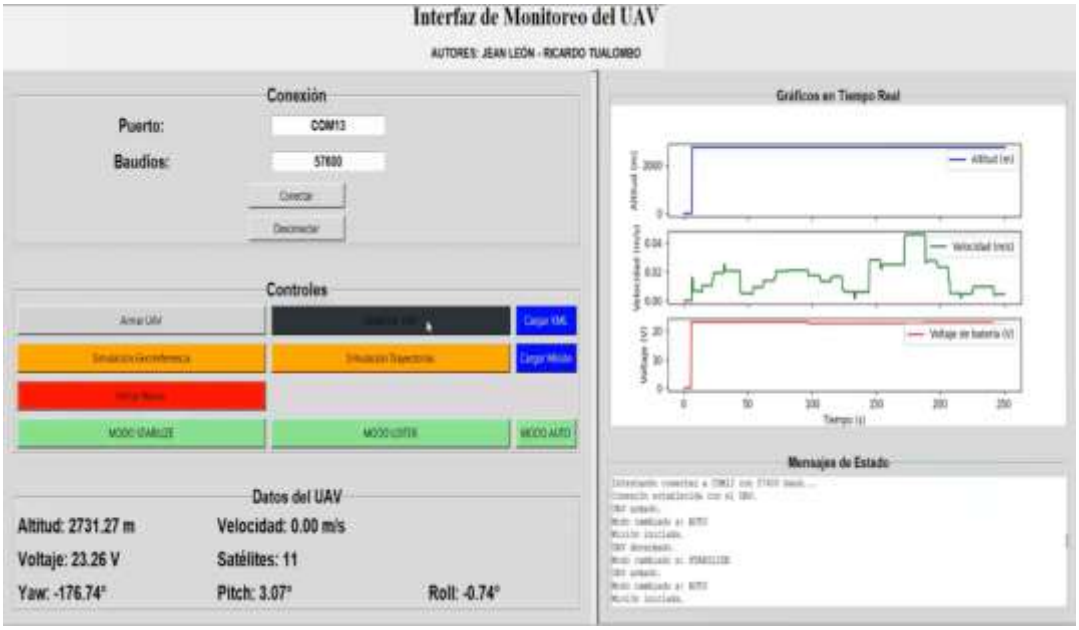


Figura 123. Interfaz gráfica realizada en Tkinter para control y monitoreo del UAV

b. Conexión y Desconexión de la estación terrena con el UAV (Hexacóptero)

Para establecer comunicación con el UAV y la estación terrena fue necesario a ver establecido una conexión previa con el dron esto mediante los módulo de telemetría integrados tanto en el dron como en la estación terrena, se utilizó la función connect_to_uav esta viene detallada principalmente por 2 variables globales que son master y connected las cuales ayudaron con la conexión e validación con el dron para

ingresar por teclado el puerto y el número de baudios para la comunicación que en este mavlink_connection que tomando las variables anteriores para establecer conexión previa, la función antes mencionada se muestra en la Figura 124, al realizar pruebas se visualizara un mensaje en la ventana de mensajes de estado conexión exitosa esto como se visualiza en la Figura 125.

```
# Función para conexión
def connect_to_uav():
    global master, connected
    try:
        port = port_entry.get()
        baud = int(baud_entry.get())
        log_text.insert(tk.END, f"Intentando conectar a {port} con {baud} baud...\n")
        master = mavutil.mavlink_connection(port, baud)
        master.wait_heartbeat()
        connected = True
        log_text.insert(tk.END, "Conexión establecida con el UAV.\n")
        threading.Thread(target=update_data, daemon=True).start()
    except Exception as e:
        log_text.insert(tk.END, f"Error al conectar: {e}\n")
```

Figura 124. Función para conexión con el UAV y la estación terrena



Figura 125. Mensaje de alerta previo a la conexión al UAV.

Para la desconexión con el dron, se estableció otra función disconnect_from_uav la cuál cambia el estado de la variable global connected a valor booleano Falso para quitar el estado anterior y cortar la conexión antes establecida con el dron se podrá visualizar el código de la función antes mencionada en la Figura 126, además indicando un mensaje de alerta para poder comprobar la desconexión del UAV como se muestra en la Figura 127.

```python
# Función para desconectar
def disconnect_from_uav():
    global master, connected
    if master:
        master.close()
    connected = False
    log_text.insert(tk.END, "Desconectado del UAV.\n")
```

Figura 126. Función para la desconexión con el UAV.



Figura 127. Mensaje de alerta previo a la desconexión con el UAV.

c. Controles con diferentes acciones predeterminadas para el UAV (Hexacoptero)

Se establecieron diferentes controles de acuerdo con comandos MAVLink previos para el control del UAV desde la interfaz de control para aquello a continuación se presenta algunas de las configuraciones que se realizaron en la interfaz.

Armado y desarmado del UAV (Hexacóptero)

Para el armado del dron se utilizó la función arm_uav esta utiliza las variables globales de conexión al dron, cabe recalcar que fue necesario primero la conexión previa para la comunicación con el dron, el comando MAVLink principal utilizado es de tipo MAV_CMD_COMPONENT_ARM_DISARM anteriormente explicado, específicamente utilizando el método command_long_send, el cual se basa como parámetro principal el sistema y componente objetivo del dron, para mandar un valor de 1 y armar el dron, teniendo en cuenta que se utilizó el método motors_armed_wait para mandar una espera de ejecución para confirmar que los motores estén armados, todo el código se visualiza en la Figura 128, además de enviar un mensaje de alerta que se visualizó en el bloque de mensajes de estado y el armado del dron en físico como se muestra en la Figura 129.

```python
#función de armado del UAV
def arm_uav():
    global master, connected
    if connected and master:
        try:
            #Comando MAVLink principal para el envio de
            #parámetros de armado del UAV
            master.mav.command_long_send(
                master.target_system,
                master.target_component,
                mavutil.mavlink.MAV_CMD_COMPONENT_ARM_DISARM,
                0,
                1, 0, 0, 0, 0, 0, 0
            )

            #Comando de espera para armado de motores
            master.motors_armed_wait()
            log_text.insert(tk.END, "UAV armado.\n")
        except Exception as e:
            log_text.insert(tk.END, f"Error al armar: {e}\n")
```

Figura 128. Función de armado de motores del UAV



Figura 129. Mensaje de alerta y visualización en físico del UAV armado

Para el desarmado del dron se utilizó la función disarm_uav también utilizó las variables globales de conexión al dron, manteniendo la conexión con el dron, su funcionamiento es utilizado con el comando MAVLink principal utilizado es de tipo MAV_CMD_COMPONENT_ARM_DISARM, específicamente utilizando el método command_long_send, se ha utilizado el método set_mode_send para enviar un mensaje MAVLink al sistema objetivo (target_system y target_component), para mandar un valor de 0 y desarma el dron, teniendo en cuenta que se utilizó el método

motors_disarmed_wait para mandar una espera de ejecución para confirmar que los motores estén desarmados, todo el código se visualiza en la Figura 130, además de enviar un mensaje de alerta que se visualizó en el bloque de mensajes de estado y el desarmado del dron en físico como se muestra en la Figura 131.

```python
#Función de desarmado del UAV
def disarm_uav():
    global master, connected
    if connected and master:
        try:
            #Comando MAVLink principal para el envio de
            #parámetros de desarmado del UAV
            master.mav.command_long_send(
                master.target_system,
                master.target_component,
                mavutil.mavlink.MAV_CMD_COMPONENT_ARM_DISARM,
                0,
                0, 0, 0, 0, 0, 0, 0
            )
            #Comando de espera para desarmado de motores
            master.motors_disarmed_wait()
            #Mensaje de alerta de desarmado del UAV
            log_text.insert(tk.END, "UAV desarmado.\n")
        except Exception as e:
            log_text.insert(tk.END, f"Error al desarmar: {e}\n")
```

Figura 130. Función de desarmado del UAV



Figura 131. Mensaje de alerta y visualización en físico del UAV desarmado

Cambios de modos de vuelo del UAV (Hexacóptero)

De acuerdo con las configuraciones que se habilitó en los modos de vuelo del dron, se utiliza la función change_mode que permitió realizar el cambio del modo de vuelo del dron utilizando especificamente un comando MAVLink. habilitando el modo personalizado mediante MAV_MODE_FLAG_CUSTOM_MODE_ENABLED. Inicialmente, se verificó si el modo solicitado está presente en el mapeo de modos del controlador del vehículo, accediendo al método mode_mapping. En caso de que el modo no sea válido, se registra un mensaje de error en el bloque de los mensajes de estado de la interfaz, junto con una lista de los modos disponibles. Si el modo es válido, se obtuvo su identificador asociado (mode_id toma el valor de acuerdo con el modo el cuál se envió) y se ha utilizado el método set_mode_send para enviar un mensaje MAVLink al sistema objetivo (target_system) este código de la función change_mode antes mencionada se visualiza en la Figura 132. Como última parte se informa en la interfaz que el modo de vuelo ha sido cambiado de manera exitosa como se muestra en la Figura 133.

```
#Función para cambio de modo de vuelo del UAV
def change_mode(mode):
    """
    Cambia el modo de vuelo del UAV.
    """
    #Mapeo de modos disponibles dentro del UAV y verifica si es valido
    if mode not in master.mode_mapping():
        log_text.insert(tk.END,f'Modo desconocido: {mode}\n')
        log_text.insert(tk.END,'Modos disponibles:', list(master.mode_mapping().keys()),"\n")
        return

    mode_id = master.mode_mapping()[mode]
    #Envio de mensajes al UAV para cambio de modo de acuerdo a la variable
    #MODE_ID que tiene el modo requerido
    master.mav.set_mode_send(
        master.target_system,
        mavutil.mavlink.MAV_MODE_FLAG_CUSTOM_MODE_ENABLED,
        mode_id
    )
    log_text.insert(tk.END, f'Modo cambiado a: {mode} \n')
```

Figura 132. Función de cambio de modo de vuelo del UAV

Se ha requerido para el inicio de las misiones para armar los motores del dron estar en modo estabilizado de forma que después que este armado se pueda pasar a modo Auto y empezar con la misión estos pasos son requeridos para tener un vuelo óptimo algunos de los mensajes de alerta en la interfaz se muestran en la Figura 134.



Figura 133. Mensaje de alerta de cambio de modo de vuelo Estabilizado y visualización en la pantalla OLED del UAV



Figura 134. Mensaje de alerta pasos requeridos para inicio de misión y visualización en la pantalla OLED del UAV

Leer archivo KML previo a la escritura en el UAV (Hexacóptero)

Para la planificación de misiones que se realizó como parte inicial para la prueba de vuelo se ha necesitado la lectura de un archivo KML esto sirvió para leer los datos y extraer solo los datos requeridos para poder cargar la misión al dron desde la interfaz,

entonces se utilizó la función read_kml, permitió procesar datos geoespaciales almacenados en archivos KML para la extracción de puntos de interés (waypoints). Esta función utiliza la librería del módulo xml antes mencioada para analizar la estructura XML del archivo y registrar los espacios de nombres necesarios para interpretar correctamente los elementos definidos en KML y GX. A través de un recorrido por los elementos “Placemark”, se extrajeron las coordenadas geográficas (latitud y longitud) desde los elementos secundarios “coordinates”, convirtiendo sus valores en flotantes. Además, se busca información de altura en el elemento secundario “description”, verificando si contiene el prefijo "Altura:"; en caso contrario, se asigna un valor predeterminado de 0 metros esta función read_kml se muestra en la Figura 135. Los datos procesados, que incluyen latitud, longitud y altura, se almacenan como tuplas en una lista denominada waypoints. Por último, se envia un mensaje de alerta de la cantidad de puntos waypoint registrados en el sistema o si se tiene errores como se visualiza en la Figura 136.

```python
# Función para leer puntos desde un archivo KML
def read_kml(file_path):
    global waypoints
    waypoints = []  # Reset waypoints
    try:
        # Registrar espacios de nombres para el archivo KML
        namespaces = {
            'kml': 'http://www.opengis.net/kml/2.2',
            'gx': 'http://www.google.com/kml/ext/2.2'
        }
        tree = ET.parse(file_path)
        root = tree.getroot()

        # Recorrer todos los Placemark
        for placemark in root.findall(".//kml:Placemark", namespaces):
            # Obtener coordenadas
            coordinates = placemark.find(".//kml:coordinates", namespaces).text.strip()
            lon, lat, _ = map(float, coordinates.split(','))

            # Obtener altura desde la descripción (si está presente)
            description_elem = placemark.find(".//kml:description", namespaces)
            if description_elem is not None and "Altura:" in description_elem.text:
                description = description_elem.text.strip()
                alt = float(description.split(":")[1].strip().replace("m", ""))
            else:
                alt = 0.0  # Altura predeterminada si no está en la descripción

            # Agregar waypoint
            waypoints.append((lat, lon, alt))

        log_text.insert(tk.END,f"Archivo KML cargado con {len(waypoints)} waypoints.\n")
    except Exception as e:
        log_text.insert(tk.END,f"Error al leer el archivo KML: {e}\n")
```

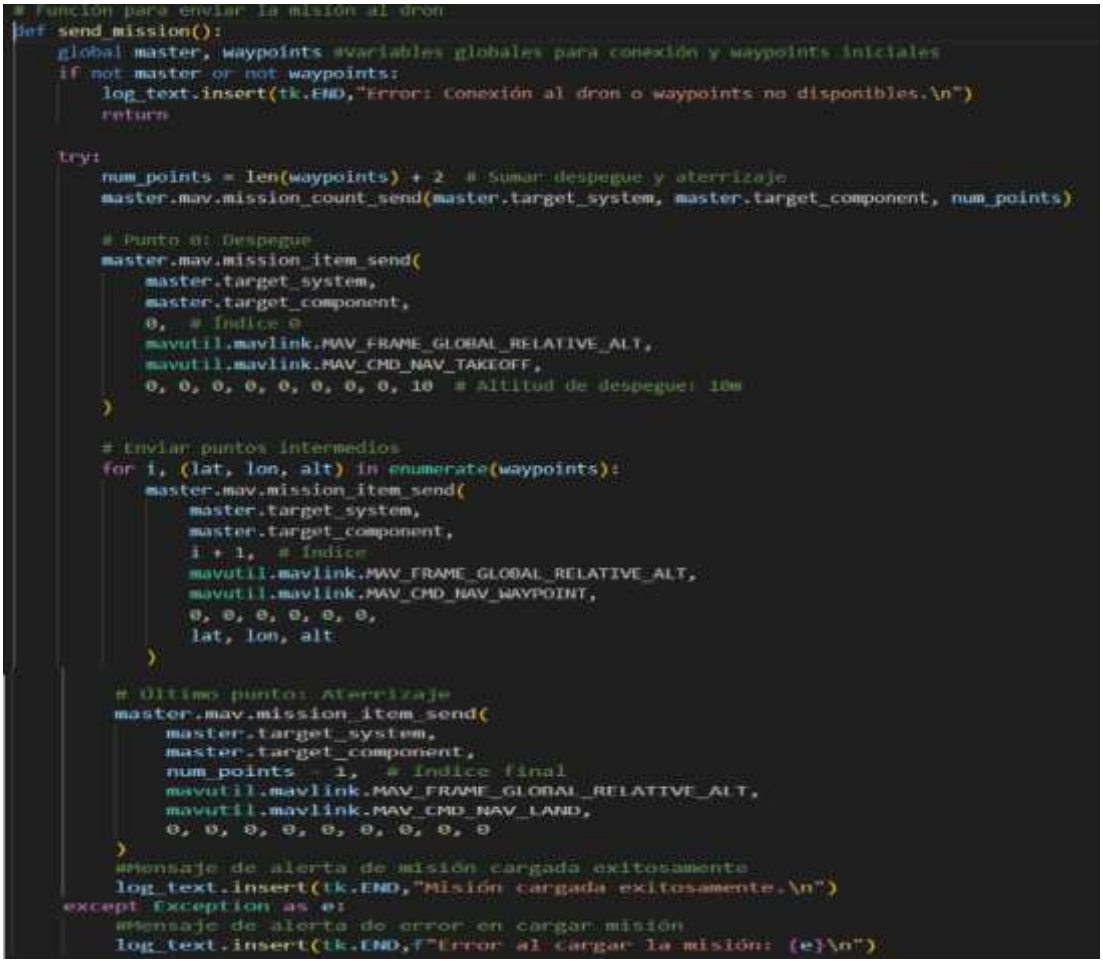
Figura 135. Función read_kml para cargar misión de la interfaz del UAV



Figura 136. Mensaje de alerta de archivo KML cargado del UAV

Cargar misión en el UAV(Hexacóptero)

Una vez que se planifico una misión mediante georreferenciación y simulación de trayectoria, la misión se carga mediante la función send_mission, que permitió cargar misiones en el dron mediante comandos MAVLink. El primer paso es comprobar la conexión con el dron y la disponibilidad de los waypoints, si alguno de ellos no está disponible registrar un mensaje de error en el bloque de mensajes de estado. Esta función calcula la puntuación total de la misión, incluido el punto de partida original y el punto de aterrizaje final. Utiliza el método Mission_count_send para informar el total de puntos de misión de drones. Luego utiliza Mission_item_send para enviar el punto de partida (despegue) con una altitud por defecto de 10 metros o asignada a una altura requerida y los parámetros correspondientes al comando MAV_CMD_NAV_TAKEOFF. Para después recorrer la lista de puntos de referencia y transmitir cada punto de referencia como un comando MAV_CMD_NAV_WAYPOINT que especifica las coordenadas de latitud, longitud y altitud relativa. Finalmente envía el último punto de misión configurado como comando de aterrizaje que es MAV_CMD_NAV_LAND esta función send_mission se visualiza en la Figura 137. Si se configuró todo satisfactoriamente, se mostrará un mensaje de alerta en el bloque de mensajes de Estado como se muestra en la Figura 138.



```
# Función para enviar la misión al dron
def send_mission():
    global master, waypoints #variables globales para conexión y waypoints iniciales
    if not master or not waypoints:
        log_text.insert(tk.END,"Error: Conexión al dron o waypoints no disponibles.\n")
        return

    try:
        num_points = len(waypoints) + 2  # Sumar despegue y aterrizaje
        master.mav.mission_count_send(master.target_system, master.target_component, num_points)

        # Punto 0: Despegue
        master.mav.mission_item_send(
            master.target_system,
            master.target_component,
            0,  # Índice 0
            mavutil.mavlink.MAV_FRAME_GLOBAL_RELATIVE_ALT,
            mavutil.mavlink.MAV_CMD_NAV_TAKEOFF,
            0, 0, 0, 0, 0, 0, 0, 10  # Altitud de despegue: 10m
        )

        # Enviar puntos intermedios
        for i, (lat, lon, alt) in enumerate(waypoints):
            master.mav.mission_item_send(
                master.target_system,
                master.target_component,
                i + 1,  # Índice
                mavutil.mavlink.MAV_FRAME_GLOBAL_RELATIVE_ALT,
                mavutil.mavlink.MAV_CMD_NAV_WAYPOINT,
                0, 0, 0, 0, 0, 0,
                lat, lon, alt
            )

        # Último punto: Aterrizaje
        master.mav.mission_item_send(
            master.target_system,
            master.target_component,
            num_points - 1,  # Índice final
            mavutil.mavlink.MAV_FRAME_GLOBAL_RELATIVE_ALT,
            mavutil.mavlink.MAV_CMD_NAV_LAND,
            0, 0, 0, 0, 0, 0, 0, 0, 0
        )
        #Mensaje de alerta de misión cargada exitosamente
        log_text.insert(tk.END,"Misión cargada exitosamente.\n")
    except Exception as e:
        #Mensaje de alerta de error en cargar misión
        log_text.insert(tk.END,f"Error al cargar la misión: {e}\n")
```

Figura 137. Función para cargar misión en el UAV



Figura 138. Mensaje de alerta de misión cargada correctamente en el UAV

Inicio de misión para el UAV desde la estación terrena

Se establece un control para iniciar la misión una vez se armado los motores y estando en el modo de vuelo “Estabilizado” del dron, la función start_mission establece el empezar la ejecución de una misión programada utilizando comandos MAVLink. Como primera estancia se verifico si existe una conexión activa con el dron; que es esencial para los controles de la interfaz. Si la conexión es válida, la función establecerá el modo de vuelo del dron en AUTO mediante el método set_mode_auto(), lo que habilita al dron para seguir la misión de forma autónoma. Posteriormente, envía

un comando MAV_CMD_MISSION_START utilizando command_long_send(), el cual indica al dron que inicie la ejecución de los puntos de la misión desde el primero el código de la función start_mission se visualiza en la Figura 139. Al terminar el proceso anterior mencionado se envió un mensaje de alerta al bloque de mensajes de estado como se indica en la Figura 140, informando que la misión ha comenzado; en el caso de tener errores se mostrara los posibles errores.

```python
# Función para iniciar la misión
def start_mission():
    global master
    #Comprobación de conexión con el dron correcta
    if not master:
        log_text.insert(tk.END,"Error: Conexión al dron no disponible.\n")
        return

    try:
        master.set_mode_auto() #Cambio a modo de vuelo auto
        #Envio de datos para empezar la misión
        master.mav.command_long_send(
            master.target_system,
            master.target_component,
            mavutil.mavlink.MAV_CMD_MISSION_START,
            0,  # Confirmación
            0, 0, 0, 0, 0, 0, 0  # Parámetros no requeridos
        )
        #Mensaje de alerta de misión iniciada
        log_text.insert(tk.END,"Misión iniciada.\n")
    except Exception as e:
        log_text.insert(tk.END,f"Error al iniciar la misión: {e}\n")
```

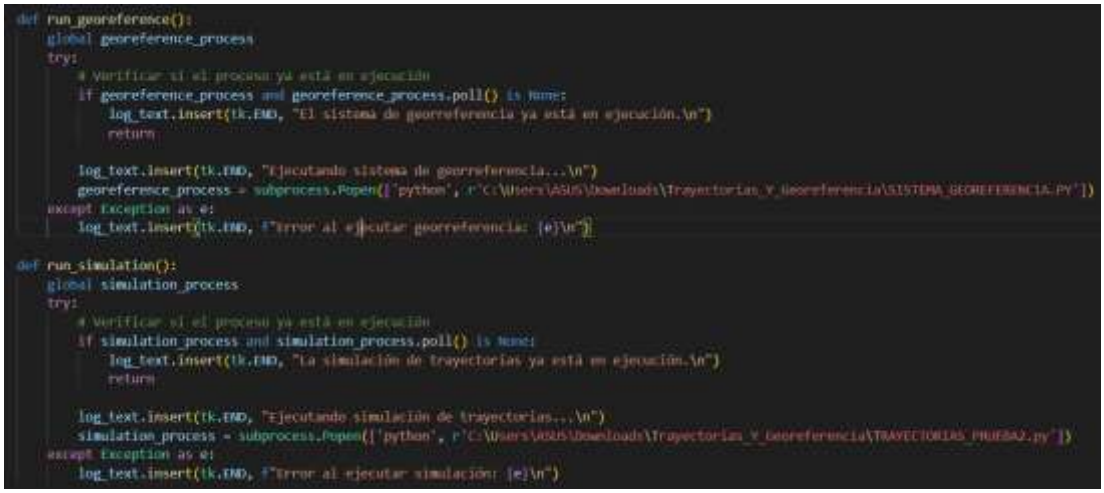
Figura 139. Función para iniciar la misión cargada en el UAV



Figura 140. Mensaje de alerta en la interfaz e inicio de misión en físico del UAV

Simulaciones de sistema de georreferencia y trayectorias para el UAV(Hexacóptero)

Para poder ejecutar los otros scripts los cuáles sirvieron para la planificación de misiones y simulación de trayectorias, se necesitó ejecutar como un subproceso para poder correr las demás simulaciones y a ver realizado un análisis antes de las pruebas de vuelo se utilizó dos funciones run_georeference y run_simulation, ambas utilizan el módulo subprocess para ejecutar archivos correspondientes ubicados en el sistema local se puede visualizar el código en la Figura 141 y se registró un mensaje que indica que se estará ejecutando el script esto de acuerdo con la necesidad de ejecutar cada script esto se muestra en la Figura 142.



```
def run_georeference():
    global georeference_process
    try:
        # Verificar si el proceso ya está en ejecución
        if georeference_process and georeference_process.poll() is None:
            log_text.insert(tk.END, "El sistema de georreferencia ya está en ejecución.\n")
            return

        log_text.insert(tk.END, "Ejecutando sistema de georreferencia...\n")
        georeference_process = subprocess.Popen(['python', r'C:\Users\ASUS\Downloads\Trayectorias_Y_Georeferencia\SISTEMA_GEORREFERENCIA.PY'])
    except Exception as e:
        log_text.insert(tk.END, f"Error al ejecutar georreferencia: {e}\n")

def run_simulation():
    global simulation_process
    try:
        # Verificar si el proceso ya está en ejecución
        if simulation_process and simulation_process.poll() is None:
            log_text.insert(tk.END, "La simulación de trayectorias ya está en ejecución.\n")
            return

        log_text.insert(tk.END, "Ejecutando simulación de trayectorias...\n")
        simulation_process = subprocess.Popen(['python', r'C:\Users\ASUS\Downloads\Trayectorias_Y_Georeferencia\TRAYECTORIAS_PRUEBA2.py'])
    except Exception as e:
        log_text.insert(tk.END, f"Error al ejecutar simulación: {e}\n")
```

Figura 141. Funciones para ejecutar subprocesos de los scripts del sistema de georreferencia y simulación de trayectorias

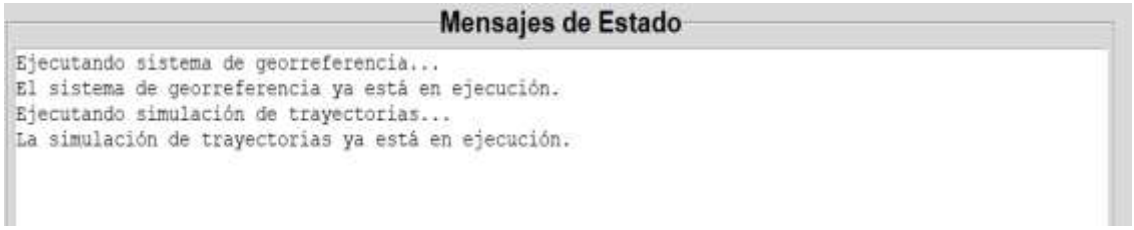


Figura 142. Mensaje de alerta ejecución de scripts para planificación de misiones y simulación de trayectorias del UAV

d. Monitoreo de parámetros del UAV (Hexacóptero)

Para el monitoreo de parámetros del dron se utilizó la función update_data esta se encargó de recibir, procesar y actualizar los datos del dron en tiempo real a través de mensajes MAVLink. Teniendo en cuenta que la conexión al dron es necesaria, la función utilizó un bucle continuo y ha utilizado la interfaz global principal para recibir datos. Dependiendo del tipo de mensaje recibido, recupera información relevante como altitud y velocidad (VFR_HUD), voltaje de la batería (BATTERY_STATUS), número de satélites visibles (GPS_RAW_INT) y la orientación del dron en relación con los valores de pitch, yaw y roll (ATTITUDE) el código de visualiza en la Figura 143. Para después actualizar las etiquetas de las gráficas correspondientes para indicar estos valores en los bloques de Datos del UAV y graficas en tiempo real. Además, sincroniza los datos recibidos con registros de tiempo, los almacena en listas globales y actualiza los gráficos instantáneamente. Esta funcionalidad incluye mecanismos para manejar errores e interrupciones, detener bucles cuando se encuentran problemas,

borrar datos guardados y mostrar mensajes de error en el bloque de mensajes de estado. Para evitar la sobrecarga, se introdujo un breve retraso (time.sleep(0.1)) en cada iteración del bucle así entonces los valores se mostrarán como la Figura 144.

```python
def update_data():
    global data_time, data_altitude, data_speed, data_voltage
    global data_yaw, data_pitch, data_roll, data_satellites
    global last_altitude, last_speed, last_voltage
    global connected

    start_time = time.time()
    while connected:
        try:
            msg = master.recv_match(blocking=True)
            if msg:
                current_time = time.time() - start_time

                # Altitud y velocidad (VFR_HUD)
                if msg.get_type() == 'VFR_HUD':
                    last_altitude = msg.alt
                    last_speed = msg.groundspeed
                    altitude_label.config(text=f"Altitud: {last_altitude:.2f} m")
                    speed_label.config(text=f"Velocidad: {last_speed:.2f} m/s")
                    # Asegurar que data_voltage también tenga un valor
                    if len(data_voltage) < len(data_time):
                        data_voltage.append(data_voltage[-1] if data_voltage else 0)

                # Voltaje de batería (BATTERY_STATUS)
                elif msg.get_type() == 'BATTERY_STATUS':
                    last_voltage = msg.voltages[0] / 1000.0  # Actualiza el último valor de voltaje
                    battery_label.config(text=f"Voltaje: {last_voltage:.2f} V")

                # Satélites disponibles (GPS_RAW_INT)
                elif msg.get_type() == 'GPS_RAW_INT':
                    data_satellites = msg.satellites_visible
                    satellites_label.config(text=f"Satélites: {data_satellites}")

                # Yaw, Pitch y Roll (ATTITUDE)
                elif msg.get_type() == 'ATTITUDE':
                    data_yaw = np.degrees(msg.yaw)
                    data_pitch = np.degrees(msg.pitch)
                    data_roll = np.degrees(msg.roll)
                    yaw_label.config(text=f"Yaw: {data_yaw:.2f}°")
                    pitch_label.config(text=f"Pitch: {data_pitch:.2f}°")
                    roll_label.config(text=f"Roll: {data_roll:.2f}°")

                # Añadir datos sincronizados a las listas
                data_time.append(current_time)
                data_altitude.append(last_altitude)
                data_speed.append(last_speed)
                data_voltage.append(last_voltage)

                update_graph()
            time.sleep(0.1)
        except Exception as e:
            log_text.insert(tk.END, f"Error al actualizar datos: {e}\n")
```

Figura 143. Función para actualización y recolección de parámetros del UAV



Figura 144. Parámetros del UAV mostrados en la Interfaz.

Graficas en tiempo real de parámetros del UAV

Para que las gráficas se actualicen instantáneamente se usó la función update_graph la cual es responsable de actualizar dinámicamente gráficos que representan parámetros importantes del dron, como la altitud, la velocidad y el voltaje de la batería, en función del tiempo transcurrido. Para aquello, se borró el gráfico existente usando el método de eliminación aplicado a cada subgráfico. Luego se utilizó los datos almacenados en las listas globales antes declaradas data_time, data_altitude, data_speed y data_Voltage para poder trazar las curvas correspondientes a cada parámetro y asignarles etiquetas descriptivas y colores únicos, el código de la función update_graph se muestra en la Figura 145.

Después también se configuro las etiquetas de los ejes y añadió las lecturas a cada subgráfica para mejorar la legibilidad de los datos visualizados. Por último, se usó el método de dibujo del objeto “canvas” para representar el gráfico actualizado en la interfaz de usuario como se visualiza en la Figura 146.

```
# Función para actualizar gráficos
def update_graph():
    if connected:
        # Limpiar gráficos
        ax[0].clear()
        ax[1].clear()
        ax[2].clear()

        # Actualizar gráficos con datos
        ax[0].plot(data_time, data_altitude, label="Altitud (m)", color="blue")
        ax[1].plot(data_time, data_speed, label="Velocidad (m/s)", color="green")
        ax[2].plot(data_time, data_voltage, label="Voltaje de batería (V)", color="red")

        # Configurar etiquetas y leyendas
        ax[0].set_ylabel("Altitud (m)")
        ax[1].set_ylabel("Velocidad (m/s)")
        ax[2].set_ylabel("Voltaje (V)")
        ax[2].set_xlabel("Tiempo (s)")
        ax[0].legend()
        ax[1].legend()
        ax[2].legend()

        # Dibujar el gráfico actualizado
        canvas_.draw()
```

Figura 145. Función de actualización de graficas en tiempo real de parámetros del UAV.



Figura 146. Graficas de parámetros del UAV en tiempo real

3.1.16 Simulación de georreferencia para planificación de misiones basado en Python

Para la planificación de rutas y trayectorias para el UAV es necesario trabajar en base a tecnologías y softwares que utilicen GPS como medio principal necesitando un sistema de georreferencia creado en base a programación en Python para comparar la precisión de este, esto para el sistema de navegación autónoma del UAV, entonces se especificara algunas de las librerías utilizadas dentro del programa de sistema de georreferencia esto se indica en la Tabla 35.

Tabla 35. Librerías utilizadas en el programa de georreferencia

Librerias	Descripción
Geopy	Uso para localizar las coordenadas de puntos de referencia en todo el mundo mediante geo codificadores de terceros y cálculo de distancias entre puntos especificando la lectura y escritura de rutas en un mapa de Openstreet Maps.
SimpleKML y Fastkml	Uso para creación y lectura de archivo kml de forma simple a base de puntos de georreferencia.
json	Uso en la creación de archivos en formato JSON de la conversión de coordenadas geográficas a coordenadas cartesianas.

Librerias	Descripción
folium	Uso en la visualización geoespacial de datos, esto como latitud y longitud de un punto georreferenciados y trazo de líneas y puntos en un Mapa.
tkinter	Uso para la creación de interfaces gráficas basada específicamente en un escenario en Python.

a. Interfaz gráfica del programa de georreferencia del UAV

Para la interfaz gráfica creada con Tkinter en un escenario en python, viene constituida con dos botones principales los cuáles son para leer un archivo KML y cargar un archivo en formato Mission lograr procesarlo en base a un algoritmo para trazar los puntos en un mapa y visualizarlos en una tabla, donde se indica el número de puntos de la misión como ID, y parámetros esenciales para la misión como latitud y Longitud, además de una descripción detallando el comando ya sea de despegue, waypoint o aterrizaje, se añadió otro parámetro importante como la altura a la cual se elevará el dron para generar los demás archivos, un bloque de mensajes como aviso de mensajes esta interfaz gráfica se visualizará en la Figura 147.

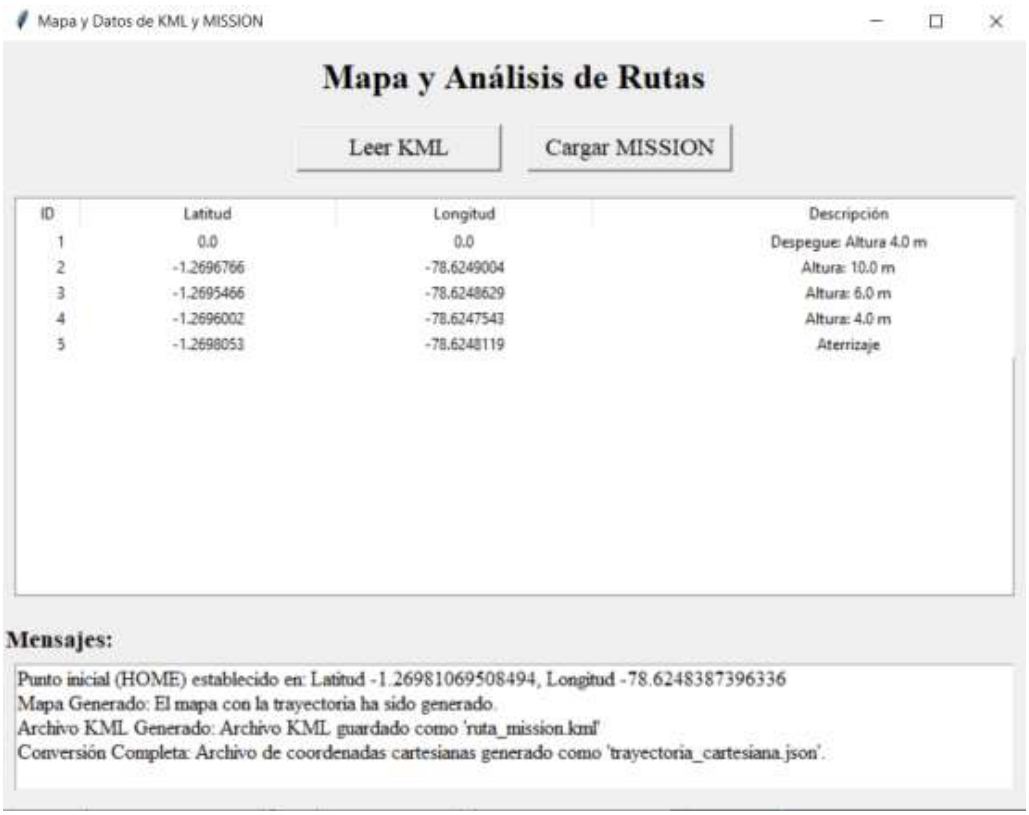


Figura 147. Interfaz gráfica realizada con la librería Tkinter en Python

b. Carga de archivos en formato MISSION en el programa de georreferencia

El programa se basó en la generación de puntos de georreferencia como guía de un archivo en formato MISSION una vez generado por la planeación de la misión realizado en el software Mission Planner, la función principal cargar_mission procesó un archivo en el formato MISSION, este extrae los datos geoespaciales relevantes y los convierte en un formato una vez procesados para la visualización y exportación, el comienzo en el solicitar la selección del archivo MISSION a cargar abriéndose una ventana emergente como se indica en la Figura 148, para cargar y procesar el contenido en formato JSON.

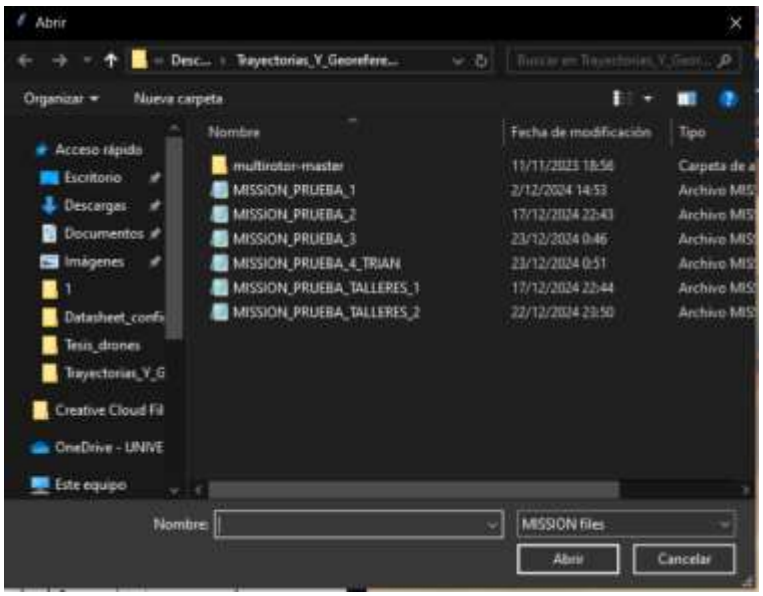
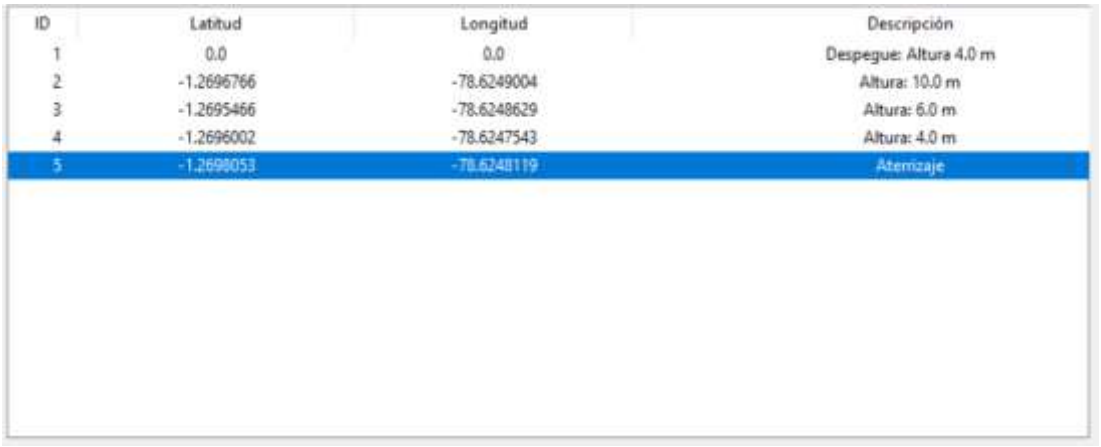


Figura 148. Cargar archivo en formato MISSION.

Se valido la existencia y la disponibilidad de la posición de inicio o punto de partida (plannedHomePosition) y se extrae las coordenadas de los puntos de la misión, incluyendo los comandos de takeoff, waypoint y land. Los parámetros procesados son latitud, longitud y altitud (Altura), y se almacenan en una lista para la visualización en la tabla de la interfaz del programa como se muestra en la Figura 149, se llama a otra función convertir_a_cartesianas para convertir las coordenadas geográficas a coordenadas cartesianas usando la posición de inicio como origen, para después pasar actualizar la tabla y un mapa trazando los puntos, generando los archivos kml, json y html del mapa OpenStreet Map visualizando un mensaje para verificar la creación correcta de los mismos como se visualiza en la Figura 150.



ID	Latitud	Longitud	Descripción
1	0.0	0.0	Despegue: Altura 4.0 m
2	-1.2696766	-78.6249004	Altura: 10.0 m
3	-1.2695466	-78.6248629	Altura: 6.0 m
4	-1.2696002	-78.6247543	Altura: 4.0 m
5	-1.2698053	-78.6248119	Aterrizaje

Figura 149. Tabla de puntos con los datos procesados de la misión cargada

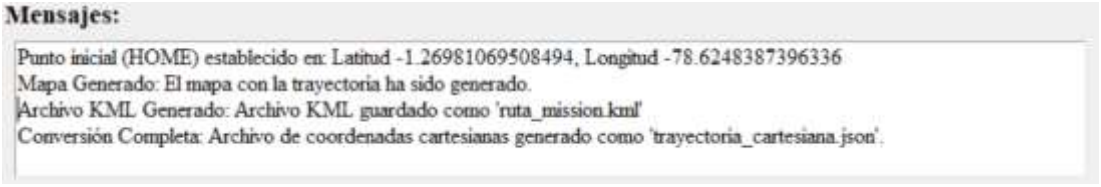


Figura 150. Creación de archivos KML, JSON y mapa generado en OpenStreetMap, además del punto inicial (HOME).

c. Generación de archivos de rutas en formato KML para lectura en el software Google Earth del programa de georreferencia

Para la generación de archivos en formato KML, se utilizó la función generar_kml esta cumple con la utilidad de crear un archivo KML que permita visualizar la trayectoria planificada con puntos trazados. Técnicamente, la función utilizó la librería "simplekml" para la estructura del archivo KML, como primera funcionalidad es la de agregar un punto inicial (representado como posición inicial o punto de HOME) con una altitud añadida como descripción de este. Luego iteró los puntos de la misión (excepto el punto inicial de la misión) en el archivo como puntos adicionales, cada uno identificado por un índice y una descripción adjunta. Finalmente, el archivo KML se guarda localmente como ruta_mission.kml e intentará abrirlo automáticamente usando un navegador predeterminado o Google Earth en caso de ser posible como se visualiza en la Figura 152. La estructura del código completo de la función generar_kml se muestra en la Figura 151, además un mensaje de verificación o de error en el bloque de mensajes.

```python
# Función para generar un archivo KML con líneas entre puntos
def generar_kml(puntos, home_lat, home_lon, home_alt):
    """Genera un archivo KML con puntos de la misión y líneas conectándolos para visualización en Google Earth."""

    # Crear el objeto KML
    kml = simplekml.Kml()

    # Añadir el punto inicial al KML con su descripción
    pnt_inicio = kml.newpoint(name="Punto Inicial", coords=[(home_lon, home_lat)])
    pnt_inicio.description = f"Altura máxima en el despegue: {home_alt} m"

    # Lista para almacenar las coordenadas de las líneas
    line_coords = [(home_lon, home_lat)]  # Coordenadas del punto inicial

    # Añadir los demás puntos al KML
    for idx, (lat, lon, desc) in enumerate(puntos[1:], start=1):
        # Crear un punto con su descripción
        pnt = kml.newpoint(name=f"Punto {idx}", coords=[(lon, lat)])
        pnt.description = desc

        # Añadir las coordenadas a la línea
        line_coords.append((lon, lat))

    # Crear una línea poligonal que conecta los puntos
    lin = kml.newlinestring(name="Trayectoria", coords=line_coords)
    lin.style.linestyle.color = simplekml.Color.red  # Color de la línea
    lin.style.linestyle.width = 2  # Grosor de la línea
# Guardar el archivo KML
kml_path = "ruta_mission.kml"
kml.save(kml_path)
mostrar_mensaje("Archivo KML Generado: Archivo KML guardado como 'ruta_mission.kml'")

# Intentar abrir Google Earth o el archivo KML directamente
try:
    webbrowser.open(kml_path)
except Exception as e:
    mostrar_mensaje(f"Error al Abrir: No se pudo abrir Google Earth automáticamente.\nArchivo guardado en {kml_path}")
```

Figura 151. Función para generar el archivo KML en el programa de georreferencia



Figura 152. Mensaje de gratificación de archivo KML generado y visualización de puntos de la ruta planificada en Google earth.

d. Creación de mapa OpenStreet para trazado de trayectorias del UAV del programa de georreferencia

Para la creación y visualización en un mapa diferente al de Google maps se implementó el mapa OpenStreet maps que nos concede la librería de geopy para trazar los puntos geoespaciales y trayectorias planificadas en la misión esto específicamente para una mejor precisión en la generación de los puntos coordenados de la misión para el UAV para aquello se utilizó la función trazar_puntos_en_mapa se usó la librería folium para la creación mapas interactivos que visualizan puntos en una ruta planificada y los conectan con líneas. Inicialmente crea un mapa con un nivel de zoom preestablecido centrado en la ubicación de inicio o ubicación del home del dron toma los datos (home_lat, home_lon) y se agregó un marcador para identificar esta ubicación. Para después se utilizó otra función auxiliar para calcular la distancia geodésica entre los puntos del camino y trazar los puntos restantes (excepto el punto inicial) en el mapa. Cada punto es representado por un marcador con una descripción emergente que describe el número de punto, algunos parámetros como latitud, longitud y distancia acumulada desde el punto anterior. Así, una polilínea conecta todos los puntos secuencialmente, resaltando todo el camino la estructura del código de esta función se muestra en la Figura 153. El mapa generado se guarda como un archivo HTML y se abrirá automáticamente en el navegador en el caso de ser posible como se muestra en la Figura 154.

```python
# Función para trazar los puntos en el mapa y mostrar latitud, longitud y el número de punto
def trazar_puntos_en_mapa(puntos, home_lat, home_lon):
    """Traza los puntos en un mapa interactivo, conecta con líneas y muestra latitud, longitud y número de punto"""
    # Crear el mapa centrado en el punto inicial
    mapa = folium.Map(location=[home_lat, home_lon], zoom_start=12)
    marker_cluster = MarkerCluster().add_to(mapa)

    # Añadir el punto inicial al mapa (solo como referencia)
    folium.Marker(location=[home_lat, home_lon], popup="Punto Inicial").add_to(marker_cluster)
    # Calcular las distancias entre puntos
    distancias = calcular_distancias(puntos)
    # Conectar los puntos con líneas y mostrar latitud, longitud y número de punto
    coordinates = [(home_lat, home_lon)]  # Inicia la lista con el punto inicial

    for i, (lat, lon, desc) in enumerate(puntos[1:], start=1):  # omitir el primer punto
        if i == 1:
            distancia = 0  # Para el punto 1, asignamos 0 metros de distancia
        else:
            distancia = distancias[i - 1] if i - 1 < len(distancias) else 0  # Obtener distancia si existe
        # Descripción con número de punto, latitud, longitud
        popup_text = f"Punto {i}: {desc}<br>Latitud: {lat}<br>Longitud: {lon}<br>Distancia: {distancia:.2f} m"
        # Añadir el marcador en el punto
        folium.Marker(location=[lat, lon], popup=popup_text).add_to(marker_cluster)
        coordinates.append((lat, lon))  # Conectar la ruta desde el punto inicial hacia los demás puntos

    # Crear la línea que conecta todos los puntos
    folium.PolyLine(coordinates, color='blue', weight=2.5, opacity=1).add_to(mapa)
    # Guardar el mapa como archivo HTML
    mapa.save("mapa_trayectoria.html")
    mostrar_mensaje("Mapa Generado: El mapa con la trayectoria ha sido generado.")

    # Abrir el mapa en el navegador
    webbrowser.open("mapa_trayectoria.html")
```

Figura 153. Función para creación del mapa OpenStreet y trazar puntos y polígonos deseados del programa de georreferencia.

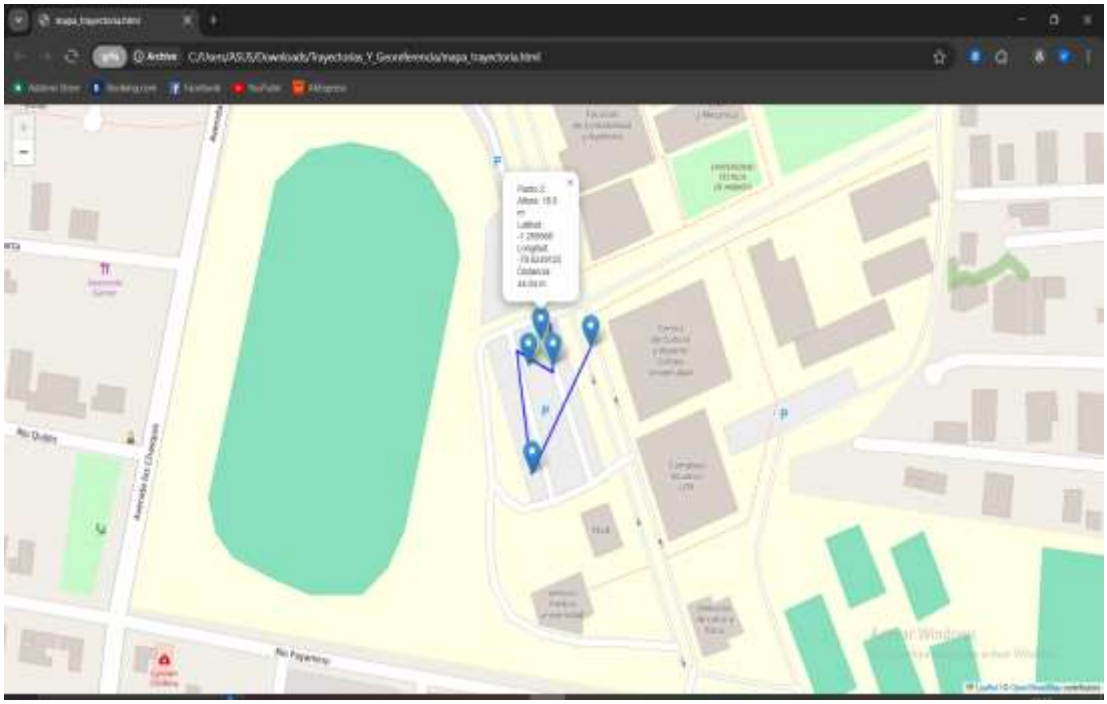


Figura 154. Mapa OpenStreet visualización de trayectoria cargada del programa de georreferencia.

También se utilizó la función Calculation_distances esta realizó el cálculo de la distancia entre puntos consecutivos en una ruta utilizando coordenadas geográficas.

Para cada par de puntos adyacentes, extrae su latitud y longitud y las utiliza como entrada para la función geodésica de la biblioteca geopy, que calcula la distancia entre dos ubicaciones en metros, dada la forma esférica de la Tierra. El valor resultante se almacena en una lista que se devolvió como salida la estructura principal del código de esta función se visualiza en la Figura 155.

```python
# Función para calcular las distancias entre puntos consecutivos
def calcular_distancias(puntos):
    """Calcula las distancias entre puntos consecutivos usando coordenadas geográficas."""
    distancias = []
    for i in range(len(puntos) - 1):
        lat1, lon1, _ = puntos[i]  # Tomamos solo latitud y longitud
        lat2, lon2, _ = puntos[i + 1]

        origen = (lat1, lon1)
        destino = (lat2, lon2)

        # Calculamos la distancia entre los puntos
        dist = geodesic(origen, destino).meters  # Distancia en metros
        distancias.append(dist)
    return distancias
```

Figura 155. Función para el cálculo de distancia entre puntos del mapa OpenStreet del programa de georreferencia.

e. Creación del archivo JSON de coordenadas geográficas a coordenadas cartesianas del programa de georreferencia

Fue necesario la creación de un archivo en formato *json* por la conversión a coordenadas cartesianas que son necesarias para la simulación de trayectorias del multirotor el formato del archivo se visualiza Figura 158, se utiliza como función principal geodetic_to_ecef realizó la conversión de coordenadas geográficas (latitud, longitud y altitud o altura) a un sistema de referencia cartesianas tridimensionales llamado ECEF (Earth-Centered, Earth-Fixed). Este sistema utiliza el centro de la Tierra como punto de partida y proyecta posiciones en un marco fijo con respecto a la Tierra.

Esta función emplea fórmulas matemáticas derivadas del modelo elipsoidal de la Tierra, considerando el radio ecuatorial y la excentricidad de la elipse terrestre para calcular la posición en los ejes X, Y y Z. Esta función convierte los valores de entrada de latitud y longitud a radianes, y el cálculo de las coordenadas se basa en el radio de curvatura en una latitud determinada y la relación trigonométrica para cada componente espacial de la proyección.

El resultado es un vector tridimensional en metros que represento la ubicación en el sistema ECEF. Este sistema es esencial para aplicaciones de navegación, georreferenciación y análisis espacial porque permitió el uso de un marco de referencia universal en lugar de depender de coordenadas geográficas relacionadas con la curvatura de la Tierra la estructura del código se muestra en la Figura 156.

```python
#Función conversion coordenadas geógraficas a ECEF
def geodetic_to_ecef(lat, lon, alt):
    """
    Convierte coordenadas geográficas (latitud, longitud, altitud) a ECEF
    (Earth-Centered, Earth-Fixed) en metros.
    """
    lat_rad = math.radians(lat) # Convertir latitud a radianes
    lon_rad = math.radians(lon) # Convertir longitud a radianes
    #Formular utilizadas para convertir las coordenadas geográficas de acuerdo al sistema ECEF
    N = R_EARTH / math.sqrt(1 - E_ECC**2 * math.sin(lat_rad)**2)
    X = (N + alt) * math.cos(lat_rad) * math.cos(lon_rad)
    Y = (N + alt) * math.cos(lat_rad) * math.sin(lon_rad)
    Z = (N * (1 - E_ECC**2) + alt) * math.sin(lat_rad)
    return [X, Y, Z]
```

Figura 156. Función de conversión de coordenadas geográficas a ECEF del programa de georreferencia.

Así, después se utilizó la función convertir_a_cartesianas para transformar una lista de puntos geográficos de acuerdo con los parámetros de latitud, longitud y datos adicionales respecto a un sistema cartesiano relativo, se tomó como referencia el punto inicial o el punto “home”, como primer punto se utilizó la función principal antes mencionada para la conversión de puntos de referencia de cada uno de la lista al sistema de coordenadas ECEF. Para después, se calculó las coordenadas relativas restando las posiciones ECEF de los puntos con respecto al punto de origen, obteniendo un sistema cartesiano local que facilita la representación y el análisis de las posiciones.

Además, la función procesó la descripción de cada punto para obtener información sobre su altura relativa y asignó un valor predeterminado de cero si los datos faltan o están mal formados. El resultado es una lista de coordenadas cartesianas tridimensionales relativas al punto de referencia, donde los valores X e Y representan desplazamientos en el plano horizontal y el valor Z corresponde a la altura la estructura del código se muestra en la Figura 157.

```python
#Función de conversión de puntos geográficos a coordenadas cartesianas
def convertir_a_cartesianas(home_lat, home_lon, puntos, home_alt):
    """
    Convierte una lista de puntos geográficos (lat, lon, alt) a coordenadas cartesianas
    relativas a un punto inicial (home_lat, home_lon).
    """
    origen = geodetic_to_ecef(home_lat, home_lon, home_alt)  # Origen basado en plannedHomePosition o el origen ECEF
    coordenadas_cartesianas = []
    for lat, lon, desc in puntos:
        cartesian = geodetic_to_ecef(lat, lon, 0)  # Ignora la altitud relativa al terreno
        try:
            alt = float(desc.split(":")[1].split()[0]) if "Altura" in desc else 0.0
        except (IndexError, ValueError):
            alt = 0.0  # Valor por defecto si no se puede extraer

        # Hacer las coordenadas relativas al origen
        coordenadas_cartesianas.append([cartesian[0] - origen[0], cartesian[1] - origen[1], alt])
    return coordenadas_cartesianas
```

Figura 157. Función de conversión de puntos ECEF a coordenadas cartesianas.

trayectoria_cartesiana.json > ...

```json
[
    [
        5120443.098866595,
        6251338.319053159,
        0.0
    ],
    [
        -27.747153798351064,
        13.779333426617086,
        20.0
    ],
    [
        -33.699325126828626,
        11.596588344313204,
        18.0
    ],
    [
        -18.807276470120996,
        14.76681930013001,
        25.0
    ],
    [
        -19.899574225069955,
        14.158469670452178,
        0.0
    ]
]
```

Figura 158. Formato del archivo json de las coordenadas cartesianas para la simulación de trayectorias.

3.1.17 Organización del código de simulación de trayectorias y control del hexacóptero basado en Python

Para la simulación de rutas del hexacóptero es basado en la cinemática representada en un apartado anterior al proyecto de investigación, el script en Python se basa en una simulación utilizando librerías de "Multirotor", se ha aplicado a un UAV (hexacóptero) para ello comenzado el especificar la organización del código con las clases principales para simular al vehículo que es la base para cargar parámetros al script y visualizar su comportamiento detallando cierta trayectoria, para la organización del código se creó un diagrama de clases que se visualiza en la Figura 159.

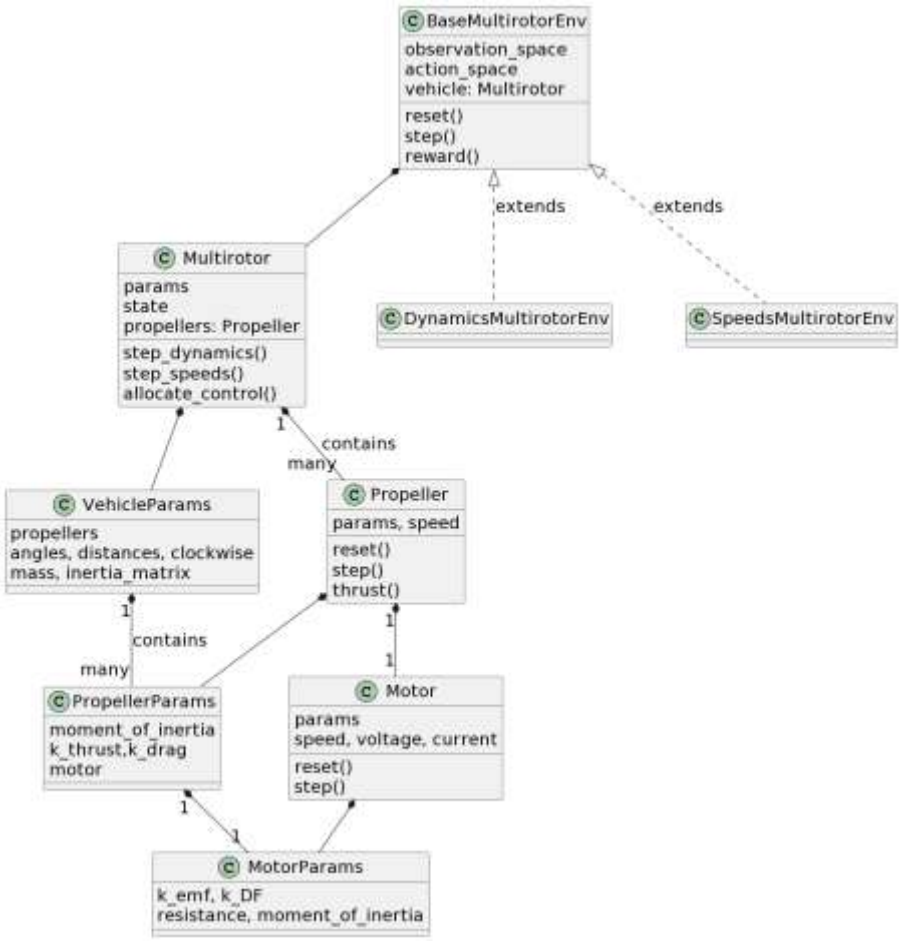


Figura 159. Diagrama de clases de la simulación de multirrotores

Para el diagrama de clases de este código de simulación de multirrotores en Python, donde se indican que clases se utilizan para modelar los diferentes componentes del

sistema y su interacción. Se explicará a continuación que contiene cada una de las clases, métodos y parámetros que incluye en cada clase.

- **BaseMultirotorEnv**: Es una clase principal para la creación de entornos de simulación de multirrotores, que en este caso es un hexacóptero. Contiene atributos (variables dentro de una clase) como observation_space, action_space, y vehicle, que es una instancia de la clase Multirotor también, incluye métodos básicos como reset(), step(), y reward(). Dos clases derivadas, "DynamicsMultirotorEnv" y "SpeedsMultirotorEnv", se extiende esta clase, específicamente para parámetros y el entorno del dron o vehículo para simularlo, que en este caso se aplica al hexacóptero.

- **Multirotor**: Este será una clase intermedia que representa el vehículo como tal para el hexacóptero con todos sus parámetros o características. Se tiene parámetros (params), un estado (state), y un conjunto de "Propellers" (hélices), que se definen como una lista de instancias de la clase Propeller. Los métodos step_dynamics(), step_speeds(), y allocate_control() se utilizan para calcular la dinámica del dron, ajustar velocidades, y asignar controles a las hélices, respectivamente.

- **VehicleParams**: Está clase es especializada esto al almacenar parámetros específicos del vehículo, como el número de hélices (propellers), sus ángulos, distancias, dirección de rotación (clockwise), masa, y una matriz de inercia (inertia_matrix). La clase Multirotor contiene una instancia de VehicleParams que define las características físicas del vehículo.

- **Propeller**: Está clase es especializada servirá para la representación de cada hélice individualmente. Para cada una de las hélices se tiene sus propios parámetros (params) y velocidad (speed). Los métodos reset(), step(), y thrust() permiten reiniciar el estado de la hélice, avanzar su dinámica, y calcular el empuje generado.

- **PropellerParams**: Está clase es especializada definirá parámetros específicos de cada hélice, como el momento de inercia, y coeficientes de empuje y arrastre (k_thrust y k_drag). Además, contiene un motor, representado por una instancia de la clase Motor.

- **Motor**: Está clase es especializada para el modelado del motor de cada hélice. Incluye parámetros (params), velocidad (speed), voltaje (voltage), y corriente (current). También tiene métodos reset() y step() para reiniciar su estado y actualizar su dinámica.

- **MotorParams**: Está clase es especializada contiene parámetros específicos del motor, como la constante de fuerza electromotriz (k_emf), la constante de torque (k_DF), resistencia, y momento de inercia. Estos parámetros ayudaron a modelar el comportamiento físico del motor en la simulación.

Una vez se definieron cada una de las clases, métodos y parámetros, es necesario especificar las relaciones que tienen entre sí, ya que en el diagrama de clases se indica la conexión entre ellas. Para ello, se especifican a continuación sus relaciones.

a. Relaciones de clases de la simulación de trayectorias del multirotor

La clase BaseMultirotorEnv es la clase base de la cual heredan DynamicsMultirotorEnv y SpeedsMultirotorEnv. En cambio, para Multirotor tiene una relación de composición con VehicleParams, lo que significa que cada multirrotor incluye un conjunto de parámetros que describen sus características físicas. Además, Multirotor también contiene una lista de Propeller, cada uno con sus propios parámetros (PropellerParams) y un motor (Motor). Para Propeller y Motor tienen sus propios parámetros específicos (definidos en PropellerParams y MotorParams, respectivamente).

Para la simulación del control del multirotor que se ha aplicado específicamente para un Hexa rotor, realizado de esta manera otro diagrama de clases que se visualiza en la Figura 160, para la simulación de control del Hexa rotor en Python, donde se indican que clases se utilizan para modelar los diferentes componentes del sistema y su interacción. Se explicará a continuación que contiene cada una de las clases, métodos y parámetros que incluye en cada clase.

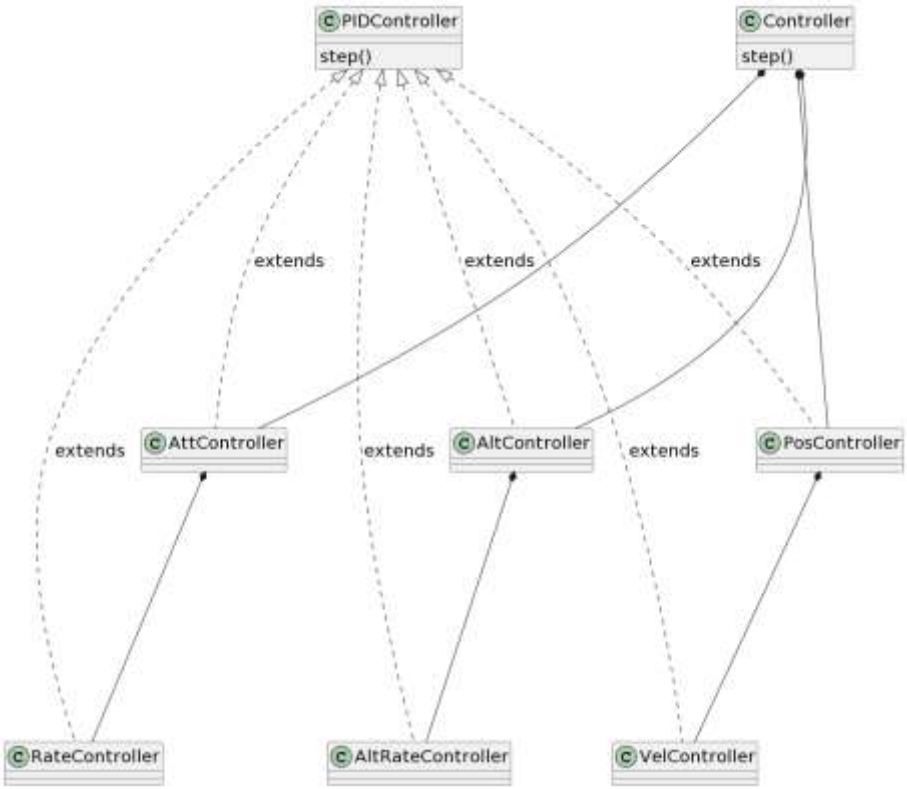


Figura 160. Diagrama de clases de la simulación de control

1. **PIDController:** Es una clase base genérica que se ha implementado un controlador proporcional-integral-derivativo (PID) es una clase principal. Esta clase viene definida por un método “step()” este realiza un paso de cálculo para actualizar el control basado en entradas, errores y las ganancias PID. Todas las clases heredan de PIDController, lo que significa que se implementó controladores PID personalizados en distintos niveles.

2. **Controller:** Esta es otra clase de base más general es una clase principal. Representa la estructura principal para controladores no necesariamente basados en PID. Para el método “step()” aplicado en este caso es cuál definió una operación genérica que todas las clases derivadas deben implementar. Esto incluye la lógica de actualización para posicionamiento o velocidad.

3. **AttController:** Esta clase es de tipo intermedia representando diferentes niveles de control de actitud, es decir, la cinemática del hexacoptero. Esta controlara la orientación tanto de roll, pitch y yaw del dron, extendiendo la funcionalidad de las clases “Controller” y PIDController, además de tener una relación jerárquica con el RateController.

4. **AltController:** Esta clase es de tipo intermedia representando diferentes niveles de control de altitud específicamente de la Altura respecto a un punto de ubicación del hexacóptero. Esta controlará la altitud del dron, asegurando que esta suba o baje a una altura deseada, extendiendo el PIDController que estará vinculado jerárquicamente con el AltRateController.

5. **PosController:** Esta clase es de tipo intermedia representando el control del posicionamiento horizontal del dron en un espacio de dos dimensiones (x, y), además de extender el Controller y estará vinculado al VelController que es la velocidad del hexacóptero.

6. **RateController:** Esta clase se utilizó como una clase especializada para controladores Finos o con más exactitud que trabajan directamente sobre las entradas del hexacóptero, como los motores. Esta clase trabajar bajo la función de AttController(Controlador de la Cinemática) y controlara las velocidades angulares para ajustar la orientación.

7. **AltRateController:** Esta clase se utilizó como una clase especializada para controladores finos o con más exactitud que trabajan directamente sobre las entradas del hexacóptero, como controlador de velocidad angular de los motores. Esta clase trabajar bajo la función de AltController (Controlador de la Altitud) y controlara la velocidad de ascenso o descenso.

8. **VelController:** Esta clase se utilizó como una clase especializada para controladores finos o con más exactitud que trabajan directamente sobre las entradas del hexacóptero, como controlador de velocidad lineal de los motores. Esta clase trabajar bajo la función de PosController (Controlador de posicionamiento) y controlara la velocidad lineal (x, y) para alcanzar una posición deseada.

b. Relaciones de clases de la simulación de control y PID

Las relaciones que más se subdividen es la Herencia (extends), todas las clases intermedias heredan de las clases base (PIDController o Controller). Para las clases especializadas (más bajas en la jerarquía) heredan de las clases intermedias. Se tiene

además conexiones jerárquicas entre cada clase especializada trabaja junto con la clase superior que es conectada y la controla.

3.1.18 Simulación de trayectorias del multirotor (hexacoptero) basado en el uso de librerías "Multirotor" en programación de Python

El modelado de trayectorias en un entorno de pruebas o simulación constituye la base para obtener indicios del comportamiento del sistema, asemejándose a un entorno con características reales. Este modelado es fundamental para la aplicación de trayectorias en pruebas físicas reales del UAV. Se describió previamente la organización del código, utilizando la librería "Multirotor" como herramienta para el estudio de rutas del UAV, empleando programación en Python.

Como primer paso, se inició con la descripción y creación de la clase intermedia llamada "Multirotor". Posteriormente, se definió el dron, que en este caso corresponde a un hexacóptero, considerando las características descritas en el Capítulo I relacionadas con la elección de drones según especificaciones técnicas, en este caso el hexacóptero Tarot 680 Pro.

Se emplearon algunas librerías adicionales para generar gráficas en 3D y realizar análisis de comportamiento del dron, así como para estudiar las características de su vuelo. Las librerías utilizadas se detallan en la Tabla 36.

Tabla 36. Librerías adicionales utilizadas en la simulación de trayectorias de multirotor

Librería	Descripción
Matplotlib	Uso en la creación y personalización de diferentes tipos de gráficos más comunes entre los cuales se utilizó diagramas de líneas, dispersión de puntos.
Numpy	Uso en el cálculo matemática y análisis de datos respectivamente con la cantidad de datos ingresados basado en arreglos.
Json	Uso en la carga de archivos JSON de la conversión de coordenadas geográficas a coordenadas cartesianas.

Librería	Descripción
mpl_toolkits (mplot3d)	Uso en gráficos con trazados 3D simples (Superficies y líneas)

a. Creación del UAV(Hexa rotor)

El UAV (Hexa rotor) se representó mediante la clase multirotor.simulation.Multirotor, que es responsable de los cálculos físicos. El *Multirotor* está parametrizado por VehicleParams y SimulationParams especificando en el apartado anterior algunos de los métodos también aplicados.

La clase VehicleParams de esta propia clase está parametrizada por PropellerParams para cada hélice del UAV que en este caso son seis hélices. Alternativamente, una hélice puede tener MotorParams que describen las propiedades eléctricas. De lo contrario, todos los cambios en la velocidad de la hélice son instantáneos. Se creo un vehículo simple que ignore la dinámica del motor, aquello parámetros se visualizarán en la Figura 161.

```
from multirotor.vehicle import PropellerParams, SimulationParams

# Parámetros del vehículo
pp = PropellerParams(
    k_thrust=0.01,
    k_drag=1e-2,
    moment_of_inertia=1e-3
    )

sp = SimulationParams(
    dt=1e-2  # Paso de tiempo de simulación
    )
```

Figura 161. Parámetros y clases utilizadas en la creación del UAV

En la Ecuación 9 el empuje vertical en Newtons Fz está determinado por el coeficiente de empuje. Para la velocidad de la hélice (radianes por segundo) $\boldsymbol{\omega}$.

$$F_z = k_{thrust} \cdot \omega^2 \quad (9)$$

De manera similar, en la Ecuación 10 el par producido alrededor del eje vertical del multirotor debido a la resistencia de la hélice, Tz está determinado por el coeficiente de resistencia.

$$T_z = k_{drag} \cdot \omega^2 \qquad (10)$$

Las librerías de "Multirotor" tuvieron una función específica para crear el UAV (hexarotor) a partir de geometrías simples, se visualizará el código en concreto en la Figura 162.

```
from multirotor.helpers import vehicle_params_factory
from multirotor.simulation import Multirotor
vp = vehicle_params_factory(
    n=6,          # number of propellers, equally spaced
    m_prop=1e-2,  # mass of propeller / kg
    d_prop=0.3,   # distance of propeller from center of mass /m
    params=pp,    # cada propeller tiene estos parámetros
    m_body=2      # masa del cuerpo del dron
    )
# Inicializar el vehículo y la simulación
vehicle = Multirotor(params=vp, simulation=sp)
```

Figura 162. Parámetros del UAV (Hexa rotor) de las características técnicas de las hélices.

El Multirotor proporciona una serie de métodos para manipular el vehículo. Los dos métodos principales serán "step_dynamics" este método tomará un vector de seis dimensiones (6D) de fuerzas y pares en la carrocería del UAV (Hexarotor) y devolverá el estado de este y el otro método "step_speeds" tomará un vector n dimensiones de velocidades de la hélice y devolverá el estado. Se deberá tener en cuenta el tipo de dato de los vectores debe coincidir con "vehicle.dtype".

El UAV se describe mediante su atributo .*state*, que será una matriz de 12 dimensiones que contiene la posición, la velocidad, la orientación y la velocidad angular. Se podrá acceder a los componentes de estado individuales mediante algunos atributos relevantes del dron, de forma que se indica algunos de los atributos con los métodos aplicados el código se visualiza en la Figura 163, además de los resultados de los estados y propiedades adicionales del UAV como se visualiza en la Figura 164.

```
# Step using forces/torques
state = vehicle.step_dynamics(np.asarray([1,2,vehicle.weight,0,0,1], vehicle.dtype))
# Step using speeds (rad/s)
state = vehicle.step_speeds(np.asarray([100, 100, 100, 100], dtype=vehicle.dtype))

print('Estado del vehiculo:\n', state)
print('\nParametros:')
print(f'{"Position inicial (m)": >32}', vehicle.position)
print(f'{"Velocidad (body-frame) m/s": >32}', vehicle.velocity)
print(f'{"Orientación /deg ": >32}', vehicle.orientation)
print(f'{"Angulo de velocidad /deg": >32}', vehicle.angular_rate)

print('\nPropiedades adicionales:')
print(f'{"Velocidad (inertial-frame) m/s": >32}', vehicle.inertial_velocity)
print(f'{"Aceleración (m)2/s": >32}', vehicle.acceleration)
print(f'{"Velocidad Angular /deg": >32}', vehicle.euler_rate)
```

Figura 163. Parámetros de estado y métodos usados para resultados de estado del UAV y propiedades adicionales.

```
Estado del vehiculo:
 [ 2.000e-04 -2.000e-04  9.200e-03 -1.388e-01  2.594e-01  1.814e+00
  2.117e-01  1.202e-01  2.410e-02  4.203e+01  2.432e+01  7.463e-01]

Parametros:
           Position inicial (m) [ 0.    -0.     0.009]
     Velocidad (body-frame) m/s [-0.139  0.259  1.814]
              Orientación /deg  [0.212 0.12  0.024]
       Angulo de velocidad /deg [42.033 24.321  0.746]

Propiedades adicionales:
  Velocidad (inertial-frame) m/s [ 0.084 -0.126  1.832]
             Aceleración (m)2/s [ 7.000e-03 -4.000e-03  1.844e+02]
         Velocidad Angular /deg [42.739 23.621  5.882]
```

Figura 164. Resultados de Estado del UAV, parámetros técnicos y propiedades adicionales

b. Visualización del UAV (Hexarotor) en un entorno de simulación en 3D

Para la creación de un ambiente en donde se pueda simular las trayectorias del UAV, se utiliza explícitamente la "*multirotor.visualize.VehicleDrawing"* que será un envoltorio alrededor de matplotlib para visualizar interactivamente el vehículo en 3D, la parte del código se utiliza un ciclo for con un rango de muestras para graficar la trayectoria, para aquello en base al método step_dynamics con un vector de 6D antes mencionado, se dibuja en el entorno y se tendrá un indicar de la posición final del UAV el código se visualiza en la Figura 165, la gráfica con la trayectoria del UAV con la posición final de este se podrá visualizar en la Figura 166.

```
from multirotor.visualize import VehicleDrawing
vehicle = Multirotor(params=vp, simulation=sp)
drawing = VehicleDrawing(vehicle, trace=True)

log = DataLog(vehicle)
vehicle.reset()
for j in range(1000):
    vehicle.step_dynamics(np.asarray(
        [0.4, 2 * np.sin(j*np.pi/100), vehicle.params.mass*sp.g +  np.cos(j*2*np.pi/1000),
         0,0,0],
        dtype=vehicle.dtype))
    drawing.axis.set_title(f'pos:{vehicle.position}')
    drawing.update()
```

Figura 165. Código de visualización en un entorno 3D para el UAV

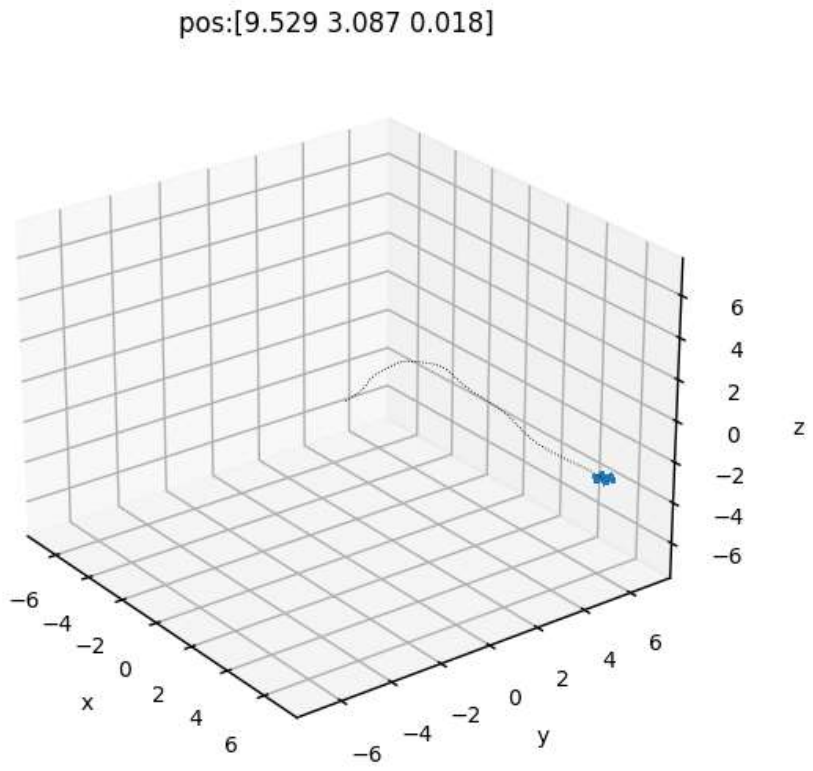


Figura 166. Resultados de la trayectoria del UAV con un indicador de la posición del dron en un entorno 3D

Para el análisis de los parámetros establecidos en el UAV (Hexarotor), con la función "*multirotor.visualize.plot_datalog"* el código se podrá visualizar en la Figura 167, la clase crea la visualización de las gráficas de mediciones de series temporales del UAV, los datos como la posición y la Orientación en cada uno de los ejes y del movimiento

del dron como el cabeceo, giñada y balanceo esto la orientación en grados y la posición en metros.

```
from multirotor.helpers import DataLog
from multirotor.visualize import plot_datalog

vehicle = Multirotor(params=vp, simulation=sp)
drawing = VehicleDrawing(vehicle, trace=True)

log = DataLog(vehicle)
vehicle.reset()
for j in range(1000):
    vehicle.step_dynamics(np.asarray(
        [0.4, 2 * np.sin(j*np.pi/100), vehicle.params.mass*sp.g +  np.cos(j*2*np.pi/1000),
         0,0,0],
        dtype=vehicle.dtype))
    drawing.axis.set_title(f'pos:{vehicle.position}')
    drawing.update()

    log.log()
log.done_logging() # converts to numpy arrays
plot_datalog(log, figsize=(6,4));
```

Figura 167. Función para la gráfica de mediciones temporales en base a la trayectoria del UAV

Las gráficas presentadas en la Figura 168, analizan el comportamiento dinámico de un dron durante su trayectoria, evidenciando cómo se controlan la posición, orientación y velocidades en el espacio. En la primera gráfica, la posición en los ejes x, y, z, muestra un movimiento sostenido hacia adelante en x, correcciones oscilatorias en altitud (z) y ligeros ajustes laterales (y), mientras que las orientaciones angulares (roll, pitch, yaw) permanecen en valores bajos, indicando maniobras suaves y estabilidad. La segunda gráfica confirma este análisis mostrando un incremento constante en la velocidad hacia adelante (x) y patrones oscilatorios en las velocidades y e z, reflejando ajustes dinámicos en altura y movimientos laterales. Finalmente, la gráfica inferior en el plano XY detalla la trayectoria real del dron, con movimientos lineales intercalados con ligeros desvíos que indican correcciones activas para mantenerse dentro de la ruta objetivo. En conjunto, las gráficas validan la estabilidad del dron y su capacidad para realizar ajustes precisos mientras sigue una trayectoria planificada, evidenciando un sistema de control efectivo y bien diseñado.

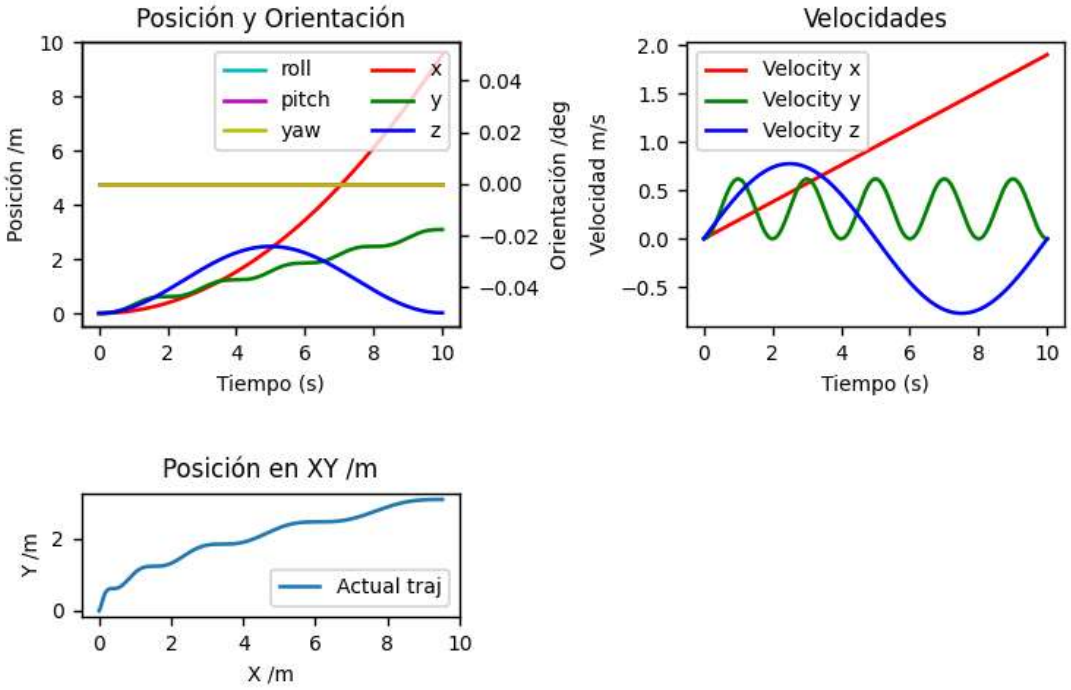


Figura 168. Gráficas de mediciones de series temporales de algunos parámetros de vuelo del UAV respecto a una trayectoria.

c. Control del UAV en un entorno de simulación

El control del UAV, con lleva con la incorporación de más parámetros como el PID del UAV para un vuelo óptimo los cuales en el momento de tener pruebas en un ambiente real es fundamental tomar en cuenta ya que el modelado de acuerdo en un ambiente de simulación es ideal y en un ambiente controlador, para ello la librería de "Multirotor" ofrece una clase "*multirotor.controller.Controller"*, que utiliza el control PID para navegar, este controlador se utiliza controlador PID en cascada que realizará un seguimiento de la posición y la guiñada, y proporciona el empuje y los momentos necesarios, se describe una variable principal "ctrl" la cuál toma como método de clase make_for que es el controlador que solo utilizará el parámetro P (Proporcional).

Un controlador tiene un método step(), que toma la posición de referencia y la guiñada, y genera la dinámica necesaria para lograrlo este viene dado por la Ecuación 11.

$$F_z, \tau_x, \tau_y, \tau_z = step(x, y, z, \psi) \qquad (11)$$

Una vez se detalló el controlador la cual genera la dinámica, se puede variar los valores en coordenadas de acuerdo con un vector con los valores de las variables de los ejes x, y, z y un ángulo que se representa el giro del UAV, así para después mapear los valores de acuerdo con una iteración dentro de un condicional para poder visualizar la matriz como se indica en el código de programación de la Figura 169, y se indicará los resultados de esta variable como en la Figura 170.

```
dynamics = ctrl.step(reference=[20,0,1,0], persist=False)
print('\nDinámica realizada(Torque y Fuerzas):')
print(f'{"F_z=%.3f, T_x==%.3f, T_y=%.3f, T_z=%.3f"}' % (dynamics[0], dynamics[1], dynamics[2], dynamics[3]))
```

Figura 169. Método para la cinemática del UAV con el objeto dynamics

```
Dinámica realizada(Torque y Fuerzas):
F_z=22.269, T_x==0.000, T_y=0.002, T_z=0.000
```

Figura 170. Resultados de la variable dynamics

Entonces después de tener la dinámica prescrita, se deberá convertirla en velocidades prescritas (radianes/s) para las hélices del UAV (Hexa rotor) que en este son 6. En esta parte entrará en juego la asignación de control esto basándose de la dinámica anterior como se visualiza en la Figura 171, y se indicará los resultados de esta variable en la Figura 172.

```
speeds = vehicle.allocate_control(dynamics[0], dynamics[1:6])
print('\nVelocidades Preescritas (rad/s):')
print(speeds)
```

Figura 171. Objeto de Velocidades del UAV (Hexarotor) de acuerdo con la dinámica

```
Velocidades Preescritas (rad/s):
[19.26  19.265 19.27  19.27  19.265 19.26 ]
```

Figura 172. Resultados de la variable speeds

Así, como último paso aplicar todas las velocidades anteriores al UAV (Hexa rotor) esto teniendo en cuenta que se añade otra variable indicándola como “state”, de estado

para indicarlas por consola como se visualiza en la Figura 173, y los resultados de esta se visualizarán en la Figura 174.

```
state = vehicle.step_speeds(speeds)
print('\nEstado de las velocidades al aplicar el vehiculo:')
print(state)
```

Figura 173. Objeto “state” para indicar el estado de las velocidades aplicadas al UAV (hexarotor)

```
[ 1.005e+01  3.118e+00  1.420e-02  2.000e+00 -1.000e-04  5.010e-02
  0.000e+00  0.000e+00  0.000e+00  1.700e-03  1.700e-03  0.000e+00]
```

Figura 174. Resultados de la velocidad aplicadas al UAV (Hexarotor)

Para visualizar las trayectorias trazadas una vez aplicada el control de acuerdo a la dinámica y velocidades del UAV (Hexarotor), se utiliza la librería VehicleDrawing esto para trazar la ruta en un entorno 3D con el UAV, la base es el condicional el cuál mapea en un rango de valores e itera los mismos para trazar la ruta el código de este se visualizara en la Figura 175, además se traza graficas ahora de las mediciones de series temporales de acuerdo con el control del UAV antes mencionado de la Posición y Orientación, velocidades, controles asignados de la dinámica del UAV Figura 176.

```
vehicle.reset()
ctrl.reset()
drawing = VehicleDrawing(
    vehicle, trace=True, body_axes=True,
    make_fig_kwargs={'xlim':(-1,10), 'ylim':(-3,3), 'zlim':(-1,2)}
)

log = DataLog(vehicle, ctrl)

for i in range(500):
    dynamics = ctrl.step(reference= [20,5,1,0.6], persist=True)
    speeds = vehicle.allocate_control(dynamics[0], dynamics[1:4])
    state = vehicle.step_speeds(speeds)
    drawing.update()
    log.log()
log.done_logging()
drawing.axis.view_init(elev=30, azim=-100)
```

Figura 175. Condicional y variables utilizadas para graficar los plots y escenario 3D

Por último, la trayectoria prescrita en los ejes XY en estas gráficas como añadidura se trazan los valores deseados a los valores simulados los cuales se adaptan con la

trayectoria impuesta, los resultados de lo antes mencionado se visualizarán en la Figura 176, donde gráficas analizan cómo el dron ajusta su posición, velocidad y dinámica para seguir una trayectoria planificada de manera eficiente. La posición en x muestra un avance continuo, mientras y e z presentan correcciones dinámicas asociadas a ajustes en altura y lateralidad. Las velocidades y orientaciones evidencian un control estable, con variaciones controladas para maniobras precisas. La dinámica aplicada por el controlador, a través de empuje y torques, garantiza estabilidad y adaptación ante desviaciones, logrando que el dron siga la ruta prescrita con precisión y robustez.

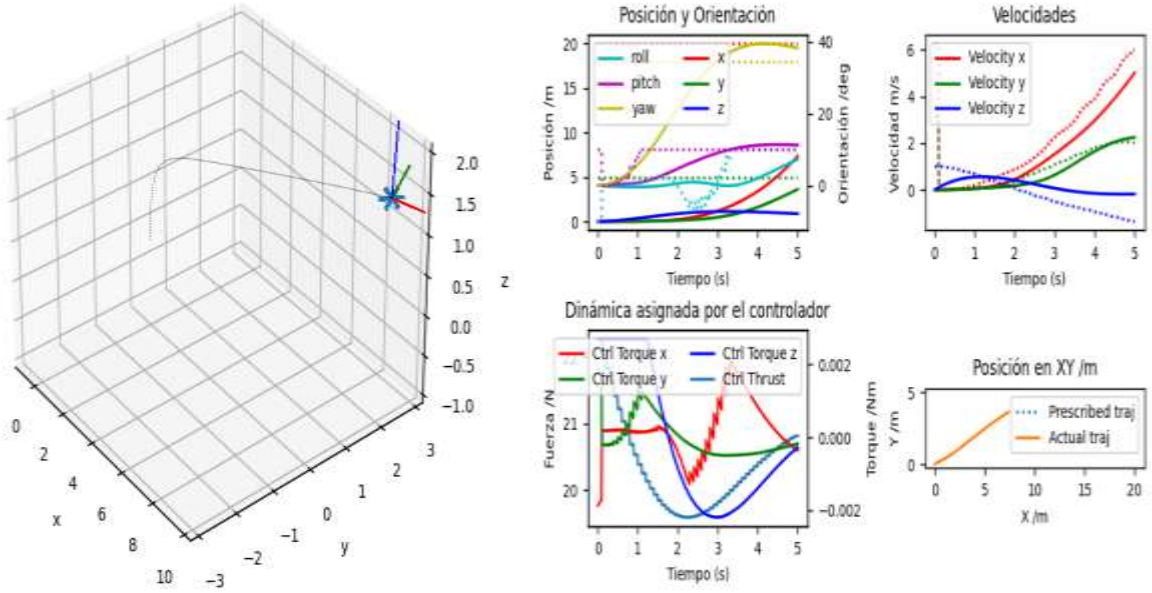


Figura 176. Resultados de la simulación de trayectorias en un entorno 3D y graficas de series temporales del UAV (Hexa rotor)

d. Optimización del control del PID del UAV (Hexarotor)

Para la optimización del control PID se basó en ajustar diferentes parámetros que se adapten de forma adecuada al comportamiento del UAV en la simulación de trayectorias, es una tarea compleja el encontrar los valores para un vuelo óptimo el uso de un enfoque basado en simulación permite ajustar los parámetros sin necesidad de realizar pruebas reales, lo que resulta necesario para entender el comportamiento del UAV (Hexarotor) y evitar accidentes. Además, la visualización de los resultados ayuda

a entender cómo los parámetros interactúan entre sí y afectan el comportamiento del sistema.

Para aquello se utiliza la función de la librería de multirotor conocida como "*multirotor.optimize*" proporcionará una función útil denominada optimized() que busco los mejores parámetros. La búsqueda de parámetros se realizó mediante la biblioteca optuna, que se instala como una dependencia este código de visualiza en la Figura 177.

```
from optuna.visualization import plot_parallel_coordinate # requires plotly
from optuna.visualization.matplotlib import plot_parallel_coordinate # requires matplotlib
from multirotor.optimize import optimize, run_sim, apply_params

# Inicializar el vehículo y la simulación
vehicle = Multirotor(params=vp, simulation=sp)
ctrl = Controller.make_for(vehicle)
study = optimize(vp, sp, ctrl, ntrials=100)


plot_parallel_coordinate(study)

apply_params(ctrl, study.best_params);

pprint(ctrl.get_params())
```

Figura 177. Función optimize en un objeto study para aplicación de parámetros del control PID óptimo.

Los resultados de la optimización del controlador se ajustan a parámetros como características de las hélices, como el número de hélices, la masa de cada una de ellas, la distancia respecto a cada brazo, el centro de masa, constantes de thrust (empuje) y drag (arrastre), el momento de inercia, además de la masa del cuerpo del dron del UAV (Hexarotor), realiza múltiples pruebas ajustándolos y evaluándolos al desempeño de este, esto se almacena en un objeto study que contendrá la mejor configuración rebuscada entre todos los intentos anteriores, los valores aplicados se visualizan en la Figura 178 de todos los valores rebuscados de los 100 intentos antes mencionados se visualizan en la Figura 179.

```
{'ctrl_a': {'k_d': array([208.764, 208.764,  21.962]),
            'k_i': array([1.202, 1.202, 0.194]),
            'k_p': array([44.687, 44.687,  1.784]),
            'max_acceleration': 0.2,
            'max_err_i': array(1.),
            'max_jerk': 100.0,
            'square_root_scaling': False},
 'ctrl_p': {'k_d': array([41.46, 41.46]),
            'k_i': array([8.115, 8.115]),
            'k_p': array([9.775, 9.775]),
            'leashing': True,
            'max_acceleration': 3.0,
            'max_err_i': array(1.),
            'max_jerk': 100.0,
            'max_velocity': 7.0,
            'square_root_scaling': True},
 'ctrl_r': {'k_d': array([29.523, 29.523, 17.64 ]),
            'k_i': array([7.343, 7.343, 0.285]),
            'k_p': array([18.526, 18.526,  3.21 ]),
            'max_acceleration': array([4.439, 4.439, 0.841]),
            'max_err_i': array(1.),
            'max_tilt': 0.17453292519943295},
 'ctrl_vz': {'k_d': 242.245389433615,
             'k_i': 8.488711710309632,
             'k_p': 45.331798118680894,
             'max_err_i': array(1.)},
 'ctrl_z': {'k_d': array([0.]),
            'k_i': array([0.]),
            'k_p': array(1.),
            'max_err_i': array(1.),
            'max_velocity': 5},
 'feedforward_weight': 0.0}
```


Figura 178. Resultados de valores optimizados del controlador PID

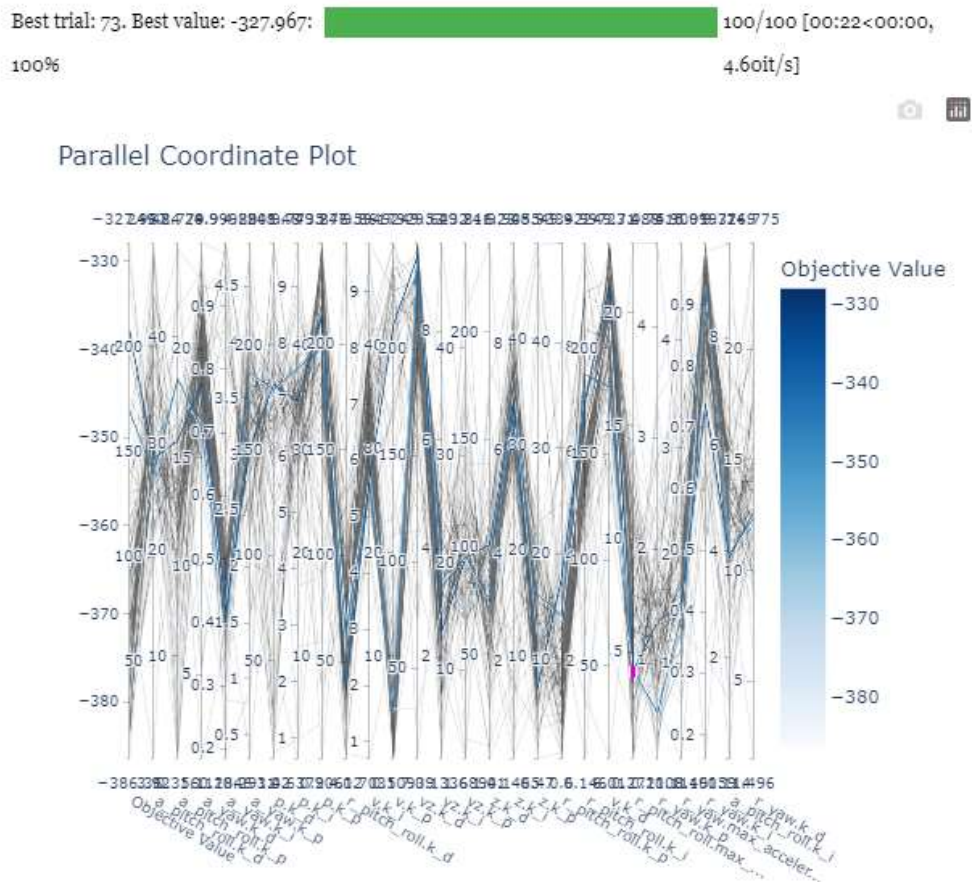


Figura 179. Grafica de todos los parámetros de optimización del controlador de PID

e. Trayectorias completas del UAV (Hexarotor)

Una vez se implementó la creación del UAV(Hexarotor) con todos los parámetros de características físicas del dron, para pasar a la visualización del entorno 3D de simulación del UAV(Hexarotor) y graficas de análisis del comportamiento de este en trayectorias predispuestas, también fue necesario el incorporar controladores y optimización del control PID del dron para la regulación del comportamiento del dron ya mencionado permitiendo que pueda seguir una secuencia de puntos con un nivel de proximidad y resolución específicas, para incorporar trayectorias en base de matrices que fue posible el incorporar diferentes rutas, esto utilizando las librerías del "*multirotor.trajectories*" este define la clase Trajectory la cual podrá tomar una lista de puntos de referencia y dividirlos en segmentos más pequeños para el controlador siendo más preciso y con una resolución elevada, este código se visualizará en la Figura 180, entonces las trayectorias trazadas por el UAV se visualizan en la Figura 181.

```
from multirotor.trajectories import Trajectory

ctrl = Controller.make_for(vehicle)   # Controlador y trayectoria
trayectoria = interpolated_points  # Asignación de la variable trayectoria para puntos interpolados
traj = Trajectory(vehicle, points=trayectoria, proximity=0.1, resolution=2.5)  #Resolución y proximidad de trayectorias

# Recorrer la trayectoria
for i, (ref, _) in enumerate(traj):
    dynamics = ctrl.step(reference=[*ref, 0])  # yaw=0
    speeds = vehicle.allocate_control(dynamics[0], dynamics[1:4])
    vehicle.step_speeds(speeds)

    drawing.axis.set_title(f'Posición actual:{vehicle.position}') # Actualizar la visualización
    drawing.update() # Actualizar la visualización
    # Agregar posición actual a la trayectoria
    trajectory_x.append(vehicle.position[0])
    trajectory_y.append(vehicle.position[1])
    trajectory_z.append(vehicle.position[2])

    # Graficar la trayectoria completa acumulada
    ax.cla()
    ax.plot(trajectory_x, trajectory_y, trajectory_z, label='Trayectoria')  # Línea de trayectoria
    ax.scatter(trajectory_x[-1], trajectory_y[-1], trajectory_z[-1], color="red", label="Dron")  # Posición actual
    ax.legend()

    ax.set_title(f'Posición actual: {vehicle.position}')  # Asegurar que el título muestre la posición actual del dron
    plt.pause(0.01)  # Aumentar la velocidad de la simulación

    log.log()
    if i == 1050:
        break
```

Figura 180. Uso de la clase Trajectory para trayectorias del UAV (Hexarotor)

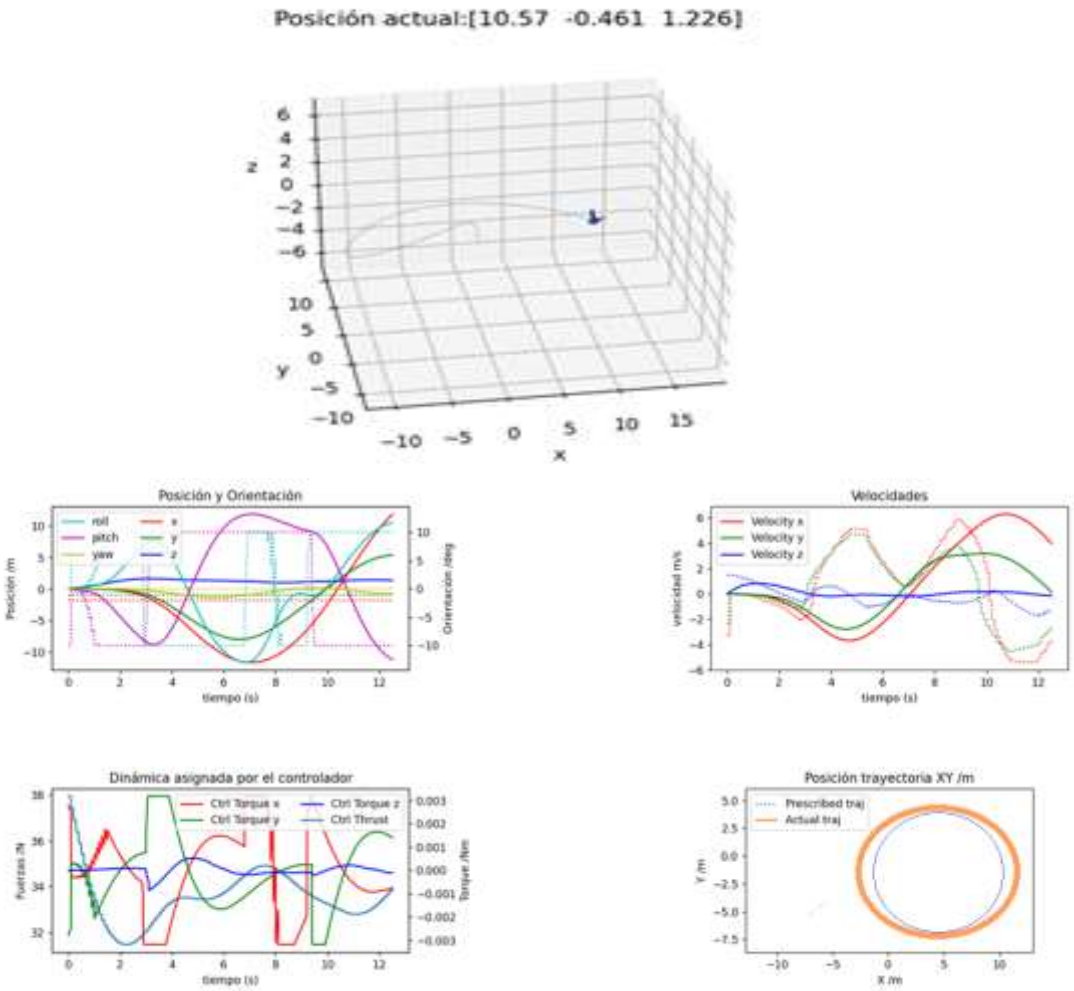


Figura 181. Resultados de simulación de trayectorias del UAV (Hexarotor) aplicando la clase Trajectory.

Algunas funciones adicionales para la simulación esto específicamente a la planificación de misiones para las trayectorias del UAV (Hexarotor), se utilizó la lectura de datos de un archivo en formato JSON esto haciendo referencia del sistema de georreferencia para cargarlo en el UAV, la base antes explicada del funcionamiento del sistema es que a partir de la ruta planificada en coordenadas geográficas convertirlas a un modelo de coordenadas cartesianas esto utilizando el sistema ECEF(Earth-Centered, Earth-Fixed), entonces leer estos datos ya anteriormente procesados se utilizó una función de carga de coordenadas explícitamente para lo antes mencionado, otra añadidura es la interpolación de puntos la funcionalidad de la misma es generar puntos intermedios entre los puntos originales para suavizar la trayectoria del UAV (Hexarotor), se hiso dividiendo la distancia entre puntos en segmentos más

pequeños, basados en una resolución especificada el código de las funciones mencionadas se visualiza en la Figura 182.

```python
# Función para cargar puntos desde JSON
def cargar_coordenadas_json(file_path="trayectoria_cartesiana.json"):
    with open(file_path, 'r') as f:
        return json.load(f)

# Función de Interpolación de puntos
def interpolate_points(points, resolution):
    interpolated_points = []
    for i in range(len(points) - 1):
        start = np.array(points[i])
        end = np.array(points[i + 1])
        dist = np.linalg.norm(end - start)
        num_segments = max(1, int(dist / resolution))
        for t in np.linspace(0, 1, num_segments):
            interpolated_points.append((1 - t) * start + t * end)
    return np.array(interpolated_points)
```

Figura 182. Funciones Cargar coordenadas json e Interpolación de puntos para trayectorias del UAV (Hexarotor)

También se utilizó otra grafica de simulación en un entorno 3D, de forma que se visualizó en una escala mucho más pequeña y que siga la ruta cargada al UAV (Hexarotor) visualizando el dron con un indicador (punto rojo), con un trazo de color de azul para la trayectoria recorrida la parte del código es añadido en la condición del mapeo de puntos de acuerdo con la iteración de puntos esto se visualiza en la Figura 183, y la gráfica se puede se visualizará en la Figura 184.

```
# Configuración inicial
fig = plt.figure()
ax = fig.add_subplot(projection='3d')
ax.set_xlabel('x(m)')
ax.set_ylabel('y(m)')
ax.set_zlabel('z(m)')

# Límites de los ejes
ax.set_xlim(-50, 50)
ax.set_ylim(-50, 50)
ax.set_zlim(-50, 50)

# Graficar la trayectoria completa acumulada
ax.cla()
ax.plot(trajectory_x, trajectory_y, trajectory_z, label='Trayectoria')  # Línea de trayectoria
ax.scatter(trajectory_x[-1], trajectory_y[-1], trajectory_z[-1], color="red", label="Dron")  # Posición actual
ax.legend()

ax.set_title(f"Posición actual: {vehicle.position}")  # Asegurar que el título muestre la posición actual del dron
plt.pause(0.01)  # Aumentar la velocidad de la simulación

log.done_logging()  # Convertir a numpy arrays
drawing.axis.view_init(elev=30, azim=-100)

plot_datalog(log, figsize=(6, 4))
plt.show()
```

Figura 183. Código para segunda gráfica de trayectorias del UAV (Hexarotor) en 3D.

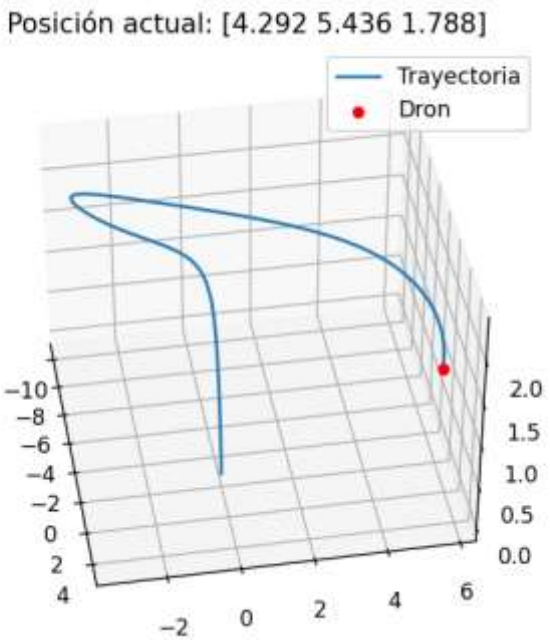


Figura 184. Resultado de la simulación trayectorias del UAV (Hexarotor) en una segunda gráfica con una escala menor.

3.2 Resultados

3.2.1 Pruebas de validación del tiempo de autonomía de vuelo del UAV

De acuerdo con los estudios mencionados respecto al tiempo de vuelo total este cálculo se indica en el apartado de cálculos de consumo energético para tiempo de vuelo del dron en d, entonces el tiempo real total de vuelo es de 19.30 minutos considerando lo antes mencionado se realizó pruebas de vuelo tomando en cuenta cada lugar para la investigación tanto en los talleres tecnológicos FISEI, así como en el garaje del campus huachi de la UTA en donde se limitó el terreno y de acuerdo a las irregularidades y obstáculos del terreno; para aquello se toma en cuenta las trayectorias mencionadas en el apartado 3.1.3.

Se especifican las pruebas realizadas consideraron los datos de la misión establecida respecto a los parámetros como el tipo de trayectoria, lugar de limitación, número de puntos en la ruta, el número de intentos por prueba diarios y la operación aérea a la cual va dedica la misión, esto se muestra en la Tabla 37.

Tabla 37. Parámetros de las pruebas de vuelo del UAV de la misión planificada

Prueba	Tipo de trayectoria	Lugar	Número de puntos en la ruta	Número de intentos Diarios	Descripción
1	Circular	Garage campus Huachi UTA	13	1	Operación de vigilancia de grandes eventos y multitudes
2	Cuadrada	Garage campus Huachi UTA	5	3	Operación de fotografía aérea
3	Patron Rectangular	Garage campus Huachi UTA	11	1	Operación de patrullaje de perímetros y áreas extensas
4	Triangular	Garage campus Huachi UTA	6	4	Operación de fotografía aérea
5	Zig Zag	Garage campus Huachi UTA	5	2	Operación de patrullaje de perímetros y áreas extensas
6	Recta	Talleres tecnológicos	6	3	Operación de fotografía aérea
7	Cuadrada	Talleres tecnológicos	6	2	Operación de fotografía aérea
8	Rombo	Talleres tecnológicos	5	5	Operación de fotografía aérea

9	Zig Zag	Talleres tecnológicos	3	3	Operación de fotografía y videografía aérea

Algunas consideraciones especificas además de los parámetros anteriores para las pruebas de vuelo es la velocidad del dron establecido en la misión para los waypoints configurados y la velocidad del viento para tener en consideración si se puede volar el dron entonces dentro del campus huachi de la UTA esto es medido con la estación meteorológica existente en los talleres tecnológicos el dato fue de 2.54813 metros/segundo y mediante una app móvil el dato fue 1.77 m/s mínimo y máximo de 2.68 m/s estos datos se recopila de una aplicación móvil y una página web que se visualizan en la Figura 185.

También se consideran parámetros como la velocidad con la que va el dron y respecto a que la altura en un rango específico, entonces respecto a estas condiciones se midió el tiempo de cada una de las pruebas con un cronómetro digital del dispositivo móvil como se visualiza en la Figura 186, los datos de cada una de las pruebas se muestran en la Tabla 38.

Tabla 38. Mediciones de tiempo respecto a parámetros establecidos de las pruebas de vuelo

Prueba	**Tiempo cronometrado (minutos)**	**Velocidad del dron (m/s)**	**Rango de Altitud (metros)**	**Observaciones**
1	12,12	2	Mín: 20 m Max: 23 m	Vuelo estable en cada punto sin cambios bruscos
2	Intento 1: 4,23 Intento 2: 4,18 Intento 3: 4,12	3	Mín: 4 m Max: 25 m	Vuelo estable, giros un poco bruscos
3	13,47	5	Mín: 10 m Max: 30 m	Vuelo estable
4	Intento 1: 3,23 Intento 2: 3,16 Intento 3: 3,26 Intento 4: 3,08	2,5	Mín: 10 m Max: 25 m	Vuelo estable
5	Intento 1: 7,15 Intento 2: 7,08	5	Mín: 18 m Max: 26 m	Vuelo regular cambios bruscos en giros
6	Intento 1: 3,43 Intento 2: 3,30 Intento 3: 3,22	7,5	Mín: 9 m Max: 16 m	Vuelo regular vibraciónes medias del marco del dron
7	Intento 2: 6,33 Intento 2: 6,22	2,5	Mín: 20 m Max: 21 m	Vuelo Estable

Prueba	Tiempo cronometrado (minutos)	Velocidad del dron (m/s)	Rango de Altitud (metros)	Observaciones
8	Intento 1: 2,05 Intento 2: 2,04 Intento 3: 2,00 Intento 4: 2,07 Intento 5: 2,09	10	Mín: 15 m Max: 21 m	Vuelo inestable demasiadas vibraciones del marco del dron
9	Intento 1: 3,55 Intento 2: 3,53 Intento 3: 3,52	1,5	Mín: 30 m Max: 35 m	Vuelo estable



Figura 185. Datos de la velocidad del viento en los talleres tecnológicos FISEI y el campus Huachi de la UTA.

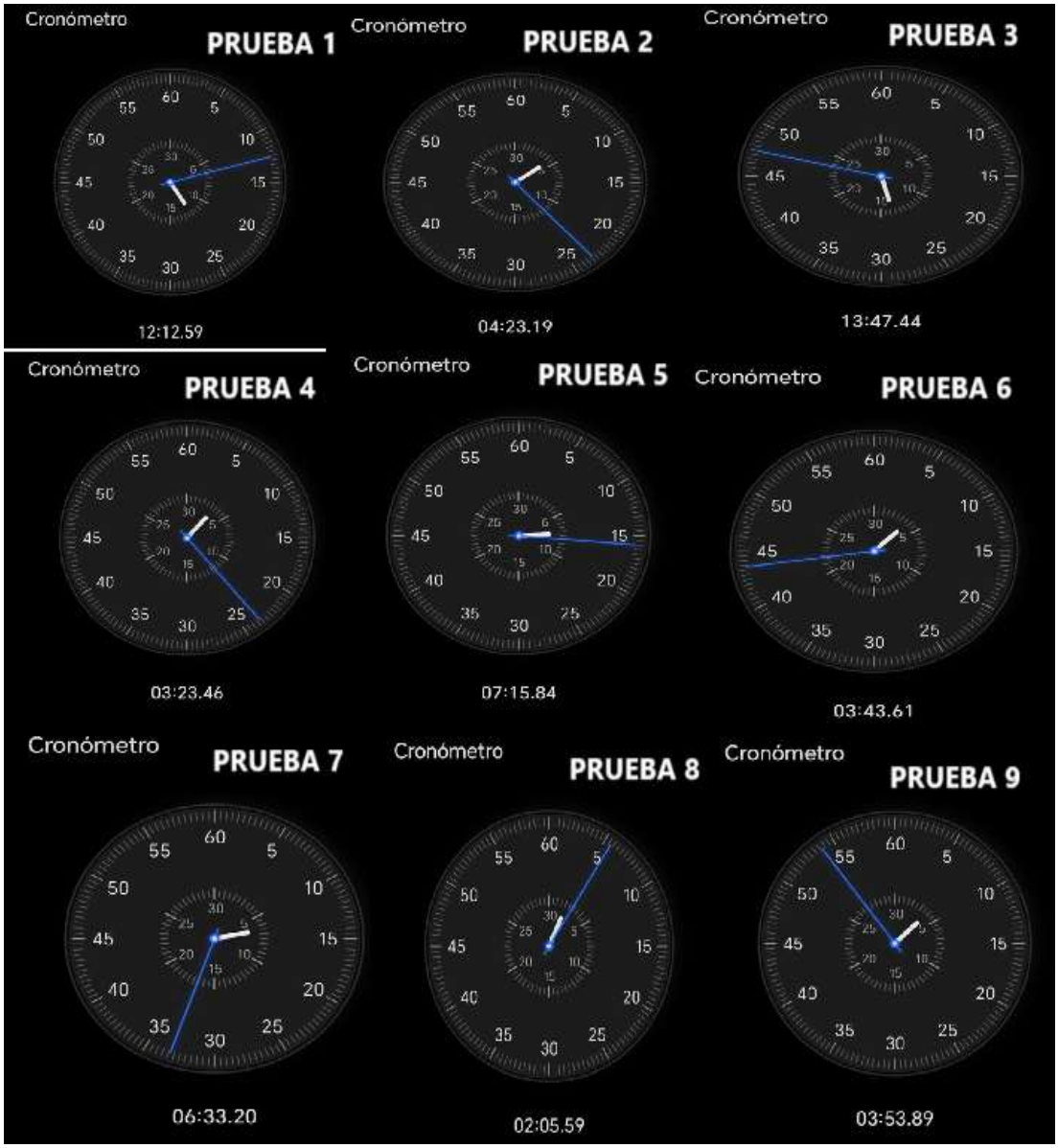


Figura 186. Mediciones de tiempo de las pruebas de vuelo.

Los resultados obtenidos de un número de intentos total por cada prueba es de 21 vuelos, validando que a mayor número de puntos georreferenciados en cada misión y respecto a la operación aérea que realizó es mayor tiempo de autonomía de vuelo mientras que a menor número de puntos georreferenciados igualmente dependiendo de la operación aérea por misión es menor tiempo de autonomía de vuelo como se puede visualizar en la gráfica de la Figura 187 en la cual se visualiza que prueba tuvo un mayor tiempo de vuelo, al a ver incrementado la velocidad del dron existió una disminución de tiempo de vuelo.

Mediante el software Mission Planner al finalizar cada prueba se pudo observar el comportamiento del dron, tomando como un diseño del dron en la simulación de color verde que se visualiza en el mapa y se visualizó el trazo de color morado completo por cada ruta respectivamente de cada prueba. Se tomo en consideración con el lugar que se realizó como los talleres tecnológicos FISEI como se visualiza en la Figura 188 y en el garaje del campus Huachi de la UTA como se visualiza en la Figura 189, la validación de que se realizó las pruebas de vuelo se indica tanto en el Anexo D y Anexo E.

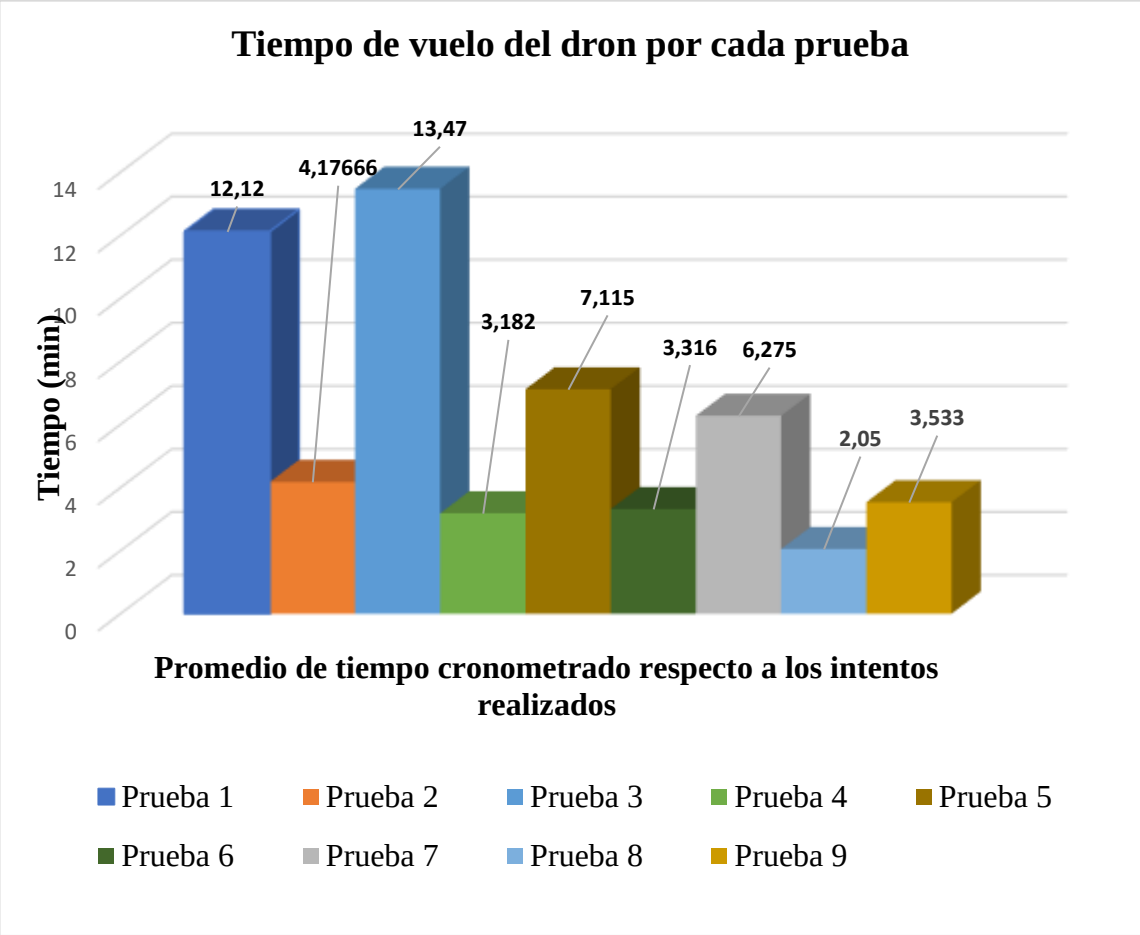


Figura 187. Gráfico de barras del tiempo del vuelo del dron respecto a cada prueba realizada.

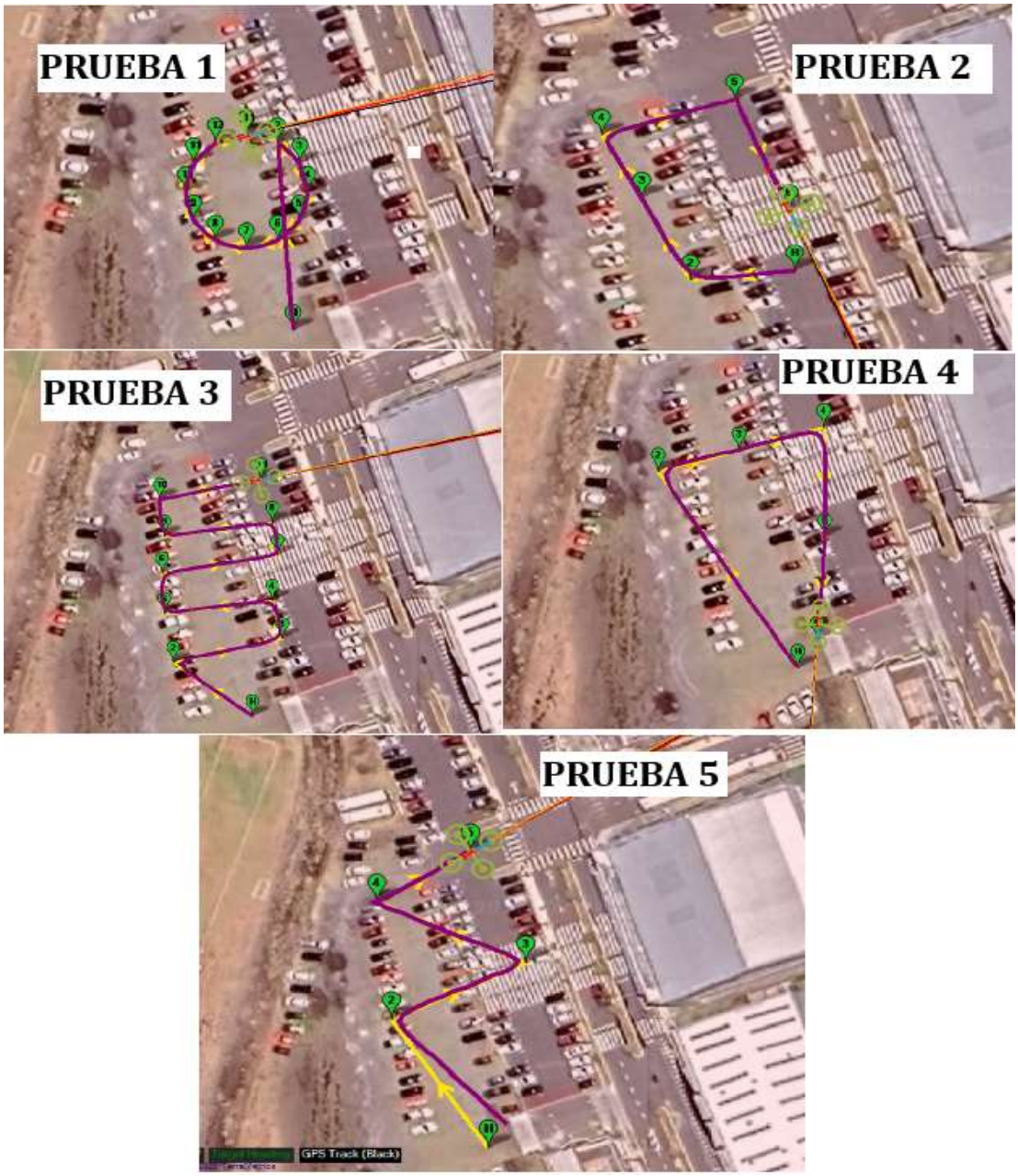


Figura 188. Pruebas realizadas en los talleres tecnológicos FISEI

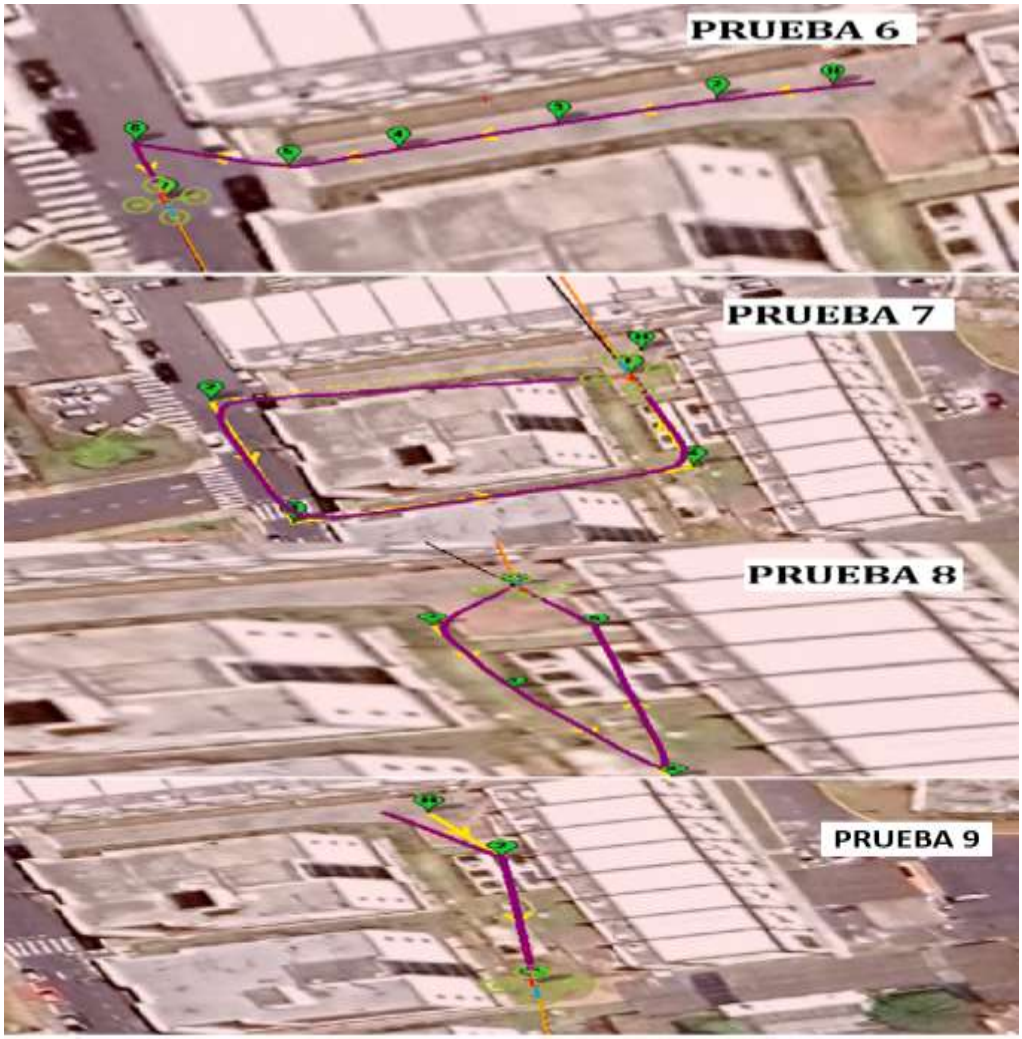


Figura 189. Pruebas realizadas en el garaje del campus Huachi de la UTA

Para verificar la altitud respecto a cada prueba realizada se verifico en la aplicación de Google earth respecto al archivo KML para simular por la aplicación mencionada con mapa satelital de Google, para cada una de las pruebas esto cada lugar limitado tanto en los talleres tecnológicos FISEI como se visualiza en la Figura 190, y en el garaje del campus Huachi de la UTA como se visualiza en la Figura 191.

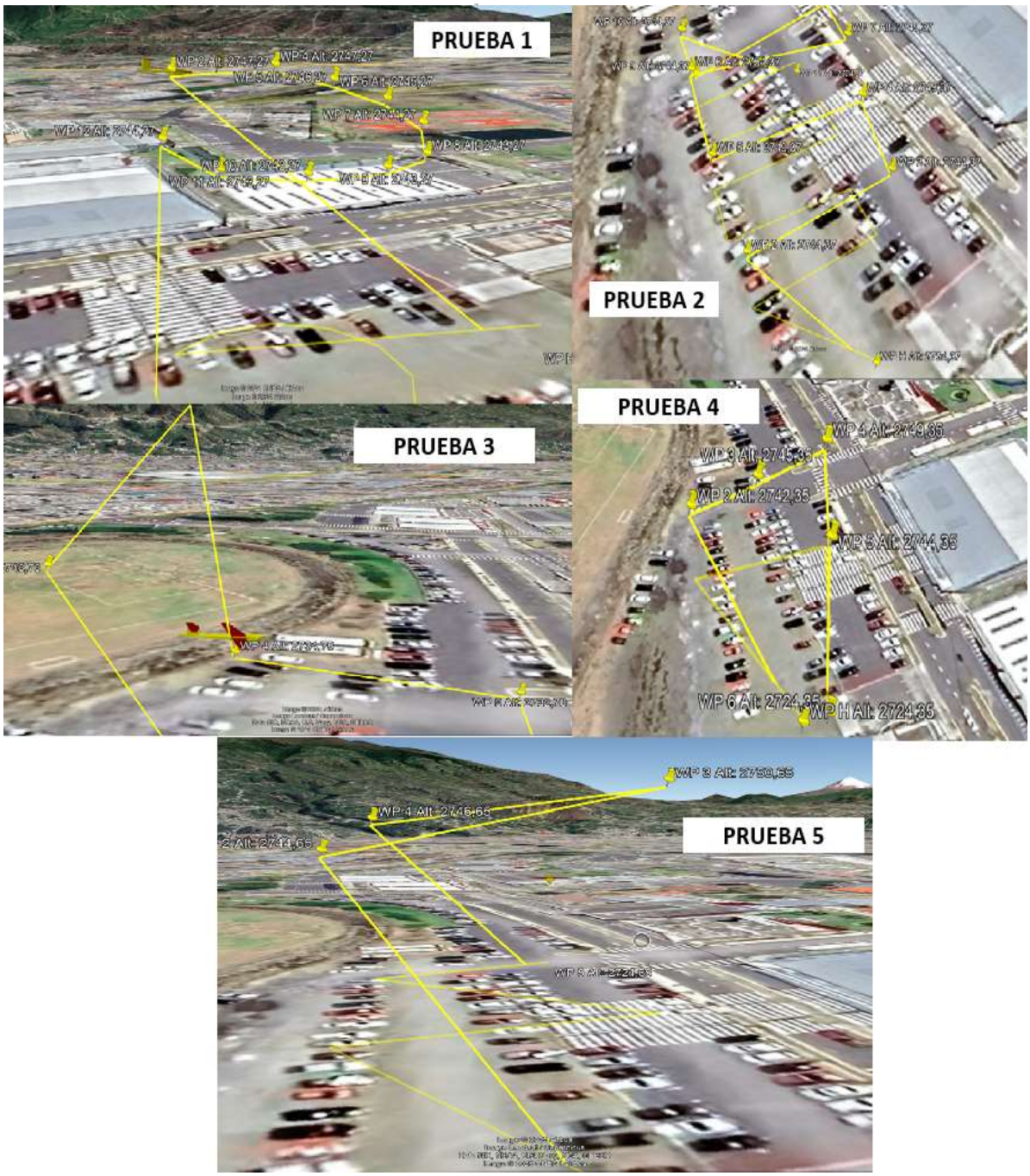


Figura 190. Pruebas realizadas en Google earth de los talleres tecnológicos de la FISEI.



Figura 191. Pruebas realizadas en Google earth del garaje del campus Huachi de la UTA.

3.2.2 Pruebas de funcionamiento de planificación de misiones con el programa de georreferencia y simulación de trayectorias del UAV.

En esta investigación se realizó un sistema de georreferencia para un sistema de navegación autónoma de un hexacóptero para la validación se realizó una comparación entre la planificación de rutas mediante el software Mission Planner, el sistema de georreferencia en Python y una app móvil “Coordenadas GPS” los datos recopilados por la aplicación se visualizan en la Figura 192.



Figura 192. Aplicación móvil para georreferenciar puntos geoespaciales.

Se tomó en cuenta la precisión de los puntos de coordenadas geográficas que son esenciales a la hora de la planificación de rutas o misiones así también las distancias entre cada punto, entonces estos sistemas de acuerdo al dispositivo de medición GPS se indicar el sistema, mapa y módulo con el cual trabaja esto se visualiza en la Tabla 39, además de algunos parámetros de los sistemas respecto al módulo GPS que tiene el dron, el ordenador y el teléfono móvil o la forma cual se consiguen los datos de los puntos georreferenciados.

Tabla 39. Parámetros de sistemas de georreferencia

Sistema	**Mapas**	**Módulo**
Planificador de misiones de Mission Planner	Google earth	u-blox M8
Sistema de georreferencia	Open Street Map	12th Gen Intel(R) Core (TM) i7-12700H / Algoritmo con librerías de geopy
App Movil “Coordenadas GPS”	USGS Topo	Exynos 850

Se realizó pruebas de planificación de rutas estableciendo los puntos georreferenciados de los datos que arroja cada uno de los sistemas para después realizar la comparación de los puntos georreferenciados para las rutas del dron, se escogen las pruebas con una

cantidad alta, media y baja de puntos geoespaciales, se tomó en consideración el cálculo del error absoluto utilizando la ecuación 12, error relativo utilizando la ecuación 13 y el error cuadrático medio (RMSE) utilizando la ecuación 14, entonces esto respecto a cada prueba para el cálculo de la eficiencia de cada uno de los sistemas.

$$E_{abs} = \sqrt{(lat_{real} - lat_{medido})^2 + (lon_{real} - lon_{medido})^2} \quad (12)$$

$$E_{real} = \frac{E_{abs}}{Distancia\ Total} \quad (13)$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(E_{abs,i})^2} \quad (14)$$

Para las medidas tomadas de latitud y longitud real se recogió datos de la página de geoportal de Ecuador, utilizando los datos de una estación física ABEC de la ciudad de Ambato para datos geoespaciales a partir de la herramienta de visor geográfico del Instituto Geográfico Militar de Ecuador, de forma que el mapa que se usó se visualiza en la Figura 193.

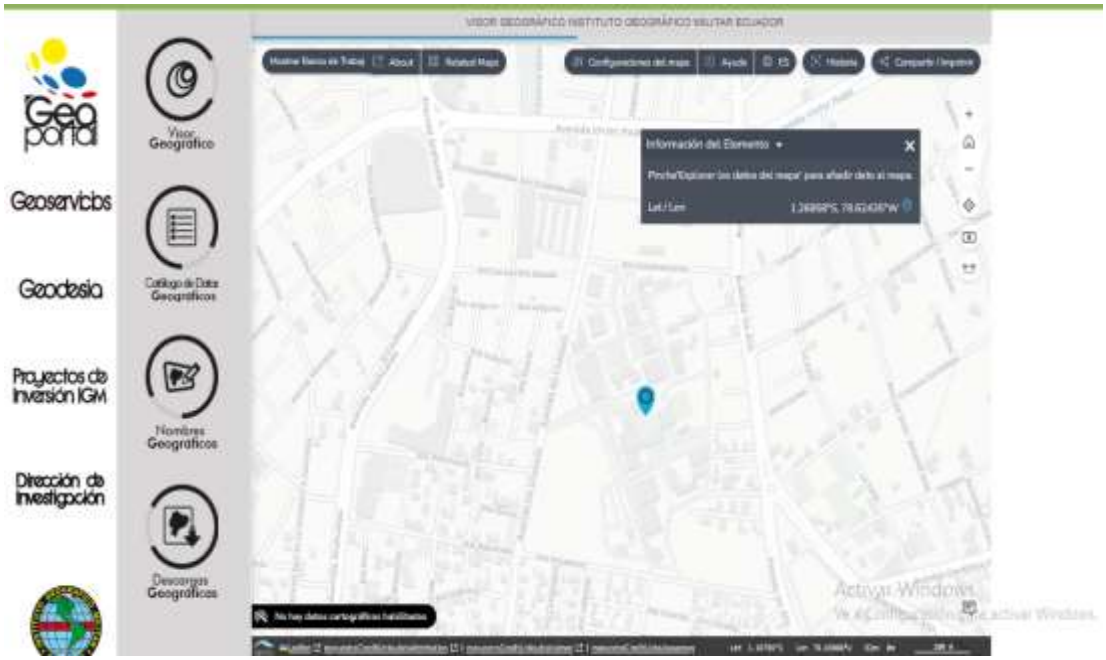


Figura 193. Visor geográfico del Instituto Geográfico militar de Ecuador [64]

a. Prueba con cantidad alta de puntos georreferenciados.

Para la prueba con cantidad alta de puntos georreferenciados se consideró la prueba 1 se tomó en cuenta los parámetros dados de la prueba que se indican en la Tabla 37, se

comparó cada punto de la prueba y se los compara obteniendo los resultados de la Tabla 40.

Tabla 40. Comparación de puntos georreferenciados con cálculo de errores de la prueba 1

N° PUNTOS	SISTEMAS DE GEORREFERENCIA	Latitud medida (°)	Longitud medida (°)	Latitud real (°)	Longitud medida (°)	Distancia de punto a punto (m)	Error Absoluto (m)	Error Relativo (%)	Error Cuadrático medio RMSE (m)
1	APP Móvil	-1,2669209	-78,6238432	-1,2669215	-78,6238438	0,5	$8{,}49\text{x}10^{-7}$	$1{,}69706\ \text{x}10^{-4}$	$4{,}05681\ \text{x}10^{-6}$
	Programa en Python	-1,2669213	-78,6238436				$2{,}83\ \text{x}10^{-7}$	$5{,}65685\ \text{x}10^{-5}$	$1{,}54397\ \text{x}10^{-6}$
	Mission Planner	-1,2669210	-78,6238433				$7{,}07107\ \text{x}10^{-7}$	$1{,}41421\ \text{x}10^{-4}$	$3{,}42592\ \text{x}10^{-6}$
2	APP Móvil	-1,2697177	-78,6248175	-1,2697184	-78,6248182	5,16	$9{,}89949\ \text{x}10^{-7}$	$1{,}91851\ \text{x}10^{-5}$	
	Programa en Python	-1,2697181	-78,6248179				$4{,}24264\ \text{x}10^{-7}$	$8{,}22217\ \text{x}10^{-6}$	
	Mission Planner	-1,2697178	-78,6248176				$8{,}48528\ \text{x}10^{-7}$	$1{,}64443\ \text{x}10^{-5}$	
3	APP Móvil	-1,269747	-78,624781	-1,269754	-78,624787	5,14	$9{,}21954\ \text{x}10^{-6}$	$1{,}79369\ \text{x}10^{-4}$	
	Programa en Python	-1,269751	-78,624785				$3{,}60555\ \text{x}10^{-6}$	$7{,}01469\ \text{x}10^{-5}$	
	Mission Planner	-1,269748	-78,624782				$7{,}81025\ \text{x}10^{-6}$	$1{,}51950\ \text{x}10^{-4}$	
4	APP Móvil	-1,2697955	-78,6247725	-1,2697961	-78,6247731	5,15	$8{,}48528\ \text{x}10^{-7}$	$1{,}64763\ \text{x}10^{-5}$	
	Programa en Python	-1,2697959	-78,6247729				$2{,}82843\ \text{x}10^{-7}$	$5{,}49209\ \text{x}10^{-6}$	
	Mission Planner	-1,2697956	-78,6247726				$7{,}07107\ \text{x}10^{-7}$	$1{,}37302\ \text{x}10^{-5}$	
5	APP Móvil	-1,2698405	-78,624781	-1,2698411	-78,624787	5,16	$6{,}02993\ \text{x}10^{-6}$	$1{,}16859\ \text{x}10^{-4}$	
	Programa en Python	-1,2698409	-78,624785				$2{,}00998\ \text{x}10^{-6}$	$3{,}89530\ \text{x}10^{-5}$	
	Mission Planner	-1,2698406	-78,624782				$5{,}02494\ \text{x}10^{-6}$	$9{,}73825\ \text{x}10^{-5}$	
6	APP Móvil	-1,2698734	-78,6248175	-1,269874	-78,6248181	5,18	$8{,}48528\ \text{x}10^{-7}$	$1{,}63809\ \text{x}10^{-5}$	
	Programa en Python	-1,2698738	-78,6248179				$2{,}82843\ \text{x}10^{-7}$	$5{,}46028\ \text{x}10^{-6}$	
	Mission Planner	-1,2698735	-78,6248176				$7{,}07107\ \text{x}10^{-7}$	$1{,}36507\ \text{x}10^{-5}$	
7	APP Móvil	-1,2698855	-78,6248625	-1,2698861	-78,6248631	5,18	$8{,}48528\ \text{x}10^{-7}$	$1{,}63809\ \text{x}10^{-5}$	
	Programa en Python	-1,2698859	-78,6248629				$2{,}82843\ \text{x}10^{-7}$	$5{,}46028\ \text{x}10^{-6}$	
	Mission Planner	-1,2698856	-78,6248626				$7{,}07107\ \text{x}10^{-7}$	$1{,}36507\ \text{x}10^{-5}$	
8	APP Móvil	-1,2698734	-78,6249075	-1,269874	-78,6249081	5,16	$8{,}48528\ \text{x}10^{-7}$	$1{,}64443\ \text{x}10^{-5}$	
	Programa en Python	-1,2698738	-78,6249079				$2{,}82843\ \text{x}10^{-7}$	$5{,}48145\ \text{x}10^{-6}$	
	Mission Planner	-1,2698735	-78,6249076				$7{,}07107\ \text{x}10^{-7}$	$1{,}37036\ \text{x}10^{-5}$	
9	APP Móvil	-1,2698405	-78,6249404	-1,2698411	-78,624941	5,15	$8{,}48528\ \text{x}10^{-7}$	$1{,}64763\ \text{x}10^{-5}$	
	Programa en Python	-1,2698409	-78,6249408				$2{,}82843\ \text{x}10^{-7}$	$5{,}49209\ \text{x}10^{-5}$	
	Mission Planner	-1,2698406	-78,6249405				$7{,}07107\ \text{x}10^{-7}$	$1{,}37302\ \text{x}10^{-5}$	
10	APP Móvil	-1,2697955	-78,6249524	-1,2697961	-78,6249531	5,14	$9{,}21954\ \text{x}10^{-7}$	$1{,}79369\ \text{x}10^{-5}$	
	Programa en Python	-1,2697959	-78,6249528				$3{,}60555\ \text{x}10^{-7}$	$7{,}01469\ \text{x}10^{-6}$	
	Mission Planner	-1,2697956	-78,6249525				$7{,}81025\ \text{x}10^{-7}$	$1{,}51950\ \text{x}10^{-5}$	
11	APP Móvil	-1,269747	-78,6249404	-1,269753	-78,624941	5,16	$6{,}02993\ \text{x}10^{-6}$	$1{,}16859\ \text{x}10^{-4}$	
	Programa en Python	-1,269751	-78,6249408				$2{,}00998\ \text{x}10^{-6}$	$3{,}89530\ \text{x}10^{-5}$	
	Mission Planner	-1,269748	-78,6249405				$5{,}02494\ \text{x}10^{-6}$	$9{,}73825\ \text{x}10^{-5}$	
12	APP Móvil	-1,2697177	-78,6249075	-1,2697183	-78,6249081	5,18	$8{,}48528\ \text{x}10^{-7}$	$1{,}63809\ \text{x}10^{-5}$	
	Programa en Python	-1,2697181	-78,6249079				$2{,}82843\ \text{x}10^{-7}$	$5{,}46028\ \text{x}10^{-6}$	

N° PUNTOS	SISTEMAS DE GEORREFERENCIA	Latitud medida (°)	Longitud medida (°)	Latitud real (°)	Longitud medida (°)	Distancia de punto a punto (m)	Error Absoluto (m)	Error Relativo (%)	Error Cuadrático medio RMSE (m)
	Mission Planner	-1,2697178	-78,6249076				7,07107 x10^{-7}	1,36507 x10^{-5}	
13	APP Móvil	-1,269702	-78,6248625	-1,269709	-78,6248631	5,16	7,02567 x10^{-6}	1,36156 x10^{-4}	
	Programa en Python	-1,269706	-78,6248629				3,00666 x10^{-6}	5,82686 x10^{-5}	
	Mission Planner	-1,269703	-78,6248626				6,02080 x10^{-6}	1,16682 x10^{-4}	

Respecto al análisis de los datos que se visualizan en la Tabla 40, se calculó un valor muy pequeño esto por tener un escalado de los datos con muchos decimales de los puntos geográficos esto para verificar una precisión exacta para cada punto, entonces hay un mayor error absoluto de punto a punto de la app móvil existiendo mayor desviación al valor real, respecto al sistema de georreferencia en Python presenta menor error absoluto siendo más preciso, y teniendo un valor RMSE como análisis global de todos los puntos se presentó una precisión mejor por el valor de error menor que es el sistema de georreferencia en Python.

Además, el análisis realizado en la prueba 1 con el sistema de georreferencia en Python generó el archivo HTML mostrando la ruta en el mapa de OpenStreet Map y generando un archivo KML de Google earth de forma que se comprobó que la ruta sea correcta comprobando los valores de cada punto georreferenciado una vez se realizó la planificación de la misión esto se visualiza en la Figura 194.

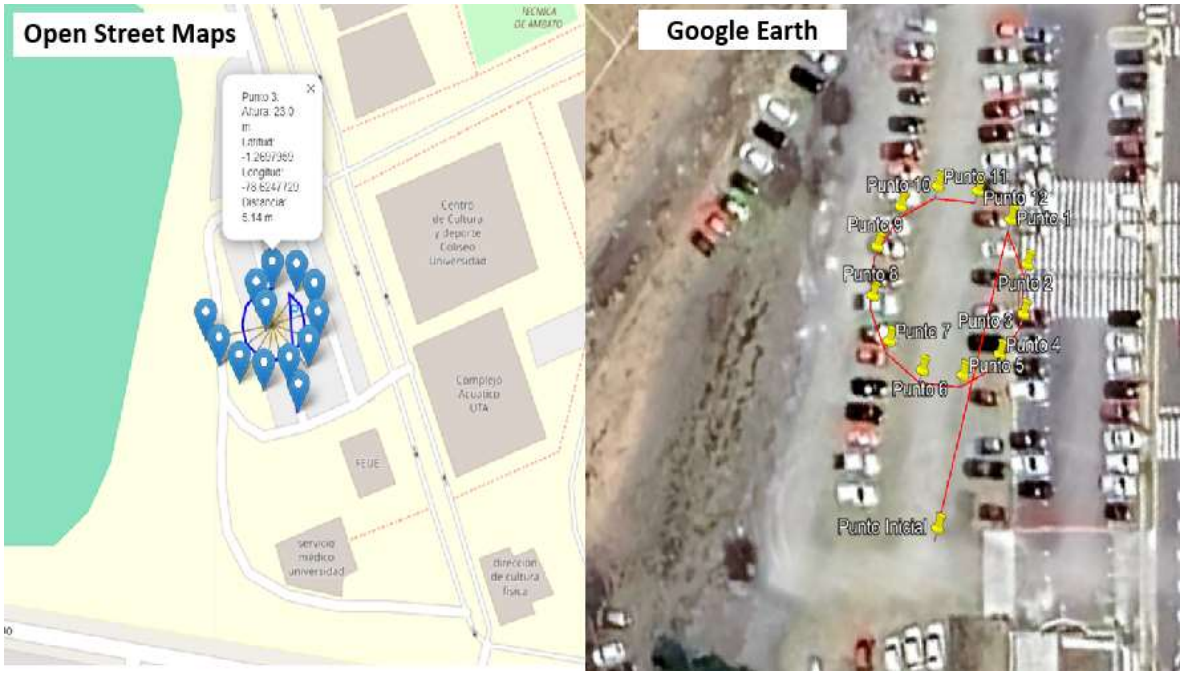


Figura 194. Ruta de la prueba 1en el mapa OpenStreet Maps y Google Earth.

b. Prueba con cantidad media de puntos georreferenciados.

Para la prueba con cantidad media de puntos georreferenciados se consideró la prueba 6 se tomó en cuenta los parámetros dados de la prueba que se indican en la Tabla 37, se comparó cada punto de la prueba y se los compara obteniendo los resultados de la Tabla 41.

Tabla 41. Comparación de puntos georreferenciados con cálculo de errores de la prueba 6

N° PUNTOS	SISTEMA	Latitud medida (°)	Longitud medida (°)	Latitud real (°)	Longitud real (°)	Distancia de punto a punto (metros)	Error Absoluto (m)	Error Relativo(%)	Error Cuadrático medio RMSE (m)
1	APP Móvil	-1,2700154	-78,624794				$8{,}48528 \times 10^{-7}$	0,000212132	$3{,}56651 \times 10^{-6}$
	Programa en Python	-1,2700158	-78,6247944			0,40	$2{,}82843 \times 10^{-7}$	$7{,}07107 \times 10^{-5}$	$2{,}82843 \times 10^{-7}$
	Mission Planner	-1,2700155	-78,6247941	-1,270016	-78,6247946		$7{,}07107 \times 10^{-7}$	0,000176777	$5{,}14782 \times 10^{-7}$
2	APP Móvil	-1,2669685	-78,6240142				$6{,}32456 \times 10^{-7}$	$3{,}32871 \times 10^{-6}$	
	Programa en Python	-1,2669689	-78,6240146			19	$2{,}82843 \times 10^{-7}$	$1{,}48865 \times 10^{-6}$	
	Mission Planner	-1,2669686	-78,6240143	-1,2669691	-78,6240144		$5{,}09902 \times 10^{-7}$	$2{,}68369 \times 10^{-6}$	
3	APP Móvil	-1,266988	-78,6240927				$6{,}32456 \times 10^{-7}$	$7{,}02728 \times 10^{-6}$	
	Programa en Python	-1,2669884	-78,6240931			9	$2{,}82843 \times 10^{-7}$	$3{,}1427 \times 10^{-6}$	
	Mission Planner	-1,2669884	-78,6240928	-1,2669886	-78,6240929		$2{,}23607 \times 10^{-7}$	$2{,}48452 \times 10^{-6}$	
4	APP Móvil	-1,2670081	-78,6241477				$6{,}32456 \times 10^{-7}$	$9{,}73009 \times 10^{-6}$	
	Programa en Python	-1,2670085	-78,6241481			6,50	$2{,}82843E \times 10^{-7}$	$4{,}35143 \times 10^{-6}$	
	Mission Planner	-1,2670082	-78,6241478	-1,2670087	-78,6241479		$5{,}09902 \times 10^{-7}$	$7{,}84465 \times 10^{-6}$	
5	APP Móvil	-1,266980	-78,6242214				$8{,}60233 \times 10^{-6}$	$9{,}66553 \times 10^{-5}$	
	Programa en Python	-1,2669884	-78,6242218			8,90	$2{,}82843 \times 10^{-7}$	$3{,}17801 \times 10^{-6}$	
	Mission Planner	-1,2669881	-78,6242215	-1,2669886	-78,6242216		$5{,}09902 \times 10^{-7}$	$5{,}72924 \times 10^{-6}$	
6	APP Móvil	-1,2670309	-78,6242094				$6{,}32456 \times 10^{-7}$	$1{,}26491 \times 10^{-5}$	
	Programa en Python	-1,2670313	-78,6242098			5,00	$2{,}82843 \times 10^{-7}$	$5{,}65685 \times 10^{-5}$	
	Mission Planner	-1,2670310	-78,6242095	-1,2670315	-78,6242096		$5{,}09902 \times 10^{-7}$	$1{,}0198 \times 10^{-5}$	

Respecto al análisis de los datos que se visualizan en la Tabla 41, a un valor medio de puntos en la misión hay un mayor error absoluto de punto a punto de la app móvil y aumento de error también del software Mission Planner existiendo mayor desviación al valor real, respecto al sistema de georreferencia en Python presenta menor error

absoluto siendo más preciso y manteniendo el error respecto a la prueba con mayor cantidad de puntos georreferenciados, y teniendo un valor RMSE como análisis global de todos los puntos se presentó una precisión mejor por un error menor que es el sistema de georreferencia en Python.

Además, el análisis realizado en la prueba 6 con el sistema de georreferencia en Python generó el archivo HTML mostrando la ruta en el mapa de OpenStreet Map y generando un archivo KML de Google earth de forma que se comprobó que la ruta sea correcta comprobando los valores de cada punto georreferenciado, pero con cantidad media de puntos una vez se realizó la planificación de la misión esto se visualiza en la Figura 195.

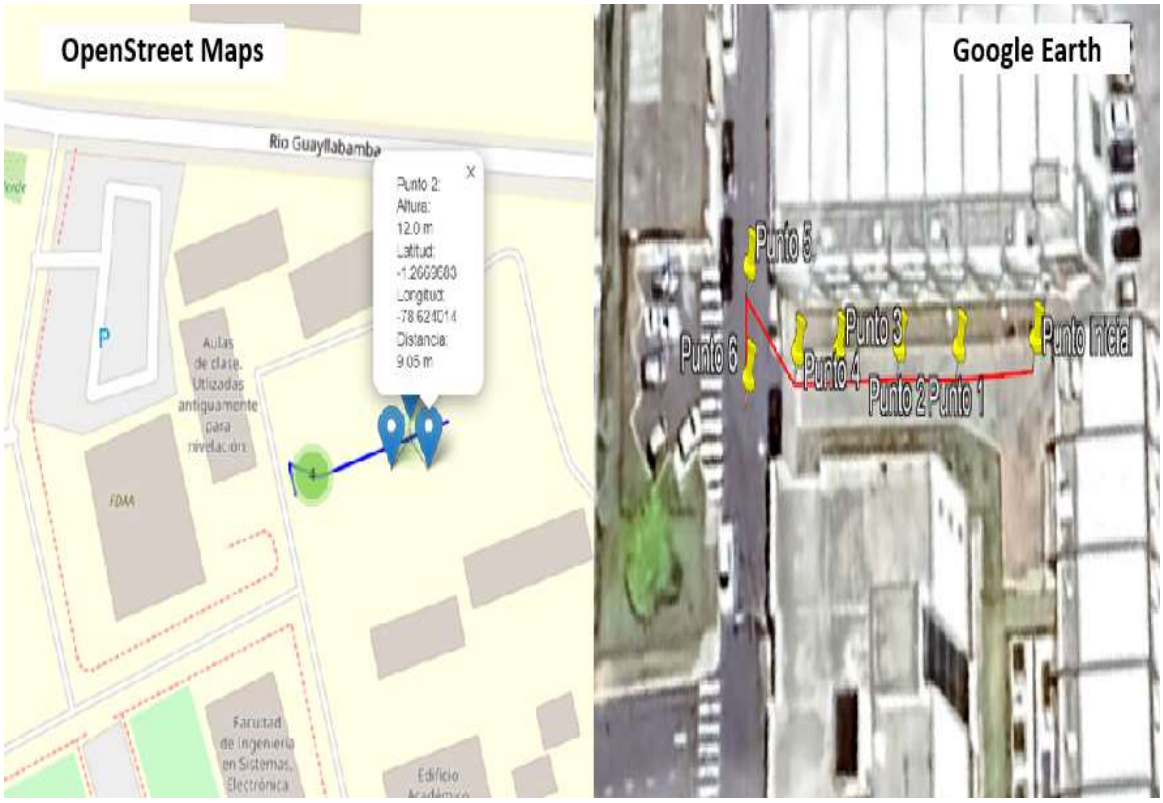


Figura 195. Ruta de la prueba 6 en el mapa OpenStreet Maps y Google Earth.

c. Prueba con cantidad baja de puntos georreferenciados.

Para la prueba con cantidad baja de puntos georreferenciados se consideró la prueba 9 se tomó en cuenta los parámetros dados de la prueba que se indican en la Tabla 37, se comparó cada punto de la prueba y se los compara obtenido los resultados de la Tabla 42.

Tabla 42. Comparación de puntos georreferenciados con cálculo de errores de la prueba 9

N° PUNTOS	SISTEMA	Latitud medida (°)	Longitud medida (°)	Latitud real (°)	Longitud real (°)	Distancia de punto a punto (metros)	Error Absoluto (m)	Error Relativo(%)	Error Cuadrático medio RMSE (m)
1	APP Móvil	-1,2669392	-78,6238893	-1,2669398	-78,6238899	0,40	$8{,}48528 \times 10^{-7}$	0,000212132	$4{,}56216 \times 10^{-6}$
	Programa en Python	-1,2669396	-78,6238897				$2{,}82843 \times 10^{-7}$	$7{,}07107 \times 10^{-5}$	$1{,}39044 \times 10^{-6}$
	Mission Planner	-1,2669393	-78,6238894				$7{,}07107 \times 10^{-7}$	0,000176777	$3{,}74166 \times 10^{-6}$
2	APP Móvil	-1,267042	-78,623794	-1,267048	-78,623799	11	$7{,}81025 \times 10^{-7}$	$7{,}10023 \times 10^{-5}$	
	Programa en Python	-1,267046	-78,623798				$2{,}23607 \times 10^{-7}$	$2{,}03279 \times 10^{-5}$	
	Mission Planner	-1,267043	-78,623795				$6{,}40312 \times 10^{-6}$	$5{,}82102 \times 10^{-5}$	
3	APP Móvil	-1,2673312	-78,6237728	-1,2673318	-78,6237734	19	$8{,}48528 \times 10^{-7}$	$4{,}46594 \times 10^{-6}$	
	Programa en Python	-1,2673316	-78,6237732				$2{,}82843 \times 10^{-7}$	$1{,}48865 \times 10^{-6}$	
	Mission Planner	-1,2673313	-78,6237729				$7{,}07107 \times 10^{-7}$	$3{,}72161 \times 10^{-6}$	

Respecto al análisis de los datos que se visualizan en la Tabla 42, a un valor medio de puntos en la misión hay un mayor error absoluto de punto a punto de la app móvil y aumento de error también del software Mission Planner existiendo mayor desviación al valor real, respecto al sistema de georreferencia en Python presenta menor error absoluto siendo más preciso y manteniendo el error respecto a la prueba con cantidad de alta y media de puntos georreferenciados, y teniendo un valor RMSE como análisis global de todos los puntos se presentó una precisión mejor por un error menor que es el sistema de georreferencia en Python respecto a los otros.

Además, el análisis realizado en la prueba 9 con el sistema de georreferencia en Python generó el archivo HTML mostrando la ruta en el mapa de OpenStreet Map y generando un archivo KML de Google earth de forma que se comprobó que la ruta sea correcta comprobando los valores de cada punto georreferenciado, pero con cantidad baja de puntos una vez se realizó la planificación de la misión esto se visualiza en la Figura 196.

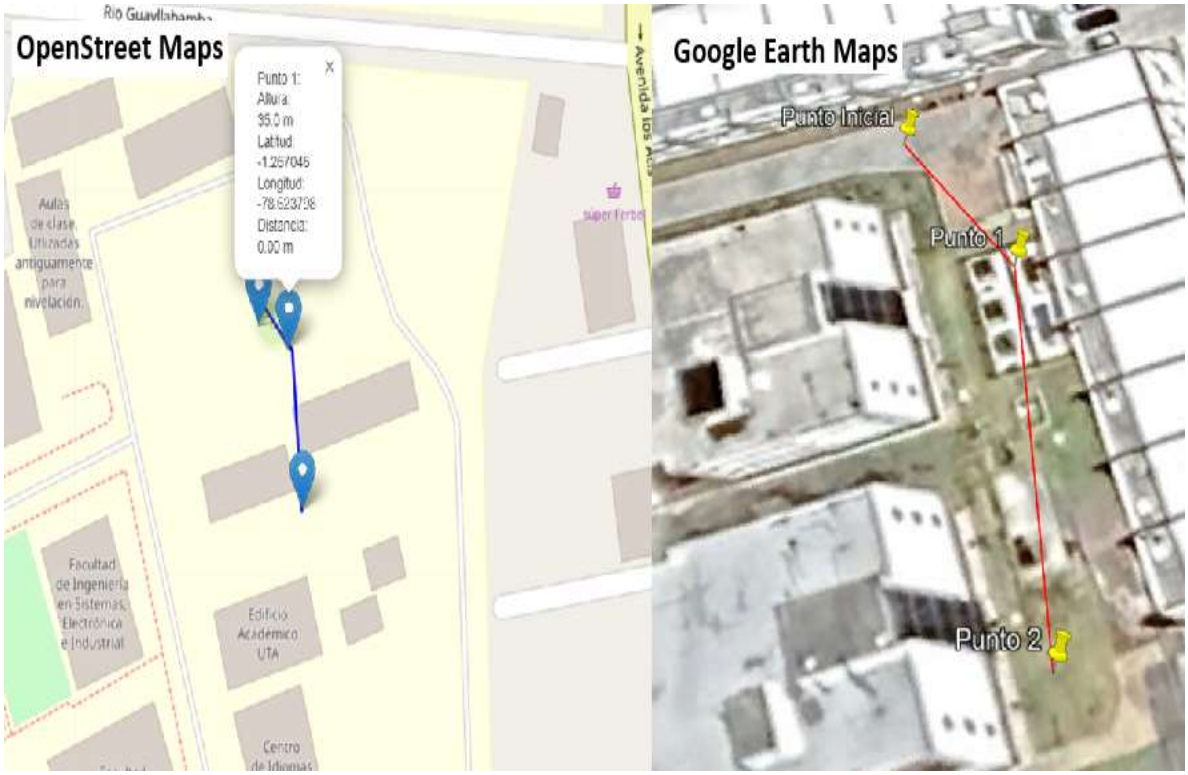


Figura 196. Ruta de la prueba 9 en el mapa OpenStreet Maps y Google Earth.

3.2.3 Eficiencia del sistema de Georreferencia en Python para el sistema automatizado de vuelo del UAV

En el análisis anterior se obtuvo los errores de acuerdo a valores medidos de los sistemas y valores reales recogidos por el geoportal del Instituto geográfico militar del Ecuador para aquello se tomó en cuenta tres pruebas esto respecto a la cantidad de puntos georreferenciados alta, media y baja, los resultados se comparan con las salidas obtenidas de los sistemas como Mission Planner, que utiliza mapas satelitales de Google Earth, una app móvil que utiliza el mapa USGS topo y el programa en Python que usa el mapa OpenStreet Map. A través de esta comparación, se buscó determinar la efectividad del sistema de georreferencia en términos de exactitud, para este cálculo se especifica un error máximo permitido (EMP) que definió un valor límite para la evaluación de un valor de precisión aceptable en cada sistema, este valor es basado en cota predefinida de acuerdo a estándares, específicamente en Ecuador se rige una normativa del reglamento a la Ley de Cartografía Nacional regulada por el Instituto Geográfico Militar (IGM), el cual incluye el valor de precisión y exactitud alta de la información cartográfica, el valor preestablecido para Sistema Global de Navegación

por Satélite (GNSS) de 0,4 se escaló el valor de error cuadrático medio (RMSE) ya que se toma algunos decimales para verificar los datos de cada punto geográfico que arroja cada sistema esto con la ecuación 15, y después se calculó la eficiencia sin escalamiento con la ecuación 16, y por último la eficiencia escalada se calculó con la ecuación 17, se muestran los valores obtenidos en la Tabla 43 por cada prueba considerada [65].

$$\text{RMSE Escalado} = -log_{10}(\text{RMSE}) \quad (15)$$

$$Eficiencia\ (\%) = \left(1 - \frac{RMSE}{EMP}\right) x100 \quad (16)$$

$$Eficiencia\ escalada\ (\%) = \frac{RMSE\ Escalado}{RMSE\ Escalado\ Máximo} \text{x} \left(\left(1 - \frac{RMSE}{EMP}\right) x100\right) \quad (17)$$

Tabla 43. Parámetro de la Eficiencia de cada sistema

Pruebas	**Sistemas**	**Error Cuadrático Medio RMSE (m)**	**RMSE Escalado**	**EMP (Error Máximo Permitido)**	**Eficiencia con EMP(%)**	**Eficiencia escalada (%)**
Cantidad de puntos alta / Prueba 1	App movil	$4{,}20549\ x10^{-6}$	5,376183301	0,4	99,99894	82,09
	Programa en Python	$1{,}54397\ x10^{-6}$	5,811360888	0,4	99,99961	88,74
	Mission Planner	$3{,}43029\ x10^{-6}$	5,464668543	0,4	99,99914	83,44
Cantidad de puntos media / Prueba 6	App movil	$3{,}56651\ x10^{-6}$	5,447756444	0,4	99,99910	83,19
	Programa en Python	$2{,}82843\ x10^{-7}$	6,548455004	0,4	99,99992	99,99
	Mission Planner	$5{,}14782\ x10^{-7}$	6,288377064	0,4	99,99987	96,02
Cantidad de puntos baja / Prueba 9	App movil	$4{,}56216\ x10^{-6}$	5,34082918	0,4	99,99885	81,55
	Programa en Python	$1{,}39044\ x10^{-6}$	5,856846629	0,4	99,99965	89,43
	Mission Planner	$3{,}74166\ x10^{-7}$	5,426935982	0,4	99,99906	82,87

El análisis de la eficiencia por cada prueba mostro que el programa de Python tiene una mejor precisión con una cantidad de puntos media y bajo, con un valor de 99,99 % y 89,43 % siendo eficiente en cuanto a la exactitud respecto a los otros sistemas, respecto a una cantidad de puntos alta se obtuvo un valor de 88,74 % considerando un ligero cambio respecto a esa prueba, pero con un valor alto a los otros sistemas. Se

calculo un valor promedio de entre las pruebas para evidenciar una eficiencia promedio de 92,73 %, teniendo un exactitud y precisión confiable respecto a los otros sistemas de georreferencia estos valores se visualizan en la Tabla 44, además de un gráfico de comparativa de porcentajes de eficiencia promedio que se visualiza en la Figura 197.

Tabla 44. Comparativa eficiencia promedio de cada uno de los sistemas

SISTEMA	EFICIENCIA PROMEDIO (%)
App movil	82,28
Pograma en Python	92,73
Mission Planner	87,45

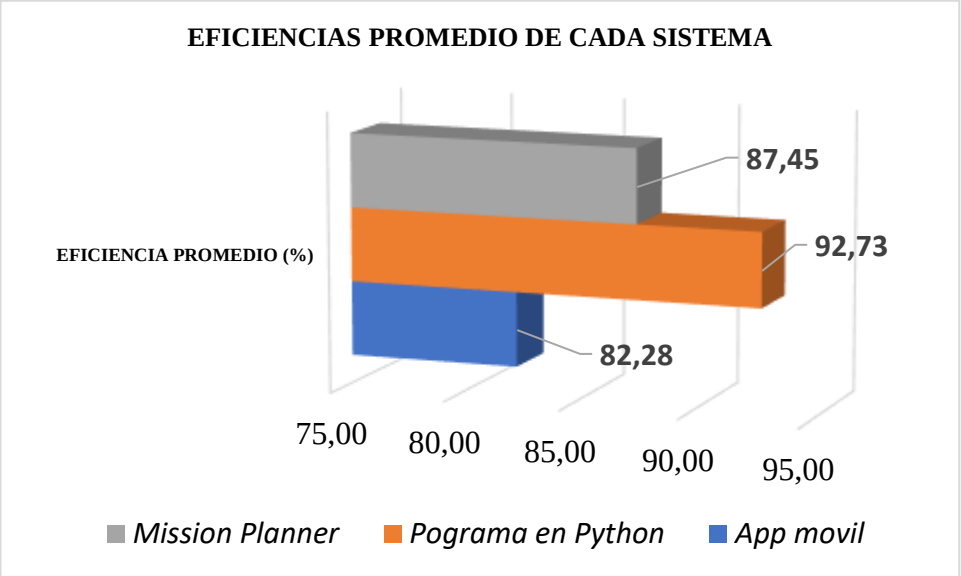


Figura 197. Gráfico de comparación de eficiencias promedio de cada sistema.

3.3 Presupuesto

El presupuesto destinado para la ejecución de este proyecto de investigación detalla los principales rubros que fueron necesarios para su desarrollo. Entre los costos se incluyen componentes de hardware, sensores, sistemas de comunicación, materiales para construcción y diseño, equipos de control y telemetría, además de otros elementos y servicios fundamentales para asegurar la funcionalidad y operatividad del sistema.

Tabla 45. Costos.

CATEGORIA	DESCRIPCION	CANTIDAD	PRECIO UNITARIO (USD)	COSTO TOTAL (USD)
Plataforma Base y Componentes Principales	Base de Dron Tarot 680 Pro	1	$ 150.00	$ 150.00
	Controladora Pixhawk 2.4.8	1	$ 120.00	$ 120.00
	Motores Brushless 5010 750Kv	6	$ 30.00	$ 180.00
	Controladores de velocidad ESC	6	$ 25.00	$ 150.00
	Batería LiPo 6s (22.2V, 6000 mA)	1	$ 80.00	$ 80.00
Sensores y equipos de Detección.	LIDAR Tf-Luna	1	$ 25.00	$ 25.00
	Modulo GPS MN-8	1	$ 30.00	$ 30.00
	Sistema FPV	1	$ 200.00	$ 200.00
	Cámara GoPro Hero 5	1	$ 350.00	$ 350.00
Comunicación y Telemetría.	Modulo telemetría (transmisor y receptor)	1	$ 60.00	$ 60.00
	Radiorreceptor: FrSky R-XSR	1	$ 30.00	$30.00
Sistemas de Estabilidad y Control	Gimbal para cámara GoPro	1	$ 140.00	$ 140.00
	Buzzer (alarma de estado)	1	$ 5.00	$ 5.00
Estación Terrena y Accesorios	Laptop Asus G512	1	$ 1200.00	$ 1200.00
	Control remoto (Radiomaster TX16S)	1	$ 150.00	$ 150.00
Construcción y Adaptaciones	Materiales de montaje (tornillos, cables, conectores, etc.)		$ 50.00	$ 50.00
	Impresiones 3D para adaptaciones		$ 25.00	$ 25.00
Gastos Operativos	Luz eléctrica (carga de baterías, laptops, dispositivos eléctricos)		$ 20.00	$ 20.00

CATEGORIA	DESCRIPCION	CANTIDAD	PRECIO UNITARIO (USD)	COSTO TOTAL (USD)
	Repuestos (hélices adicionales, baterías adicionales)		$ 90.00	$ 90.00
	Internet		$ 30.00	$ 30.00
Otros Costos	Transporte		$ 360.00	$ 360.00
	Documentación (impresiones, copias, etc.)		$ 50.00	$ 50.00
	Alimentación		$ 320.00	$ 320.00
TOTAL ESTIMADO				$ 3815.00

CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- En el desarrollo de sistemas de navegación autónoma para UAVs, se determinó que, entre diversas tecnologías como ROS, ORB-SLAM, visión artificial, DroneKit, MAVLink, y sensores como LIDAR y GPS que la integración de GPS y MAVLink es la opción más adecuada de acuerdo a los elementos del sistemas, precisión y eficiencia, además del uso en la mayoría de drones existentes en la actualidad. El GPS proporciona datos exactos de posicionamiento para la navegación y el seguimiento en tiempo real, mientras que MAVLink permite una comunicación fluida entre el UAV y la estación de control, optimizando la autonomía y supervisión del sistema.

- Las tecnologías de sistemas de navegación autónoma que se escogió para un hexacóptero de acuerdo a investigaciones previas revisadas durante el desarrollo del proyecto, la más apta fue la tecnología GPS para planificación de rutas con un sistema de georreferencias que se ajustó al sistema creado, una variación de la eficiencia de acuerdo a los estudios mencionados entre 7% y 12% esto específicamente para sistemas de navegación por satélite (GNSS) o posicionamiento preciso de puntos (PPP) el cuál garantizo la estabilidad y exactitud de las trayectorias del hexacóptero.

- Las pruebas de vuelo realizadas con el hexacóptero confirmaron la relación entre el tiempo de funcionamiento y las condiciones operativas, como la velocidad de vuelo, el tipo de ruta y la altitud. Se observó que, a medida que la velocidad de vuelo aumentaba a más de 10 metros por segundo y la altitud superaba los 30 metros, el tiempo de vuelo disminuía para completar la misión. Por otro lado, a altitudes más bajas (por debajo de 4 metros) y con una velocidad reducida de menos de 2 metros por segundo, el dron lograba un mayor tiempo de vuelo, lo que lo hace ideal para misiones de corta duración o en entornos controlados.

- El sistema de navegación autónoma desarrollado en Python para videovigilancia en la Universidad Técnica de Ambato mostró una eficiencia promedio del 92,73% en rutas con puntos georreferenciados altas, medias y bajas, comparada con la precisión del software libre Mission Planner y una aplicación móvil para la planificación de rutas con errores cuadráticos medios (RMSE) ajustados a un error menor para rutas con un mayor número de puntos georreferenciados para sistemas de navegación por satélite (GNSS).

4.2 Recomendaciones

- Para maximizar la autonomía de vuelo del hexacóptero, se recomienda utilizar baterías de mayor capacidad y optimizar el balance energético del sistema. Esto implica integrar controladores de motores de alta eficiencia y revisar el peso total del UAV, priorizando componentes ligeros como hélices de fibra de carbono y estructuras robustas. Además, se sugiere implementar métodos de gestión de energía que ajusten dinámicamente el consumo según la demanda de los sistemas electrónicos durante la operación.

- Dado el enfoque en la seguridad del campus universitario, es recomendable incorporar cámaras de mayor resolución y ángulos ajustables para optimizar la cobertura de vigilancia. Se sugiere usar tecnología de cámaras térmicas en combinación con cámaras RGB para garantizar visibilidad en condiciones de baja iluminación. Para complementar el sistema, se deben configurar zonas de patrullaje automatizadas con tiempos predefinidos, asegurando un monitoreo constante y eficiente en las áreas críticas del campus.

- Es esencial integrar herramientas avanzadas con sensores de detección de obstáculos que se puedan implementar con librerías especializadas en algoritmos de optimización basados en Python. Estas herramientas pueden mejorar la precisión en el diseño de rutas, permitiendo evitar obstáculos de manera más eficiente. Además, se recomienda realizar simulaciones exhaustivas utilizando software como MATLAB antes de ejecutar misiones en campo, con el objetivo de prevenir y solucionar posibles fallos en el comportamiento del UAV.

- Para garantizar un manejo óptimo del sistema, es necesario realizar capacitaciones regulares al personal encargado del monitoreo y operación del UAV. Estas deben enfocarse en el uso avanzado de software diseñado en Python, configuración de parámetros de vuelo, y mantenimiento preventivo del dron. Asimismo, se recomienda mantenerse actualizado en tecnologías emergentes como ROS (Robot Operating System) enfocado en los protocolos MAVLink o MAVProxy además de incluir visión artificial, para implementar mejoras continuas en el sistema de navegación autónoma.

REFERENCIAS BIBLIOGRÁFICAS

[1] E. y. Observatorio, «Inseguridad y delincuencia en Ambato octubre 2020 – marzo 2022,” Edu.ec,» [En línea]. Available: https://obest.uta.edu.ec/wp-content/uploads/2022/07/inseguridad_y_delincuencia_en_ambato_2022.pdf.. [Último acceso: 01 Octubre 2024].

[2] O. Tungurahua, «Introducción En este contexto, es importante capacitar y fortalecer a los miembros policiales, militares,» [En línea]. Available: Introducción En este contexto, es importante capacitar y fortalecer a los miembros policiales, militares. [Último acceso: 01 Octubre 2024].

[3] D. R. y. M. Bravo, «Repositorio de Universidad Santo Tomás,» 30 Marzo 2019. [En línea]. Available: https://repository.usta.edu.co/handle/11634/19029. [Último acceso: 01 Octubre 2024].

[4] A. Meythaler, *Desarrollo de un sistema de Navegación Autónoma para UAV basado en FPGA,* Latacunga - Ecuador: Departamento de Eléctrica y Electrónica Universidad de las Fuerzas Armadas ESPE-EL, 2020.

[5] D. S. Iglesias, *Aplicación de técnicas de Visión Artificial en la retransmisión y grabaciones realizadas por Drones,* España, Coruña: Facultad de Informática de la Universidad de da Coruña, 2022.

[6] P. G. García, *Sistema de seguimiento de personas desde un dron,* España, Alicante: Universidad de Alicante, 2022.

[7] F. M. C. Sarango, *Sistema de localización de personas en catástrofes utilizando vehículos aéreos no tripulados,* Ambato, Ecuador: Universidad Técnica de Ambato, 2024.

[8] «Dictionary of military and associated terms,» United States Department of Defense, Agosto 2005. [En línea]. Available: http://www.bits.de/NRANEU/others/jp-doctrine/jp1 02(05).pdf. [Último acceso: 15 10 2024].

[9] J. M. P. Ramos, *Sistema de Vigilancia Adaptativo basado en la Coordinación de UAVs en Entornos Afectados por Catástrofes*, Castilla - La Mancha: ESCUELA SUPERIOR DE INFORMÁTICA, 2026.

[10] C. d. IDC, «IDC,» Profesionales de Drones, 5 Febrero 2024. [En línea]. Available: https://idc.apddrones.com/drones/drones-para-seguridad-y-vigilancia-idc/. [Último acceso: 15 10 2024].

[11] C. d. D. RC, «DRON RC,» Shenzhen Baichen Trading Co., Ltd., 10 Enero 2024. [En línea]. Available: https://rcdrone.top/es/products/tarot-m860-dron?srsltid=AfmBOorr_o_AQNHrREYvfhLZWXgbL9j2-Zq6DWNOZwewwnlR2BQkhysM. [Último acceso: 10 15 2024].

[12] C. d. Ubuuy, «Ubuuy - Tarot Hexacopter DIY Dro,» QWinOut, 06 04 2021. [En línea]. Available: https://www.ubuy.ec/en/product/47C77O0Q8-qwinout-arfpnp-full-set-hexacopter-diy-drone-kit-tarot-690mm-frame-with-750kv-motor-gps-pix-2-4-8-32-bit-flight-controller-arf?srsltid=AfmBOoo1G5vw34DpCUZYZm-20ZBnDXamAlRsnTxxefUpCn7vSjkmbUlD&ref=hm-google-redirect. [Último acceso: 22 11 2024].

[13] A. Martí, «XATAKA,» Blog Xataka, 30 Octubre 2019. [En línea]. Available: https://www.xataka.com/drones/dji-mavic-mini-caracteristicas-precio-ficha-tecnicaAT. [Último acceso: 14 Noviembre 2024].

[14] E. F. S. P. Kevin Jair Sandoval Jaramillo, *SISTEMA TELEMETRIZADO DE CARGA INALÁMBRICA UTILIZANDO TECNOLOGÍA INTELIGENTE PARA VEHÍCULOS AÉREOS NO TRIPULADOS (UAV) EN LA UNIVERSIDAD TÉCNICA DE AMBATO,* Ambato: Universidad Tècnica de Ambato, 2024.

[15] R. V. B. P. ESPINOSA, «EMPLEO DE LOS UAV, EN OPERACIONES DE SEGURIDAD Y VIGILANCIA EN LAS ÁREAS ESTRATÉGICAS EN EL ECUADOR,» *Revista De Ciencias De Seguridad Y Defensa*, pp. 86-100, 2021.

[16] Santos, «GRUPO UAS,» Colaboradores de Blog grupo UAS, 19 12 2024. [En línea]. Available: https://grupo-uas.com/los-drones-en-la-seguridad-aplicaciones-en-vigilancia-proteccion-y-emergencias/?utm_source=chatgpt.com. [Último acceso: 29 12 2024].

[17] «dji Enterprise blog,» DJI Enterprise, 24 Julio 2024. [En línea]. Available: https://enterprise-insights.dji.com/es/blog/drones-policia-para-seguridad-ciudadana?utm_source=chatgpt.com. [Último acceso: 29 12 2024].

[18] R. A. Española, «Diccionario panhispánico del español jurídico,» 10 Marzo 2023. [En línea]. Available: https://dpej.rae.es/lema/videovigilancia. [Último acceso: 01 Octubre 2024].

[19] A. E. d. P. d. Datos, *Guia sobre el uso de videocamaras para seguridad,* Madrid, 2015.

[20] M. Lechner, «Tecnologías aplicadas a la seguridad ciudadana: desafíos para la videovigilancia,» RIDAA, 2016, pp. 1-17.

[21] VISUALNETCAM, «Alarmas y Videovigilancia,» 15 Junio 2024. [En línea]. Available: https://alarmasyvideovigilancia.com/diferentes-tecnologias-videovigilancia/. [Último acceso: 01 Octubre 2024].

[22] A. Devasia, «The Network Installers,» 29 Septiembre 2024. [En línea]. Available: https://thenetworkinstallers.com/es/blog/aplicaciones-de-video-vigilancia/. [Último acceso: 01 Octubre 2024].

[23] S. STAFF, «stym,» 18 Marzo 2021. [En línea]. Available: https://stym.com.ar/soluciones-para-edificios-inteligentes/solucion-de-edificios-inteligentes-hikvision/. [Último acceso: 01 Octubre 2024].

[24] C. d. U. GROUP, «UMILES,» UMILES GROUP, 07 Noviembre 2023. [En línea]. Available: https://umilesgroup.com/drone-vigilancia/. [Último acceso: 15 10 2024].

[25] RUSHFPV, «Amazon - Rush Tank II Ultimate 5.8GHz VTX con audio inteligente,» Colaboradores de Amazon, 24 09 2021. [En línea]. Available: https://www.amazon.com/-/es/RUSHFPV-Ultimate-5-8GHz-audio-inteligente/dp/B08FYTVY4Y. [Último acceso: 14 11 2024].

[26] Yoidesu, «Amazon - Yoidesu 1.2G VTX,,» Colaboradores de Amazón, 02 Julio 2024. [En línea]. Available: https://www.amazon.com/-/es/transmisor-frecuencia-latencia-compacto-carreras/dp/B0D8L7GGGK. [Último acceso: 14 Noviembre 2024].

[27] Makerfire, «AMAZON - Matek Video Transmitter 5.8G VTX,» Colaboradores de Amazon, 28 Septiembre 2018. [En línea]. Available: https://www.amazon.com.mx/Transmitter-VTX-HV-Control-Female-Quadcopter/dp/B0789G4VYK. [Último acceso: 14 Noviembre 2024].

[28] U-Angel, «AliExpress - Controlador de Motor sin escobillas para Dron teledirigido, controlador de cardán de 3 ejes, 1 /2 / 5 piezas, Storm32, BGC, 32 bits, V1.32, DRV8313,» Tienda Online AliExpress, 14 10 2024. [En línea]. Available: https://es.aliexpress.com/i/32390138874.html. [Último acceso: 22 10 2024].

[29] PAMENET, «Amazon - PAMENET Controlador de cardán sin escobillas BGC 3.1 de 2 ejes/controlador PTZ controlador de motor con sensor 6050 para FPV Multirotor Piezas de repuesto Accesorios,» Tienda Online Amazo, 02 12 2023. [En línea]. Available: https://www.amazon.es/PAMENET-Controlador-escobillas-controlador-Multirotor/dp/B0D1KK3M1F. [Último acceso: 22 10 2024].

[30] Steadywin, «AliExpress - BaseCam-SimpleBGC, controlador de cardán de 3 ejes, codificador pequeño, PCB, 2-IMU I2C, Set de revisión C, Alexmos, 32 bits,» Tienda Online - AliExpress, 23 09 2022. [En línea]. Available: https://es.aliexpress.com/i/1005001770140515.html. [Último acceso: 22 10 2024].

[31] D. Plaza Rey, «Navegación autónoma de drones y automatización de rutas aplicadas a la limpieza de edificios,» 2017.

[32] C. Benito, *Implementación en ROS de un control para seguimiento de trayectoria para un cuatrirotor*, Puebla: BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA, 2023.

[33] J. Martinez-Carranza, L. Valentin, F. Marquez-Aquino, J. C. González-Islas y N. Loewen, «Detección de obstáculos durante vuelo autónomo de drones utilizando SLAM monocular.,» *Res. Comput. Sci.*, vol. 114, p. 111–124, 2016.

[34] M. A. Acuña Chumpitazi y R. A. Tito Vargas, «Diseño de un vehículo aéreo no tripulado con visión artificial para detección de plagas y malformaciones de cultivos en la localidad de Caqui, Huaral.,» 2023.

[35] hamishwillee, «GIT HUB,» Drone kit python, [En línea]. Available: https://github.com/dronekit/dronekit-python.

[36] Readytosky, «AMAZON - Readytosky Pixhawk PX4 Controlador de vuelo Pixhawk 2.4.8 32 Bit tablero de control de vuelo+interruptor de seguridad+zumbador+divisor I2C módulo de expansión+tarjeta SD de 4 GB,» Tienda Online - Amazon, 20 03 2023. [En línea]. Available: https://www.amazon.com/-/es/Readytosky-Controlador-interruptor-seguridad-expansión/dp/B07CHQ7SZ4#:~:text=Readytosky%20Pixhawk%20PX4%20-%20Controlador%20de,tarjeta%20SD%20de%2016%20GB&text=1%20Frecuencia%3A%20168%20MHz%2C%20256%20K%20de%20RAM.&text=1.,de%2. [Último acceso: 21 10 2024].

[37] C. d. vistronica, «VISTRÓNICA - Controlador De Vuelo Ardupilot APM 2.8 - pines 90°,» Tienda Online - VISTRÓNICA SAS, 12 02 2024. [En línea]. Available: https://www.vistronica.com/aeromodelismo/controlador-de-vuelo-ardupilot-apm-28-pines-90-detail.html. [Último acceso: 21 10 2024].

[38] Completestore, «Amazon-Actualización NAZE32 SP Racing F3 Control de vuelo Acro 6 DOF/Deluxe 10 DOF para Multicopter (Deluxe),» Tienda online Amazon, 15 02 2023. [En línea]. Available: https://www.amazon.com/-/es/Completestore-Actualización-NAZE32-Control-Multicopter/dp/B01DVUQN9M. [Último acceso: 21 10 2024].

[39] A. Sánchez, «GPS en Drones: Para qué sirve y cuándo se utiliza,» UMILES, 23 Agosto 2023. [En línea]. Available: https://umilesgroup.com/gps-drones/.. [Último acceso: 25 11 2025].

[40] CN(Origen), «Holybro Micro M8N GPS,» Aliexpress, 23 02 2021. [En línea]. Available: https://es.aliexpress.com/item/1005005603284783.html. [Último acceso: 17 11 2024].

[41] C. d. Naylamp, «Módulo GPS NEO-8M (compatible),» Naylamp Mechatronics - Perú, [En línea]. Available: https://naylampmechatronics.com/sensores-posicion-inerciales-gps/1076-modulo-gps-neo-8m-compatible.html. [Último acceso: 17 11 2024].

[42] NestNiche, «Módulo GPS M8N - Amazon,» Amazon, 06 09 2023. [En línea]. Available: https://www.amazon.com/-/es/receptora-navegador-integrada-compatible-controlador/dp/B0CHFK661W. [Último acceso: 17 11 2024].

[43] «REPOSITORIO INSTITUCIONAL,» UNIVERSIDAD DE GUAYAQUIL, 01 Mayo 2024. [En línea]. Available: https://repositorio.ug.edu.ec/items/33fcfdb0-cdd8-4a08-a5fd- 741d16660c50.. [Último acceso: 01 0ctubre 2024].

[44] U. N. P. R. GALLO, «REPOSITORIO INSTITUCIONAL,» 10 Diciembre 2019. [En línea]. Available: https://repositorio.unprg.edu.pe/handle/20.500.12893/7942.. [Último acceso: 10 Octubre 2024].

[45] «DiaRace,» Industrial & Hobby Aircraft SL, 24 Marzo 2023. [En línea]. Available: https://iha-race.com/producto/caddx-camara-de-accion-walnut/. [Último acceso: 21 Octubre 2024].

[46] C. d. Amazon, «Amazon tienda online,» Amazon, 17 Marzo 2017. [En línea]. Available: https://www.amazon.com/-/es/FPVDrone-1200TVL-cámara-definición-Quadcopter/dp/B06XPX18VY. [Último acceso: 21 Octubre 2024].

[47] Amazon, «Huawei E586 DESBLOQUEADO Inalámbrico HSPA + 21 M Pocket WIFI 3G Módem móvil 21Mbps,» 14 Junio 2023. [En línea]. Available: https://www.amazon.de/-/en/Huawei-E586-UNLOCKED-Wireless-Pocket-Mobile/dp/B0061JPI4C. [Último acceso: 21 Octubre 2024].

[48] Robosfusion, «Amazon - Telemetría LoRa,» Tienda Online Amazon, 23 10 2022. [En línea]. Available: https://www.amazon.com/-/es/Telemetría-alcance-robótica-transmisión-Ardupilot/dp/B0CP3X3KLG. [Último acceso: 12 09 2024].

[49] CN(Origen), «HolyBro SiK radio de telemetría V3 100mW 500mW 433MHZ 915MHZ para APM Pixhawk Radiolink MINI PIX RC Drone piezas de controlador de vuelo,» Tienda Online AliExpress, 12 06 2022. [En línea]. Available: https://es.aliexpress.com/i/1005005978905366.html. [Último acceso: 16 12 2024].

[50] GALVISYAJAIRA, «Mercado Libre - Youngrc 3dr 915mhz 100mw Radio Telemetria Aire Y Terreno Mod,» Tienda Online Mercado Libre, 18 06 2022. [En línea]. Available: https://articulo.mercadolibre.com.co/MCO-2301114742-youngrc-3dr-915mhz-100mw-radio-telemetria-aire-y-terreno-mod-_JM#polycard_client=search-nordic&position=15&search_layout=stack&type=item&tracking_id=e011651a-b44a-4828-8ccc-10f8bb3aab48. [Último acceso: 22 10 2024].

[51] H. A. Pérez-Sánchez, E. U. Benítez-Rendón y M. Díaz-Rodríguez, «Sistema de georeferenciado de imágenes con drones,» *Ra Ximhai,* vol. 13, p. 65–77, 2017.

[52] D. G. d. A. C. (DGAC), «Reglamento de uso de aeronaves pilotadas a distancia (RPAS),» 23 01 2022. [En línea]. Available: https://www.dgac.gob.ec.. [Último acceso: 20 01 2025].

[53] E. C. Calderón Díaz JA, «Pino Candelillo,» Instituto Nacional de Bosques, Guatemala, 2017.

[54] Catplantas, «JBR,» 10 01 2020. [En línea]. Available: https://www.cicy.mx/Documentos/CICY/JBR/2020/CatPlantas/Palmeras_catalogo.pdf. [Último acceso: 18 12 2024].

[55] E. (. E. A. R. C. N. S.A), «Guías de Diseño Redes Aéreas,» Empresa Eléctrica Ambato, Ambato, 2021.

[56] INEN, «Señalización Vial. Parte 1. Señalización Vertical,» Instituto Ecuatoriano de normalización (INEN), Quito - Ecuador, 2011.

[57] V. J. Racines Parreño, *Reciclaje de los espacios interiores y transformación para sus nuevos usos*, Ambato: Universidad técnica de Ambato, 2018.

[58] AV-E, «AV Electronics - Batería Lipo Turnigy,» Colaboradores de AV Electronics, [En línea]. Available: https://avelectronics.cc/producto/bateria-lipo-turnigy-11-1v-1500mah-3s-20c/?srsltid=AfmBOor1Ed0h634K72wUbtsijkwzBYJdmb7qLr2GFit5O1Mu-ezrk3eZ. [Último acceso: 17 11 2024].

[59] C. d. Amazon, «Amazon - HRB 6S,» YOWOO, [En línea]. Available: https://www.amazon.com/-/es/6000mAh-Batería-50C-100C-Quadcopter-Helicóptero/dp/B06XKRWD4Z. [Último acceso: 17 11 2024].

[60] «UAV NAVIGATION,» 28 11 2024. [En línea]. Available: https://www.uavnavigation.com/es/productos/ahrs-imu/que-es-una-imu#:~:text=Acelerómetros%3A%20miden%20las%20fuerzas%20gravitatorias,sumarán%20las%20fuerzas%20de%20inercia..

[61] vpolooro, «Sribd,» 31 10 2023. [En línea]. Available: https://es.scribd.com/document/681341618/05-Planificacion-y-Analisis-de-La-Mision. [Último acceso: 17 12 2024].

[62] A. D. Team, «ArduPilot - Mission Commands¶,» ArduPilot, 23 10 2020. [En línea]. Available: https://ardupilot.org/copter/docs/common-mavlink-mission-command-messages-mav_cmd.html#common-mavlink-mission-command-messages-mav-cmd. [Último acceso: 17 12 2024].

[63] V. H. A. J. V.-A. Christian P. Carvajal, «Visual Servoing Using Sliding-Mode Control with Dynamic,» *MDPI*, nº 8, p. 22, 2024.

[64] M. d. d. N. y. I. G. Militar, «GeoPortal,» Infraestructura de Datos Espaciales para instituto Geográfico Militar, 2017. [En línea]. Available: https://www.geoportaligm.gob.ec/portal/. [Último acceso: 14 01 2025].

[65] IGM, «Normalizacion técnica cartográfica,» Infraestructura de Datos espaciales para Instituo Geográfica Militar, 2017. [En línea]. Available: https://www.geoportaligm.gob.ec/portal/index.php/normalizacion-tecnica/.

[66] C. d. Automatec, «AUTOMATEC,» BLOG AUTOMATEC, 10 Marzo 2020. [En línea]. Available: https://automatec.com.mx/blog/tipos-de-grabadores-de-video-para-cctv. [Último acceso: 01 Octubre 2024].

[67] C. d. M. Libre, «Mercado Libre - Sensor Medidor De Distancia Ultrasonido Hc-sr04,» PAGINA DE VENTA ONLINE ECUADOR, 01 02 2023. [En línea]. Available: https://articulo.mercadolibre.com.co/MCO-2679357168-sensor-medidor-de-distancia-ultrasonido-hc-sr04-arduino-pic-_JM#is_advertising=true&position=1&search_layout=stack&type=pad&tracking_id=7ccb407c-f889-4252-bdae-5c5eff761b94&is_advertising=true&ad_domain=. [Último acceso: 21 10 2024].

[68] C. d. D. ROBOT, «DF Robot - Fermion: URM37 Ultrasonic Distance Sensor Breakout,» Blog tienda online DF ROBOTS, 15 01 2023. [En línea]. Available: https://www.dfrobot.com/product-53.html. [Último acceso: 21 10 2024].

[69] S. PUIG, «Mercado Libre - Modulo Sensor Ultrasonico Gh311,» Tienda online Ecuador - mercado libre, 23 05 2023. [En línea]. Available: https://articulo.mercadolibre.com.ec/MEC-554047842-modulo-sensor-ultrasonico-gh311-proximidad-3-pines-_JM#reco_item_pos=0&reco_backend=retrieval-v2p_marketplace&reco_backend_type=low_level&reco_client=vip-v2p&reco_id=57bf8bd8-e92e-4b8b-a147-d9a45a0fa202&r. [Último acceso: 21 10 2024].

[70] R. T. d. Shopify, «RC DRONE - Cuadricóptero de fibra de carbono ZD550,» Ready to sky, 23 02 2022. [En línea]. Available: https://rcdrone.top/es/products/zd550-zd680-kit?srsltid=AfmBOornaU1y2MNBCLeF3vnXMZV5dDTk7Rr4TINy6LzD_S-8uMuOcpTc. [Último acceso: 14 11 2024].

ANEXOS

Anexo A. Listado de abreviaturas

BGC: Brushless Gimbal Controller

BJT: Conector medida estandar

CPU: Unidad Central de Procesamiento

DGAC: Dirección General de Aviación

ECEF: Centrado en la Tierra, Fijo en la tierra

ESC: Control Electrónico de Velocidad

EMP: Error Máximo Permitido

FPV: Video en Primera Persona

GPS: Sistema de Posicionamiento Global

GHz: Giga Hercios

GNSS: Sistema de Navegación por Satelite

HDOP: Horizontal Dilution of Precision

HUD: Heads-up Display

HTML: HyperText Markup Languaje

I2C: Inter-Integrated Circuit

IA: Inteligencia artificial

IGM: Instituto Geográfico Militar

IMU: Unidad de Medidas Inerciales

JSON: JavaScript Object Notation

KML: Lenguaje de marcado de Keyhole

LCD: Pantalla de Cristal Líquido

MAVLink: Micro Air Vehicle Link

MAV: Micro Air Vehicle

MHz: Mega Hercios

SSD: Unidad de Estado Sólido

S/N: Signal Noise

OLED: Diodo Orgánico de Emisión de Luz

PDB: Placa de Distribución de Energía

PID: Controlador Proporcional, Integral y Derivativo

PWM: Modulación por ancho de pulsos

PX4: PixHawk 4

RC: Radio Control

RF: Radio Frecuencia

RGB: Red, Green and Blue

RTL: Return to Launch

RMSE: Error Cuadrático Medio

UAV: Vehículo Aéreo No Tripulado

UART: Universal Asynchronous Receiver-Transmitter

USGC: U.S. Geological Survey

VTX: Video Transmisor

WSN: Redes de Sensores Inalámbricos

Anexo B. Manual del software de configuración SIMPLEBGU de la placa de control del gimbal.



SimpleBGC Software Manual de Usuario

Board ver.: 1.0
Firmware ver.: 2.3
GUI ver.: 2.3

Translated by Air Camera, Spain

Basic Settings. Configuración básica.

Nota: Antes de ajustar su controladora, instale la cámara en el soporte con firmeza y compruebe que el centro de gravedad está centrado con sus respectivos ejes, tanto como sea posible.

- **P, I, D - parámetros de regulación PID para todos los ejes.**
  - P - describe el poder de la respuesta a la perturbación . Los valores más altos significa una reacción fuerte en respuesta a la perturbación externa. Elevar este valor hasta que la calidad de la estabilización ante perturbaciones rápidas sea adecuada. Si el valor " P " es demasiado alto, las oscilaciones del eje comenzarán a estar presentes. Estas oscilaciones se empeoran si hay vibraciones que llegan a la placa del sensor IMU. Si se producen oscilaciones, aumentar el parámetro " D" por 1 o 2 unidades y vuelva a intentarlo, para elevar el valor de "P " de nuevo.
  - D - El valor " D" reduce la velocidad de reacción. Este valor ayuda a eliminar oscilaciones de baja frecuencia. Un valor de "D " que es demasiado alto puede causar oscilaciones de alta frecuencia, sobre todo, cuando el sensor IMU está expuesto a vibraciones.
  - I - Este valor cambia la velocidad a la que se mueve el sistema a comandos RC entrantes y determina la velocidad de retorno a neutral. Los valores bajos dan lugar a una reacción lenta y suave frente a comandos de RC y de volver a la posición neutral . Aumente este valor para acelerar la movimiento.

- **Limit Accelerations** - Esta opción le permite limitar las aceleraciones angulares en caso de control brusco mediante comandos RC o de serie (útil para evitar tirones o pasos saltados, control de cámara más suave , menos impacto en el chasis del multirotor). Cuanto menor es el valor, más suave es la rotación de la cámara bajo control.

- **POWER** - máximo voltaje suministrado a los motores (0 - 255, donde 255 representa el total de la tensión batería). Elija este parámetro de acuerdo a sus características del motor. *Ajuste básico*:
  - ¡Los motores no deben estar demasiado calientes! Temperaturas de motor de más de 80C causarán un daño permanente a los imanes del motor .
  - Un valor de potencia que es demasiado bajo no proporcionará suficiente fuerza para el motor para mover el sistema y estabilizar la cámara adecuadamente. Un valor bajo de POWER será más notable en condiciones ventosas, cuando el sistema no está bien equilibrado, o si el sistema sufre de excesiva fricción mecánica. Baje lentamente el parámetro de POWER para encontrar su valor óptimo. Busque el valor más bajo que proporcione una buena estabilización y par de retención adecuado.
  - El aumento de la potencia es igual a elevar el valor de "P " de los ajustes PID . Si se eleva el valor POWER, debe volver a ajustar los valores de PID también.

- **INVERT** - invertir el sentido de giro del motor. Es muy importante seleccionar el sentido de rotación correcto del motor de para no dañar tu gimbal. Para determinar la direcoión correcta , ajuste los valores P, I, y D a 0 y los valores de potencia a 80 (o más si sus motores no producen suficiente fuerza para mantener / mover la cámara). Nivele el soporte de cámara horizontalmente y haga clic en el botón **AUTO** en los ajustes de "configuración de motor ". El gimbal hará pequeños movimientos para determinar la dirección correcta de rotación del motor . Espere a que el proceso de auto-calibración termine. A continuación, vuelva a configurar los valores PID y a ajustar los valores de potencia (POWER).

- **NUM.POLES** - Número de polos del motor . Este valor tiene que ser igual al número de imanes en la campana de su motor. Durante el proceso de calibración " auto" que se ha descrito anteriormente , este valor se detecta automáticamente. Sin embargo , a veces este valor se determina incorrectamente durante el proceso de calibración "auto" y tendrá que ser verificado y posiblemente corregido manualmente. Muchos motores sin escobillas gimbal se construyen con 14 polos (o imanes) y utilizan un esquema de bobinado DLRK. Cuente los imanes del su motor e introduzca este valor si el valor no es correcto en el GUI.

- **External FC Gain** - valor de ganancia para que coincidan con los datos del gimbal de su controladora de vuelo (FC opcional) . Para una mejor estabilización y utilización de algunas de las características adicionales, se requiere el conocimiento de los ángulos de inclinación del chasis. SimpleBGC IMU no proporciona dicha información. La mayoría de FC tienen salidas de servo para conectar a gimbals (sistemas de movimiento de cámara) . Estas salidas deben conectarse al controlador SimpleBGC través EXT_ROLL y entradas EXT_PITCH .
  - Activar salidas de control de gimbal en la FC y establecer los límites del rango de ángulos que generalmente utiliza (por ejemplo, + -30 grados de inclinación del chasis deben equivaler a un rango de servo completo sobre 1000-2000).
  - Desactivar todos los filtros y suavizado de configuración de la FC (si existieran).
  - En la pestaña **RC-settings**, asegúrese de que las entradas EXT_ROLL , EXT_PITCH no se utiliza para controlar gimbal (es decir, no se elige como fuente de cualquier otra tarea de control RC).
  - En la pestaña **REALTIME DATA**, comprobar la disponibilidad de señales EXT_FC_ROLL, EXT_FC_PITCH y asegurarse de que se asignan a los ejes correctamente. (Un ángulo de +-30º de inclinación debe causar EXT_FC_ROLL cambio de aproximadamente 900 .. 2100ms. Lo mismo para el EXT_FC_PITCH.)

© 2013 Basecamelectronics®

Anexo C. Monitorización desde la estación Base.



Anexo D. Pruebas de vuelo alrededor de los Talleres Tecnológicos.

Como podemos observar en las figuras D1, D2, D3 se representan las pruebas de vuelo en el sector de TALLERES TECNOLOGICO FISEI para determinar su autonomía y capacidad de videovigilancia.



Figura D1. Vuelo del dron observado desde la estación Base.



Figura D2. Monitorización de Videovigilancia a través de GoPro.



Figura D3. Fotografía tomada desde UAV en vuelo.

Anexo E. Pruebas de vuelo alrededor de Garaje Campus Huachi.

En la figura E1 se puede visualizar el vuelo del dron en el estacionamiento del campus huachi de la Universidad técnica de Ambato, y para las figuras E2, E3 podemos observar su operabilidad donde se coordina su autonomía y su videovigilancia en el espacio mencionado.



Figura E1. UAV ubicado en el estacionamiento para sus operaciones.



Figura E2. Vuelo Autónomo del UAV.



Figura E3. UAV cumpliendo la misión establecida.

Anexo F. Prueba de vuelo en el ESTADIO UNIVERSITARIO.



Figura F1. Prueba de trayectoria del Dron autónomo.

Anexo G. Manual de manejo del dispositivo

Para el uso del dispositivo del sistema de navegación autónoma del UAV (hexacoptero) respecto a la investigación realizada e implementada se debe considerar el manejo del dron previamente a tener un conocimiento de vuelo de drones, algunos de los sistemas a considerar son el sistema FPV con el monitor FPV, sistema del gimbal para la cámara Go Pro Hero 5, y manejo del software de planificación de misiones e interfaz de control y monitoreo. Las limitaciones del sistema se tendrán en consideración que es un prototipo realizado para investigación, la estructura y añadidura de otros elementos dependerá de investigaciones futuras a realizar.

- **ESPECIFICACIONES TÉCNICAS**

Algunas de las especificaciones necesarias de forma general de acuerdo a ciertos parámetros que son necesarios para la puesta en funcionamiento y considerar ciertos valores para manejar o configurar el dispositivo se indican algunos de los parámetros en la Tabla G1.

Tabla G1. Especificaciones técnicas del sistema

Parámetro	Descripción
Modelo y tipo de UAV	Hexacóptero Modelo Tarot FY690S
Controlador de vuelo	Pixhawk 2.4.8
Sensores	GPS, IMU, magnetómetro, cámara, brujula, E-Compas OMC3883L, Giroscopio MPU600.
Transmisión de datos	Telemetría (915 MHz / 433 MHz), WiFi.
Frecuencia de video y radio control	5.8 GHz (para transmisión FPV), 2,4 GHz para radio control.
Potencia del modulo de VTX	25, 200, 500 mW con una capacidad de 40 CH
Autonomía de vuelo	Tiempo de operación con carga estándar de 19,30 minutos.
Capacidad de carga	Peso máximo de sensores o cámaras adicionales de un valor de 0,75 gr.
Baterías de alimentación	Dos baterias, Lipo de 6 celdas con 22.2 V 6000 mAh, Lipo 3 celdas con 11.1 V 1500 mAh

Es necesario resaltar el modelo de la placa central controladora de vuelo del UAV, de entre cada uno de los puertos, los protocolos de comunicación y además la utilidad se indica en la Figura G1.



Figura G1. Partes de la placa controladora de vuelo Pixhawk 2.4.8

La asignación de los pines, así como el voltaje de operación a tener en cuenta para cada conector variara entre el número de pines, estos conectores son un modelo BJT esto se muestra en la Tabla G2.

Tabla G2. Asignación de pines para conexión con la placa Pixhawk

Puerto	**Número de Pines**	**Señal**	**Voltaje**
TELEM1, TELEM2 Ports	1 (red)	VCC	+5V
	2 (blk)	TX (OUT)	+3.3V
	3 (blk)	RX (IN)	+3.3V
	4 (blk)	CTS	+3.3V
	5 (blk)	RTS	+3.3V
	6 (blk)	GND	GND
SERIAL 4/5 Port	1 (red)	VCC	+5V
	2 (blk)	TX (#4)	+3.3V
	3 (blk)	RX (#4)	+3.3V
	4 (blk)	TX (#5)	+3.3V
	5 (blk)	RX (#5)	+3.3V
	6 (blk)	GND	GND
GPS port	1 (red)	VCC	+5V
	2 (blk)	TX (OUT)	+3.3V

Puerto	Número de Pines	Señal	Voltaje
	3 (blk)	RX (IN)	+3.3V
	4 (blk)	CAN2 TX	+3.3V
	5 (blk)	CAN2 RX	+3.3V
	6 (blk)	GND	GND
ADC 6.6V¶	1 (red)	VCC	+5V
	2 (blk)	ADC IN	up to +6.6V
	3 (blk)	GND	GND
ADC 3.3V	1 (red)	VCC	+5V
	2 (blk)	ADC IN	up to +3.3V
	3 (blk)	GND	GND
	4 (blk)	ADC IN	up to +3.3V
	5 (blk)	GND	GND
I2C	1 (red)	VCC	+5V
	2 (blk)	SCL	+3.3 (pullups)
	3 (blk)	SDA	+3.3 (pullups)
	4 (blk)	GND	GND
CAN	1 (red)	VCC	+5V
	2 (blk)	CAN_H	+12V
	3 (blk)	CAN_L	+12V
	4 (blk)	GND	GND
SPI	1 (red)	VCC	+5V
	2 (blk)	SPI_SCK	3.3V
	3 (blk)	SPI_MISO	+3.3V
	4 (blk)	SPI_MOSI	+3.3V
	5 (blk)	!SPI_NSS	+3.3V
	6 (blk)	!GPIO	+3.3V
	7 (blk)	GND	GND
POWER	1 (red)	VCC	+5V
	2 (blk)	VCC	+5V
	3 (blk)	CURRENT	up to +3.3V
	4 (blk)	VOLTAGE	up to +3.3V
	5 (blk)	GND	GND
	6 (blk)	GND	GND
SWITCH	1 (red)	VCC	+3.3V
	2 (blk)	!IO_LED_SAFETY	GND
	3 (blk)	SAFETY	GND
Spektrum/DSM Port	1 (white)	Signal	+3.3V
	2 (black)	GND	GND
	3 (red)	VCC	+3.3V

- **COMPONENTES DEL SISTEMA**

Se detalla algunas de las partes principales del sistema para tener en constancia que elementos son utilizados en el UAV (Hexacoptero), se dmuestra algunos de los componentes como se muestra en la Tabla G3.

Tabla G3. Elementos del sistema

Componente	Descripción
Hardware	Placa Controladora de vuelo, motores brushless, helices de fibra de carbono, ESC, PDB, baterías Lipo, chasis, modulo GPS, radio Transmisor.
Sistema de navegación	GPS / GNSS
Comunicación	Módulos de radio (telemetría), transmisión de video FPV y radio control.
Software de control	Mission Planner ó Programa en python.
Plataforma de videovigilancia y operaciones aéreas	Monitoreo en tiempo real, almacenamiento en la estación terrena.

- **PROCEDIMIENTOS DE OPERACIÓN**

Encendido y Configuración Inicial

Para iniciar con la puesta en marcha del dron es necesario algunos pasos para evitar accidentes y no tener problemas durante el vuelo, estos pasos se indican a continuación:

1. Verificar nivel de baterías es necesario tener en cuenta que las baterías Lipo de acuerdo al valor tendrán un valor mínimo y máximo de carga, este será 25.2 V valor máximo y 19,4 V valor mínimo para el UAV y para el gimbal de 12.9 V valor máximo y 10.1 V valor mínimo, considerando tener un nivel de 22.9 V como valor mínimo para realizar vuelos, además de verificar el nivel de la batería del monitor FPV y la cámara del gimbal.

2. Verificación de conexiones correctas de todos los sistemas del UAV, ajuste de hélices correcto además se verifica si están en buen estado para el vuelo y verificar la conexión de módulo de telemetría visualizando los indicadores LED de los módulos que no titile el led Verde y el radio control con el UAV que no indique ningún mensaje de emergencia.

3. Encendido del UAV conexión con la alimentación a baterías, puesta y encendido de la cámara para verificar el nivel estabilizado de la cámara en el caso de requerir calibrar se debe realizar lo que se indica en el apartado Calibración del gimbal

mediante el software SimpleBGC_GUI_2.2 y montaje del gimbal al UAV., y control por el radio control de los servos motores estable, establecer la conexión del monitor FPV con el sistema FPV, esperar la inicialización de los sensores y módulo GPS, además de la verificación de la conexión con la estación terrena y el UAV.

4. Calibración del UAV, utilizando la herramienta de Mission Planner para la configuración, es muy necesario para un vuelo óptimo calibrar todos los sensores del dron esto se explica en el apartado de configuraciones en Calibración completa del UAV con el software Mission Planner para la puesta en marcha.

5. Revisión de parámetros de vuelo en la interfaz de control de Python, estas consideraciones son el número de satélites receptados es necesario mencionar que un valor mínimo es de 8 Satélites y un valor máximo de 14 Satélites para un vuelo óptimo, además de considerar las condiciones climáticas y ambientales en el lugar donde se vuele el dron.

6. Revisión de modos de vuelo ajustados y en correcto funcionamiento del UAV, en este caso específicamente el modo de vuelo Automático o AUTO requerido para cumplir la misión planificada.

Planificación de la Misión

Se requiere la planificación de la trayectoria para empezar con la misión deseada para el dron esto se ubica en la parte de la interfaz del dron de control se encuentra alguno de los apartados adicionales esto como se muestra en la Figura G2, para aquello se utiliza la simulación del sistema de georreferencia se empieza ejecutando el programa se ingresa el archivo JSON en formato MISSION que contiene las coordenadas geográficas como los comandos que entiende la placa controladora de vuelo algunos de los más utilizados se explica en el apartado de Configuración de rutas autónomas para la planificación de misiones para el UAV en el software Mission Planner, los datos esenciales a recopilar son latitud, longitud, altura y comando a utilizar entre despegue, waypoint y aterrizaje como se muestra en la Figura G3.

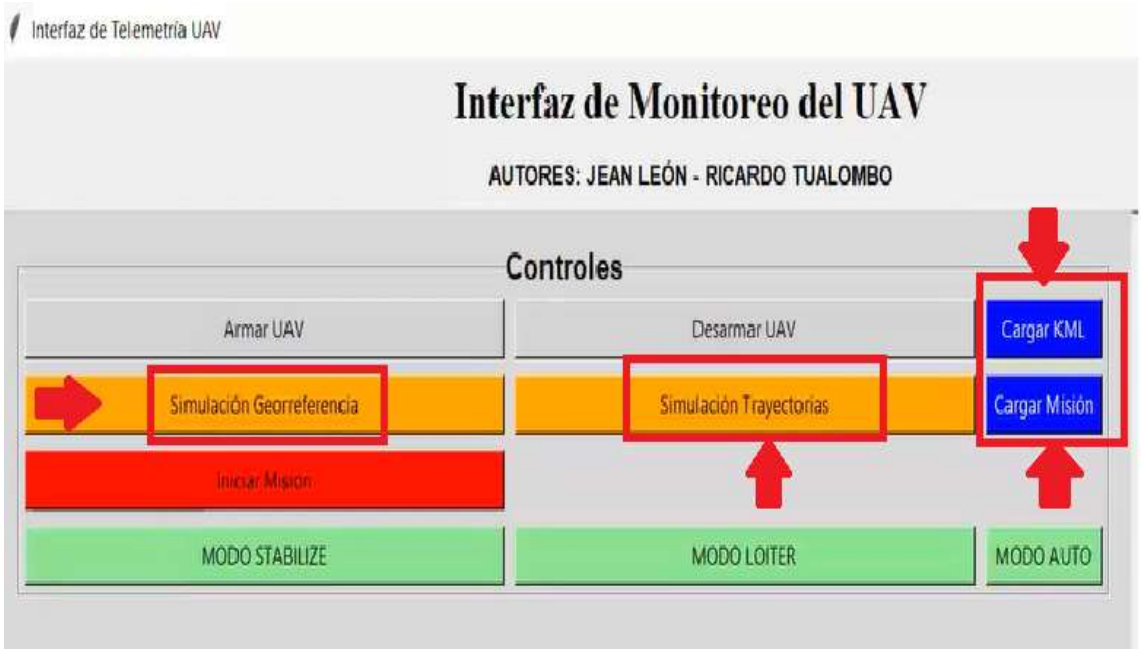


Figura G2. Interfaz de control del dron.

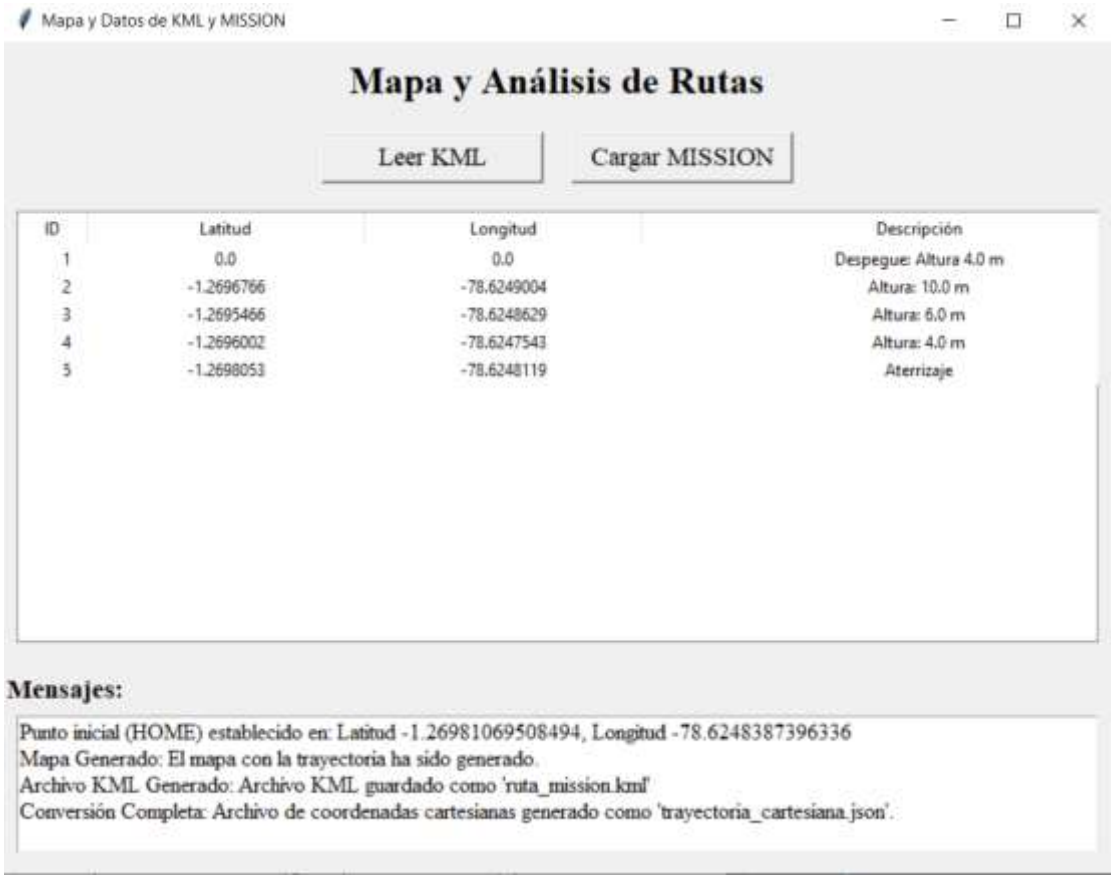


Figura G3. Interfaz de simulación de sistema de georreferencia.

El programa a su vez genera archivos KML y HTML de la trayectoria a realizar para visualizarlo en el mapa de Open Street Maps y Google Earth como se muestra en la

Figura G4, además para verificación de datos y puntos los cuáles se establece en la misión, los archivo KML generado sirven para cargar la misión al UAV.

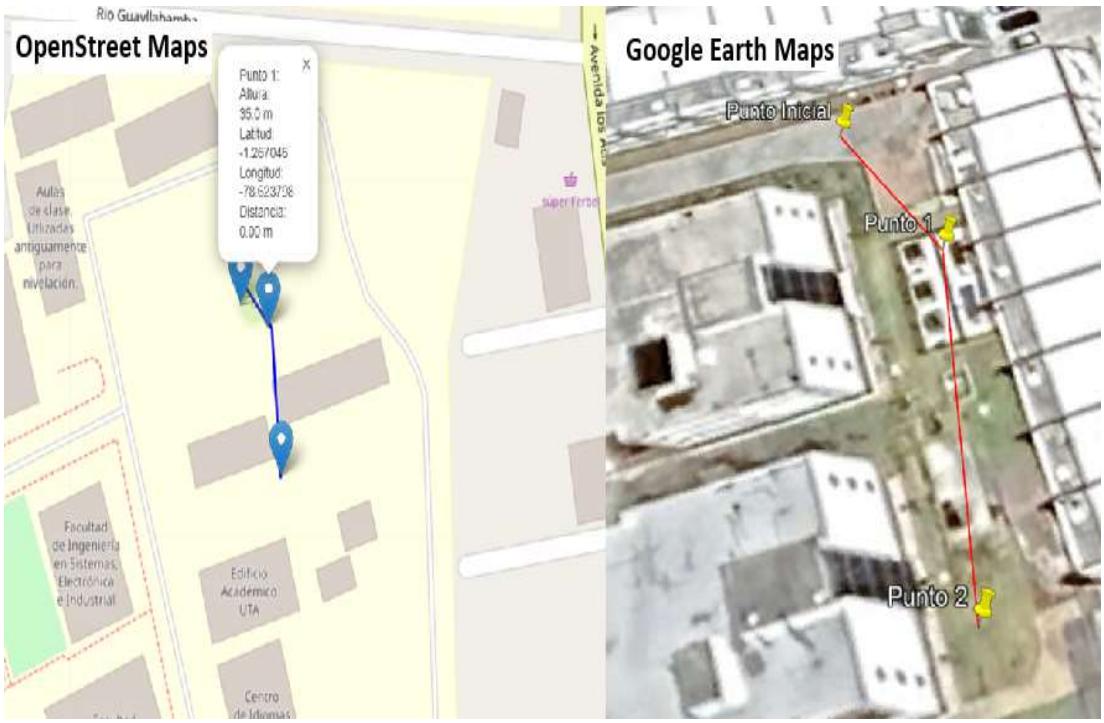
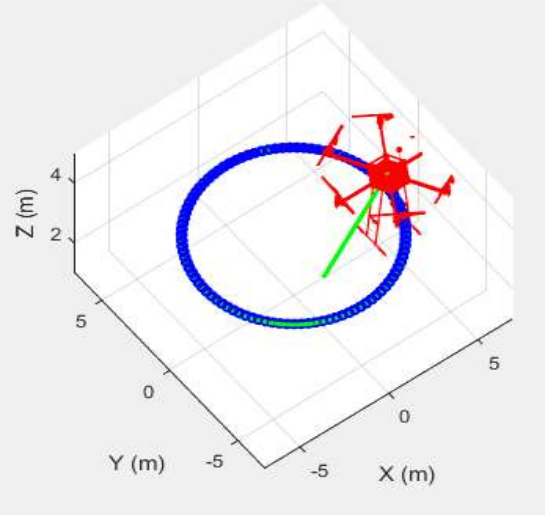


Figura G4. Simulación de mapas del sistema de georreferencia.

También se puede ejecutar la simulación de la trayectoria y verificar el comportamiento del mismo en un ambiente ideal, esto muestra gráficas de la trayectoria cumplida y el comportamiento en el trascurso de la trayectoria del dron para un previo análisis como se muestra en la Figura G5.



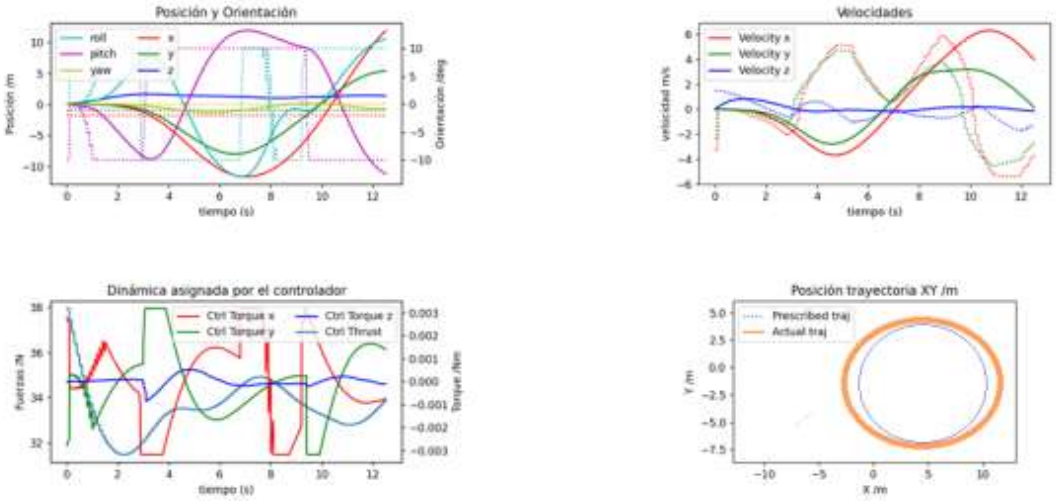


Figura G5. Simulación de trayectorias.

Una vez se realiza la planificación y simulación de trayectorias de la misión previa establecida es necesario cargarlo a la placa esto utilizando el botón “CARGAR KML” archivo KML con los puntos y datos establecidos de la misión, cargando para verificar cuantos puntos tendrá para la verificación y se carga con el botón de “CARGAR MISIÓN” de la interfaz verificando los mensajes de carga como se muestra en la Figura G6.

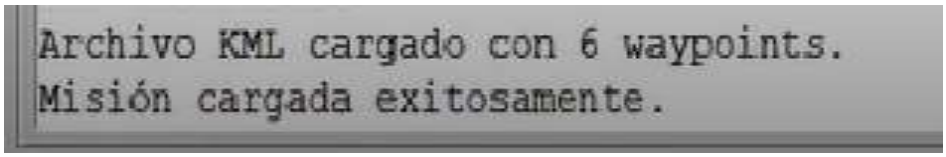


Figura G6. Mensajes de verificación de cargar de misión al UAV.

Ejecución del Vuelo

Para iniciar con la misión una vez se ha planificado la misión, también se toma en cuenta las condiciones ambientales y el ajuste además algunos parámetros del UAV como la velocidad en modo automático, modos de vuelo en correcto funcionamiento, armado y desarmado del dron en modo estabilizado, es de suma importancia el haber establecido del punto HOME en donde estará el UAV en el punto inicial para que en caso de problemas del GPS, se active de forma automática el protocolo de emergencia modo RTL o Return to Launch asegurarse de que está configurada a una altitud adecuada a la misión previamente planificada de forma que tenga un retorno seguro,

esto se configura con el software Mission Planner en Config > Full Parameter List ingresando RTL, y configurando el parámetro de altitud en metros.

Se requiere quitar el modo seguro de motores utilizando el switch conectado a la placa de control y equipado en el dron para desactivarlo este switch como se visualiza en la Figura G7, se empieza por ARMAR o encender los motores del UAV esto en modo Estabilizado o "STABILIZE", para después utilizar el botón de color rojo de "INICIAR MISIÓN" de la interfaz de control para que el dron inicie con la misión cargada pasando a modo AUTO de forma automática e inicie el comando Inicio de misión.

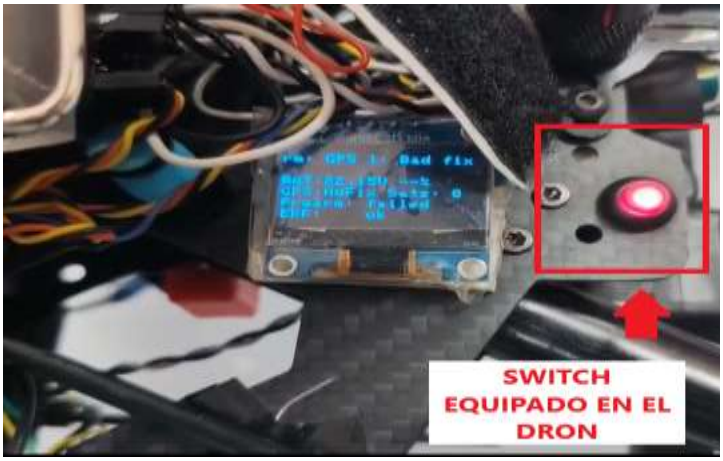


Figura G6. Mensajes de verificación de cargar de misión al UAV.

Aterrizaje y Apagado

Una vez se finaliza la misión en el último punto establecido como LAND o aterrizaje el dron seguirá los siguientes pasos que se indican a continuación:

1. Esperar la estabilización del UAV o desarmar los motores una vez se llegó a tierra con el botón "DESARMAR UAV" de la interfaz de control, en el caso de no hacerlo se desarmarán los motores una vez se encuentre en tierra en algunos segundos.

2. Cambiar a modo "Stabilize" para iniciar otra posible misión o tener control del manejo del dron.

3. Activar el modo seguro de motores con el switch integrado en el UAV, para no tener posibles accidentes con las hélices del dron.

4. Desconectar la alimentación de las baterías Lipo del UAV y del sistema del gimbal, apagar el monitor FPV y la cámara del gimbal.

• SEGURIDAD Y PROTOCOLOS DE EMERGENCIA

La seguridad al momento de manipular un hexacoptero es esencial ya que tiene un tamaño considerable, además de utilizar unas hélices de fibra de carbono que son mucho más resistentes y pueden causar accidentes. Para aquello se requirió establecer algunos protocolos o caso de emergencia de acuerdo a situaciones de concurrencia en una misión algunos de aquellos se pueden visualizar en la Tabla G4.

Tabla G4. Protocolos de emergencia de acuerdo a situaciones.

Situación	Protocolo de Seguridad
Visualización de vuelo inestable del dron	Activar modo manual ó Stabilize y guiar el dron manualmente por el piloto.
Perdida de señal de GPS	Activación automática de Return to Launch (RTL) o retorno al punto de partida (Punto HOME).
Batería baja	Activación automática de Return to Launch (RTL) o retorno al punto de partida (Punto HOME).
Zona de vuelo restringida	Configurar geocerca en software de control.
Interferencia en transmission de video	Cambiar canal en el sistema FPV (5.8 GHz).
Vibraciones excesivas de la camara del gimbal	Ajuste de vibraciones de la placa de control del gimbal

• MANTENIMIENTO Y SOLUCIÓN DE PROBLEMAS

En el caso de requerir mantenimiento para el sistema completo se puede tener problemas de acuerdo a ciertos componentes para aquello se especifican algunos mantenimientos recomendados esto se muestra en la Tabla G5.

Tabla G5. Mantenimiento de componentes.

Componente	Mantenimiento Recomendado
Batería	Carga y balanceo regular, evitar descarga total.
Motores y hélices	Inspección visual antes de cada vuelo.
Sensores	Calibración antes de cada vuelo con la herramienta de software libre Mission Planner.
Transmisión de video	Verificar conexiones de antena y señal FPV.
Gimbal	Calibración periódica del gimbal de acuerdo al peso de la camara a utilizar.

Diagnóstico de Fallos Comunes

De acuerdo a los resultados conseguidos de las pruebas realizadas en el sistema de navegación autónoma del UAV para videovigilancia y operaciones aéreas, se tiene algunos fallos que se visualizaron y se pudieron corregir para aquello se muestra los diagnósticos de los mismos en la Tabla G6.

Tabla G6. Diagnostico a fallos de las pruebas realizadas del sistema

Problema	Causa Posible	Solución
GPS no conecta	Baja señal o interferencia	Cambiar ubicación, reiniciar GPS, condiciones ambientales no aptas para el vuelo.
Vuelos Inestables del UAV	Ajuste de parámetros incorrectos	Ajuste de PID, velocidad de motores, estructura del chasis o partes del UAV en mal estado.
Calibración incorrecta de sensores	Sensores en mal estado	Verificación de estado de los sensores en el software Mission Planner.
Motores no arrancan	Failsafe activado	Revisar mensajes de alerta en el software Mission Planner.
	Errores de calibración	Calibrar de forma correcta todos los sensores del UAV
	Conexión de elementos incorrecta	Verificar la conexión del radio control, conexión del modulo de telemetría número de baudios y puerto incorrectos, sensores.
	Errores en sensores	Revisar mensajes de alertar en el software Mission Planner.
Pérdida de video	Interferencia en 5.8 GHz	Cambiar frecuencia o usar antena de mayor ganancia.

Anexo H. Comparativa de tecnologías y técnicas empleadas para sistemas de navegación autónoma

Tabla H1. Tabla comparativa de proyectos

PARÁMETRO	**Investigación 1**	**Investigación 2**	**Investigación 3**	**Investigación 4**	**Investigación 5**	**Investigación 6**	**Investigación 7**	**Investigación 8**	**Investigación 9**	**Investigación 10**	**Investigación 11**
Título del Estudio	Diseño e implementación de un drone autónomo para el monitoreo de una red wsn de sensores meteorológicos utilizando dispositivos xbee para transmisión de datos en la espoch	Sistema de teledetección aérea para control de agentes patógenos y enfermedades en cultivos de brócoli con el uso de visión artificial	Sistema autónomo de pulverización para fumigación de plantaciones de frutilla asistido por un drone	Navegación autónoma de drones y automatización de rutas aplicadas a la limpieza de edificios	Navegación autónoma de un dron para localizar un transmisor de radio frecuencia basado en aprendizaje por refuerzo	Red de drones autónomos utilizando una arquitectura de red (fanet) para uso alternativo de levantamiento de información agrícola a pequeña escala	Diseño y construcción de drone autónomo para medición de radiación solar	Dron de vuelo autónomo con reconocimiento basado en inteligencia artificial.	Diseño y evaluación de un dron autónomo basado en navegación por GPS	Navegación Autónoma y Evasión de Obstáculos en UAV usando Aprendizaje por Refuerzo	Deteccion de obstaculos durante vuelo autonomo de drones utilizando SLAM monocular
Año de Publicación	2017	2023	2019	2020	2021	2023	2018	2020	2018	2019	2020
Tipo de UAV	Drone Cheerson CX-20	Cuadricóptero Realacc x210Pro	DJI Phantom 3 Estándar	Multirrotor	VTOL	Parrot Disco Pro-Ag, F450	Cuadricóptero	Cuadricóptero	VTOL	Hexacoptero	Cuadrirotor
Controladora	Arduino Mega 2560	MASTKSYS FLIGHT CONTROLLER F405-STD	DJI N3	Pixhawk 4	ArduPilot	CPU Dual core ARM Cortex -A9, NAZA V2.	Ardupilot APM 2.6	NAVIO2	Arduino DUE	Ardupilot Pixhawk 2.4.8	BEBOP, parrot

PARÁMETRO	Investigación 1	Investigación 2	Investigación 3	Investigación 4	Investigación 5	Investigación 6	Investigación 7	Investigación 8	Investigación 9	Investigación 10	Investigación 11
Microcontrolador o Procesador	ATM2560	F405-STD	DJI Mobile SDK	STM32F427	ARM Cortex-A53	Uso de Laptop Dell Inspiron 15	Controladora basada en la plataforma Arduino	Rasperry pi 3	ARM Cortex-A53	Controladora basada en una JETSON NANO	Una Laptop con procesador Intel Core i7-4720HQ
Sistema de Sensores	Sensores de temperatura LM35; Acelerómetros y giroscopios integrados en el sistema de control.	Cámara GoPro Hero 5	Sin sensores	Cámara FPV	Cámara RGB, GPS, IMU	Cámara multiespectral SEQUOIA	Acelerómetro de 3 ejes, Giroscopio de 3 ejes, Barómetro MEAS MS5611 y GPS	Cámara RPI-CAM-V2	Cámara RGB, GPS, IMU	Acelerómetro de 6 ejes, Giroscopio de 6 ejes, Barómetro MEAS MS5611 y GPS MN8	Camara FPV
Técnicas de control del UAV	Control tradicional mediante electrónica con el software Mission Planner.	Control mediante IA utilizando algoritmos en python con librerías de YOLO V5	Control mediante IA utilizando un algoritmo con App de Adroid Studio	Control tradicional mediante un algoritmo de programación en Python utilizando librerías de dronekit	Control mediante IA con algoritmos por Reforcing Learning como ANAIV Y RL	Control mediante IA con algoritmos de programación como D* Lite, SLAM	Control tradicional mediante electrónica en un algoritmo de control en cascada	Control mediante IA utilizando algoritmos con librería de python Tensorflow API	Control tradicional mediante algoritmos de programación como A*	Control mediante IA utilizando algoritmos por Reforcing Learning	Control mediante IA con algoritmos de programación como ORB-SLAM
Sistema de Evasión de Obstáculos	Vuelo autónomo en una ruta preprogramada	Sin sistema	Sin Sistema	LiDAR con RRT*	Cámara con detección visual de obstáculos	LiDAR	Sin sistema	Inteligencía Artificial aplicando redes neuronales Transfer learning modelo SSD	Cámara con detección visual de obstáculos	Sin sistema	Camara FPV

PARÁMETRO	Investigación 1	Investigación 2	Investigación 3	Investigación 4	Investigación 5	Investigación 6	Investigación 7	Investigación 8	Investigación 9	Investigación 10	Investigación 11
Aplicación y tamaño de Ámbito de pruebas	Para comunicación en una red de sensores inalámbrica (WSN) .	Agricultura	Agricultura	Limpieza de edificios en Áreas Urbanas	Detección de señales RF	Levantamiento de información agricola	Industrias de materiales radiactivos	Modelo de aprendizaje en asignaturas de IA	Para fotografía en Campos agrícolas	Industrias de materiales radiactivos	Campo práctico para investigación
Precisión de Navegación	30 cm	40 cm	30 cm	10 cm	20cm	20 cm	10 cm	N/S	20cm	10 cm	40-60 cm
Duración de Vuelo	45 minutos	4 minutos	10 minutos	25 minutos	18 minutos	7 minutos	N/S	18 minutos	30 minutos	N/S	10 minutos
Tecnología	Módulos Xbee Serie 2 para red de sensores inalámbricos (WSN).	YOLO v5x	Protocolo Bluetooth	Protocolo MAVlink	ROS	N/S	Raspberry Pi con módulo de cámara y sistema de seguimiento solar	Tensorflow API	ROS	Raspberry Pi con módulo de cámara y sistema de seguimiento solar	ROS
Eficiencia del Sistema	80%	91.9%	90% Para pulverización	92%	85%	90%	80%	80%	85%	80%	80%