



FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

IMPLEMENTACIÓN DE UN ROBOT DE NAVEGACIÓN AUTÓNOMA
TERRESTRE PARA EVASIÓN DE OBSTÁCULOS USANDO EL
DISPOSITIVO PIXHAWK

AUTORES

ARIANA MICHELLE MERA CHAMORRO
DAVID ALEJANDRO RUANO GONZÁLEZ

AÑO

2019



FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

IMPLEMENTACIÓN DE UN ROBOT DE NAVEGACIÓN AUTÓNOMA
TERRESTRE PARA EVASIÓN DE OBSTÁCULOS USANDO EL DISPOSITIVO
PIXHAWK

Trabajo de titulación presentado en conformidad con los requisitos establecidos
para optar por el título de Ingeniera en Electrónica y Redes de Información e
Ingeniero en Redes y Telecomunicaciones

Profesor Guía

MSc. Jorge Luis Rosero Beltrán

Autores

Ariana Michelle Mera Chamorro

David Alejandro Ruano González

Año

2019

DECLARACIÓN DEL PROFESOR GUÍA

“Declaro haber dirigido este trabajo, implementación de un vehículo terrestre de navegación autónoma y evasión de obstáculos, a través de reuniones periódicas con los estudiantes Ariana Michelle Mera Chamorro y David Alejandro Ruano González, en el semestre 201910, orientando sus conocimientos y competencias para un eficiente desarrollo del tema elegido y cumpliendo con todas las disposiciones vigentes que regulan los Trabajos de titulación”.

Jorge Luis Rosero Beltrán
Master en Ciencias con Especialidad en Automatización
C.I. 1803610185

DECLARACIÓN DEL PROFESOR CORRECTOR

"Declaro haber revisado este trabajo, implementación de un vehículo terrestre de navegación autónoma y evasión de obstáculos, de los estudiantes Ariana Michelle Mera Chamorro y David Alejandro Ruano González, en el semestre 201910, dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación".

David Fernando Pozo Espín
Master en Automática y Robótica
C.I. 1717340143

DECLARACIÓN DE AUTORÍA DEL ESTUDIANTE

“Declaramos que este trabajo es original, de nuestra autoría, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de autor vigentes.”

Ariana Michelle Mera Chamorro
C.I. 0401872288

David Alejandro Ruano González
C.I. 1722690763

AGRADECIMIENTOS

En mi primer lugar doy gracias a Dios por haberme dado la fuerza y valor para alcanzar esta meta. Agradezco a mi familia por toda su confianza y apoyo.

Agradezco también al Ingeniero Jorge Rosero por toda la colaboración brindada para desarrollar este trabajo.

Ariana Mera

DEDICATORIA

Dedico este trabajo a mis padres por todo su amor, esfuerzo y sacrificio, gracias a ellos he logrado llegar hasta aquí. A mis hermanos Carla y Carlos por su cariño y apoyo incondicional, por estar conmigo en todo momento.

Para todos ustedes con cariño toda mi constancia y esfuerzo plasmado en este trabajo. Gracias por ser mi motivación.

Ariana Mera

AGRADECIMIENTOS

Primeramente, le agradezco a mi universidad, que a lo largo de cinco años me aportó un sin número de enseñanzas, me ha hecho madurar con cada vivencia y conocimiento adquirido.

Agradezco también de manera muy especial al Ingeniero Jorge Rosero, quién nos motivó de gran manera para culminar con éxito este proyecto de titulación.

David Ruano

DEDICATORIA

Dedico este trabajo de titulación a mis padres y hermana quienes han sido un ejemplo de superación tanto en lo personal como en lo profesional, guiándome con amor y con mucha sabiduría en cada paso y metas que me he propuesto.

Además, le dedico a mi sobrino Joaquín, quién es la alegría, luz, y me llena de energía para seguir adelante.

David Ruano

RESUMEN

Anteriormente los sistemas de navegación autónoma se encontraban en su mayoría presentes en aeronaves y vehículos militares no tripulados, debido a su alto costo. Actualmente y gracias a los avances tecnológicos en sistemas de posicionamiento, sistemas de medición inercial, hardware de cómputo, y software de navegación es posible conseguir en el mercado una variedad de controladores de navegación, los cuales integran todos estos sistemas mencionados. Pixhawk es un estándar en hardware de navegación que nació como un proyecto estudiantil hace más de diez años. Actualmente Pixhawk ha desarrollado una variedad de controladores sobre los cuales es posible cargar un software de piloto automático, los más recomendables y más usados son ArduPilot y PX4, ambos de código abierto.

En este trabajo de titulación se realiza el estudio y análisis de la tarjeta de navegación Pixhawk Cube, comprendiendo su funcionamiento y el de todos sus componentes. El segundo paso es la implementación de un vehículo terrestre no tripulado también conocido como UGV o *rover*, usando el controlador de navegación estudiado anteriormente y el software de navegación ArduRover. El UGV es capaz de realizar el seguimiento de puntos de una trayectoria, los cuales son marcados por coordenadas geográficas. Además, en el caso que existan obstáculos en la trayectoria el UGV es capaz de detectarlos, y evadirlos cambiando su trayectoria. La evasión de obstáculos se realiza con un sensor de distancia LiDAR, la configuración de parámetros y modificación del código ArduRover.

Se realizan distintas pruebas para verificar el correcto funcionamiento del vehículo, mediante el uso de imágenes satelitales se fija una serie de coordenadas geográficas, las cuales son enviadas al controlador de navegación. EL UGV realiza el recorrido desplazándose sobre los puntos de la trayectoria en una superficie plana con obstáculos. Además, durante las pruebas se realizan los ajustes necesarios.

ABSTRACT

Previously, autonomous navigation systems were mostly present in aircraft and unmanned military vehicles, due to their high cost. Currently, thanks to technological advances in positioning systems, inertial management systems, computer hardware, and navigation software, a variety of navigation controllers are available in the market, which integrate all these systems. Pixhawk is a standard in navigation hardware that was born as a student project more than ten years ago. Currently, Pixhawk has developed a variety of drivers in which the autopilot software can be loaded, the most recommended and most used are ArduPilot and PX4, both open source.

In this titling work, the study and analysis of the Pixhawk Cube navigation card is carried out, including its operation and of all its components. The second step will become an unmanned ground vehicle also known as UGV or rover. With the previously studied navigation controller using the ArduRover navigation software. The UGV will be able to track the points of a trajectory, which will be marked by geographical coordinates. In addition, in the case of finding obstacles, the UGV will be able to detect them, and evade them by changing its trajectory. Obstacle avoidance will be done by a distance sensor, the rover parameters configuration and the modification of the ArduRover code.

Different tests are performed to verify the correct operation of the vehicle, by using satellite images, a series of geographical coordinates is fixed, which are sent to the navigation controller. The UGV travels along the points of the trajectory on a flat surface with obstacles. In addition, during the tests, necessary adjustments are made.

ÍNDICE

1. Introducción.....	1
1.1. Justificación	2
1.2. Alcance	3
1.3. Objetivos.....	3
1.3.1. Objetivo General.....	3
1.3.2. Objetivos Específicos	3
1.4. Metodología	4
2. Marco Teórico	4
2.1. Vehículo Terrestre no Tripulado.....	5
2.2. Controlador de Navegación	5
2.2.1. Interfaces de comunicación	6
2.3. Piloto automático	7
2.4. Telemetría	8
2.5. Estación terrena.....	8
2.6. Codificador PPM-Sum.....	9
2.7. Sistema de Radio Control.....	9
2.8. GPS/Brújula	10

2.9. Sensores de proximidad	10
2.10. Controlador o Driver de actuadores	10
2.11. Batería.....	11
2.12. Módulo de distribución de energía	11
2.13. Actuadores	12
2.14. Chasis	12
3. Diseño.....	12
3.1. Diseño de Hardware	12
3.1.1. Chasis.....	13
3.1.2. Controlador de navegación.....	14
3.1.3. Sistema de Radio Control.....	30
3.1.4. Telemetría.....	34
3.1.5. Batería	35
3.1.6. Codificador.....	40
3.1.7. Controlador o Driver de actuadores.....	41
3.1.8. Sensor de proximidad.....	43
3.1.9. Actuador del mecanismo de rotación del sensor de proximidad...	46
3.1.10. Piezas de montaje	47
3.2. Diseño de software.....	49
3.2.1. Piloto Automático.....	49

3.2.2. Estación terrena.....	53
3.2.3. Configuración de Telemetría.....	55
3.2.4. Configuración Inicial del control de navegación.....	60
3.2.5. Configuración Avanzada.....	67
3.2.6. Evasión de obstáculos.....	71
4. Pruebas y Resultados.....	77
4.1. Armar el UGV.....	77
4.2. Control Manual.....	78
4.3. Navegación Autónoma.....	80
4.4. Precisión GPS.....	84
4.5. Evasión de obstáculos.....	87
5. CONCLUSIONES Y RECOMENDACIONES.....	90
5.1. Conclusiones.....	90
5.2. Recomendaciones.....	91
REFERENCIAS.....	93

1. Introducción

La idea de los vehículos no tripulados inicio desde principios del siglo XX, sin embargo, eran proyectos netamente militares, pero gracias a las múltiples investigaciones y avances tecnológicos, actualmente es un tema que está al alcance y puede ser utilizado en diferentes aplicaciones. Comúnmente a los drones se les identifica como vehículos aéreos no tripulados, pero este término agrupa a cualquier tipo de vehículo, ya sea vehículos aéreos hasta vehículos terrestre y sumergibles. Las investigaciones actuales se centran en el desarrollo de aplicaciones que buscan crear robots con sistemas embebidos que les permitan realizar tareas de forma autónoma (Delgado, 2016).

Existen grandes avances sobre los vehículos terrestres no tripulados como el que se menciona en el artículo. - “Farmers Reap New Tools From Their Own High-Tech Tinkering”, publicado en mayo del 2016 por el diario “The Wall Street Journal”, esta trata acerca de la implementación de un tractor de agricultura autónomo desarrollado por Matt Reimer’s, el cual utiliza un controlador de navegación Pixhawk. “El tractor es un John Deere de ocho años que el Sr. Reimer de 30 años modificó con piezas de drones, software de código abierto y una tableta de Microsoft Corp como controlador de tierra” (Bunge, 2016). Este proyecto ayuda en la cosecha de trigo y representa un ahorro para Reimer’s en el salario de los trabajadores encargados de manejar el tractor.

Pixhawk se originó en el año 2008 como un proyecto estudiantil en ETH Zurich para el control de vuelo autónomo. Como se menciona en la página oficial “Pixhawk es un proyecto independiente de hardware abierto que ofrece diseños de hardware de piloto automático de alta disponibilidad, bajo costo y de gama alta para las comunidades académicas, hobbies e industriales” (Meier, 2018).

En diciembre del 2017 fue presentado en la Revista del Diseño Innovativo, de la Organización Internacional de Ciencia y Tecnología ECOFRAN el Artículo. Instrumentación y desarrollo de una plataforma de vehículo terrestre para la obtención de posicionamiento RTK, con el objetivo de otorgar a un vehículo terrestre las herramientas necesarias para desarrollar aplicaciones de

autonomía. Estas herramientas o conjuntos de sensores de posición, orientación y aceleración superan al sensor de GPS uno de los más utilizados a nivel mundial, debido a su poca precisión para aplicaciones donde se manejan distancias en unidades de centímetros (Rodríguez, Trejo y Espinoza, 2017).

1.1. Justificación

Los avances tecnológicos en el campo de los vehículos no tripulados no solo se limitan a los vehículos aéreos no tripulados UAV, sino que también a los vehículos terrestres no tripulados UGV y los vehículos acuáticos no tripulados USV.

Los sistemas autónomos pueden ser utilizados en diferentes aplicaciones tanto en el campo militar como en el campo civil. En el campo de la industria civil podemos mencionar aplicaciones en la agricultura, captura de fotografía y video, transporte de carga, búsquedas, reconocimientos y exploraciones de terreno, recolección de basura tanto en lugares abiertos como cerrados. En el campo militar los sistemas autónomos podrían ser de gran apoyo en labores como, determinar áreas peligrosas para los soldados, proporcionar vigilancia, transportar suministros, fijar objetivos, desactivación de artefactos explosivos, resguardo de zonas fronterizas (Delgado, 2016).

El principal problema que motivo a la realización de este trabajo es la cantidad de desechos que se generan en parques, calles y en general lugares donde hay gran afluencia de personas, esto afecta al cuidado del ambiente y a la imagen por lo que es necesario se realicen tareas de limpieza diariamente. El compromiso de mitigar el impacto ambiental y crear conciencia de la contaminación es un problema que involucra a toda una sociedad y es importante adoptar medidas que aporten un valor agregado en el cuidado del planeta. Por todo lo mencionado se propone como aplicación un sistema para la recolección de basura tomando como base el UGV que se desarrolla en este proyecto.

1.2. Alcance

En este trabajo de titulación se desarrolla la implementación de un prototipo de vehículo terrestre no tripulado (UGV) que tiene la capacidad de desplazarse sobre una superficie plana con obstáculos. El proyecto cuenta con los siguientes componentes:

- Una estructura que aloja los diferentes componentes de comunicación, sensores, hardware de piloto automático, microcontroladores y un sistema de tracción.
- Estación terrena desde donde se planificará la misión del vehículo.
- Sistema de comunicación entre el UGV y la Estación Terrena.
- Sensores de proximidad para la evasión de obstáculos.
- Algoritmo de navegación automática que será el controlador del desplazamiento de vehículo terrestre.

1.3. Objetivos

1.3.1. Objetivo General

Implementar un prototipo de vehículo terrestre autónomo que integre sistemas de navegación y evasión de obstáculos para ser utilizado en aplicaciones que requieran seguimiento de trayectorias.

1.3.2. Objetivos Específicos

- Aprender el funcionamiento del controlador de navegación Pixhawk Cube que proporcionará las bases necesarias para la implementación del prototipo.
- Implementar un vehículo terrestre que integre un sistema de navegación y sistema de evasión de obstáculos a través del uso de sensores.
- Realizar pruebas y ajustes necesarios para el correcto funcionamiento del vehículo terrestre.

1.4. Metodología

Una correcta definición de los métodos a utilizar garantiza el cumplimiento de los objetivos planteados en este trabajo. Se utiliza el método inductivo, exploratorio y experimental ya que se trata de un proyecto donde se requiere investigación, pruebas y la integración de varios componentes.

El método inductivo conduce a cumplir con el objetivo final que es desarrollar un vehículo terrestre autónomo basándose en una investigación y experimentación inicial, cada uno de los componentes que integraran el proyecto se estudia de manera independiente hasta llegar a unificarlos. El método exploratorio, permite la familiarización con las diferentes tecnologías necesarias para el desarrollo de los sistemas de navegación y cada uno de sus subcomponentes que lo integran. Además, ayuda a tener un conocimiento inicial acerca del proyecto a realizar. El método experimental se usa en cada una de las fases de desarrollo del proyecto para realizar pruebas y correcciones donde se observa si los resultados obtenidos son los esperados.

2. Marco Teórico

En este capítulo se presenta las principales definiciones necesarias para comprender el trabajo de titulación. Se habla de los diferentes componentes que conforman el vehículo terrestre no tripulado (UGV), representado en un diagrama de bloques como se muestra en la Figura 1, se definen conceptos de UGV, sistema de radio control, sistema de telemetría, controlador de navegación, sensores y sistemas mecánicos.

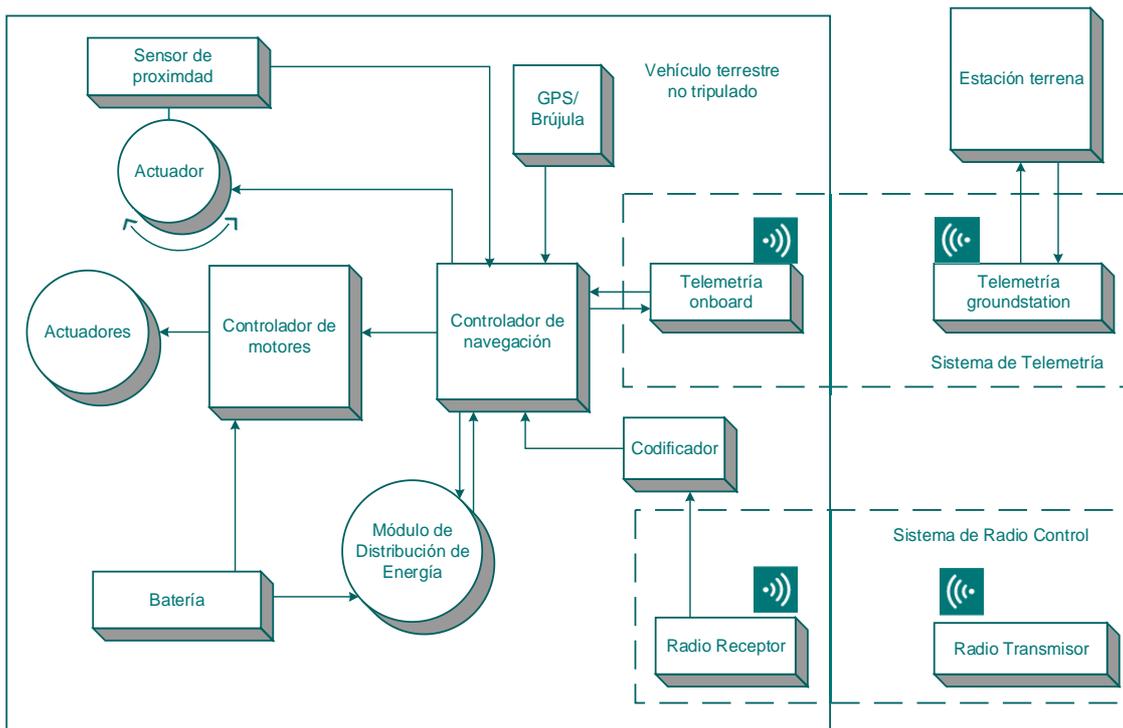


Figura 1. Componentes UGV.

2.1. Vehículo Terrestre no Tripulado

El vehículo terrestre no tripulado (*UGV, Unmanned Ground Vehicle*), es un conjunto de sistemas electrónicos y mecánicos que se desplaza sobre una superficie, puede llevar a bordo sensores para analizar el entorno y sistemas de telemetría para la comunicación con la estación terrena. El UGV es operado de forma autónoma o de forma manual por medio de un radio control (PX4 Dev Team, 2018).

Los UGV pueden ser utilizados en diferentes aplicaciones aprovechando los beneficios que pueden brindar en cuanto a eficiencia, tiempo de trabajo continuo, llegar a lugares de difícil acceso, trabajar en condiciones climáticas adversas, ser utilizados en situaciones de alto riesgo y en tareas con rutinas repetitivas (Alke, 2016).

2.2. Controlador de Navegación

Es una plataforma de hardware desarrollada para la navegación autónoma de vehículos no tripulados, generalmente está construido con los siguientes

elementos: 1) dos procesadores, uno principal y un procesador entrada/salida optimizado para la lectura/escritura sobre interfaces, liberando así de estas funciones al procesador principal, 2) sensor magnético o brújula, 3) sensor de gravedad o acelerómetro, 4) sensor inercial o giroscopio, 5) sensor de presión atmosférica o barómetro, 6) interfaces para los actuadores, 7) interfaces PWM para servos, 8) interfaces seriales para sistemas de telemetría, 9) módulo GPS, 10) interfaz para señales PPM receptor RC, 10) un bus I2C y 11) un bus CAN (ArduPilot Dev Team, 2018).

Con todo el hardware que integra esta plataforma se puede desarrollar de manera más sencilla el código de navegación autónoma sin preocuparse de integrar todo un conjunto de sensores, interfaces y microcontroladores.

2.2.1. Interfaces de comunicación

- UART

Universal Asynchronous receiver/transmitter es un circuito integrado encargado de implementar comunicación serial, de forma general actúa como intermediario entre interfaces paralelas e interfaces seriales. Su comunicación es asíncrona, es decir el envío y recepción de cada ráfaga no es controlada por un reloj y es necesario agregar bits de inicio y fin para recibir la información correctamente. UART utiliza dos conexiones para la comunicación Tx/Rx (Bloom, 2015), como se observa en la Figura 2.

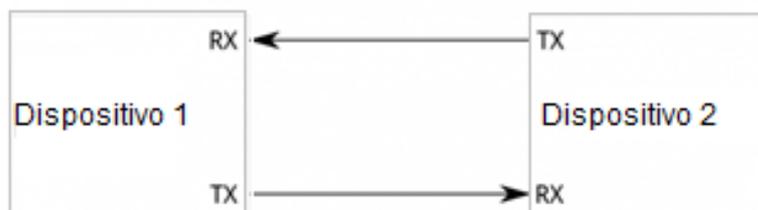


Figura 2. Comunicación Serial.

Tomado de Hord, 2016.

- I2C

Inter integrated circuit es un bus que fue desarrollado por Philips Chips en 1982. Este protocolo permite conectar en un solo bus varios dispositivos como esclavos y maestros, utilizando únicamente dos conexiones para la comunicación, como se muestra en la Figura 3. Adicional, utiliza una señal de reloj la cual es generada por el dispositivo maestro y una señal de datos (Hord, 2016).

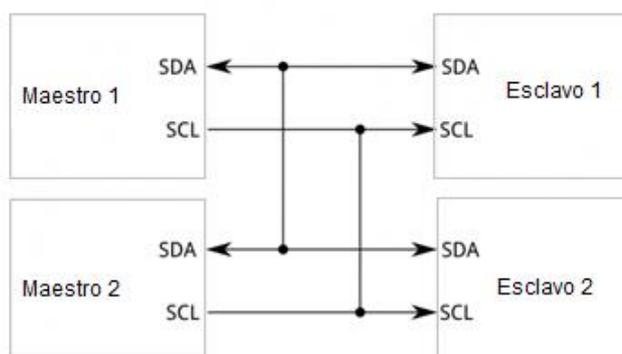


Figura 3. Comunicación I2C.

Tomado de Hord, 2016.

- CAN

Este protocolo fue desarrollado propiamente para aplicaciones en vehículos, sin embargo, puede satisfacer una variedad de aplicaciones gracias a su alta velocidad y bajo costo. El protocolo CAN permite la transmisión simultánea de todos los nodos (Huang, 2009)

2.3. Piloto automático

El software de piloto automático o pila de navegación es el encargado de controlar el vehículo no tripulado. Se pueden ejecutar en sistemas operativos como Linux, NuttX, MacOS (PX4 Dev Team, 2018). El sistema operativo depende del hardware de control de navegación sobre el cual se trabaje.

Existen dos principales herramientas de software de piloto automático que son: PX4 y ArduPilot, ambos desarrollados en un entorno Linux. Estas pilas de navegación pueden ser utilizadas para vehículos no tripulados como: rover,

botes, submarinos, aviones, o helicópteros de n hélices. Además, al ser software de código abierto es posible modificarlo para diferentes aplicaciones (ArduPilot, 2016).

2.4. Telemetría

La telemetría proporciona un enlace de comunicación remota entre la estación terrena y el vehículo no tripulado, para la recolección de datos y envío de instrucciones mediante un protocolo de comunicación MAVLink. El uso de telemetría permite enviar instrucciones, configurar parámetros, recibir datos y operar el vehículo desde la estación terrestre en tiempo real (PX4 Dev Team, 2018).

Existen diferentes tipos de radios de telemetría que pueden ser usados para el enlace de comunicación, estos incluyen: Bluetooth, ESP8266 wifi, SiK Radio, XBee (ArduPilot Dev Team, 2018). La comunicación de estas tecnologías se realiza mediante la emisión de ondas de radio.

2.5. Estación terrena

Es una aplicación que se ejecuta en una computadora, Tablet o celular, sobre diferentes sistemas operativos como Windows, Linux, Android o MacOS. Se comunica con el vehículo no tripulado a través de telemetría. La estación terrena se usa para configurar parámetros, establecer rutas, obtener información de sensores y controlar el vehículo (PX4 Dev Team, 2018).

Existen múltiples aplicaciones de estaciones terrenas, entre ellas: Mission Planner, QGroundControl, APM Planner, MAVProxy, AndroPilot, UgCS, Tower, MAVPilot. La decisión de seleccionar una aplicación de estación terrena depende de las actividades que se busquen realizar, la compatibilidad con la pila de piloto automático y que se adapte a las necesidades requeridas (ArduPilot Dev Team, 2018).

2.6. Codificador PPM-Sum

Es un dispositivo de hardware que integra un microcontrolador para la codificación de varias señales de servo PWM (pulse width modulated), en una sola señal PPM (pulse position modulation) como se ilustra en la Figura 4.

No es posible conectar directamente un radio receptor a un controlador de navegación ya que estos integran una interfaz de entrada PPM en lugar de 8 entradas PWM, por lo que es necesario utilizar un codificador PPM (ArduPilot Dev Team, 2018).

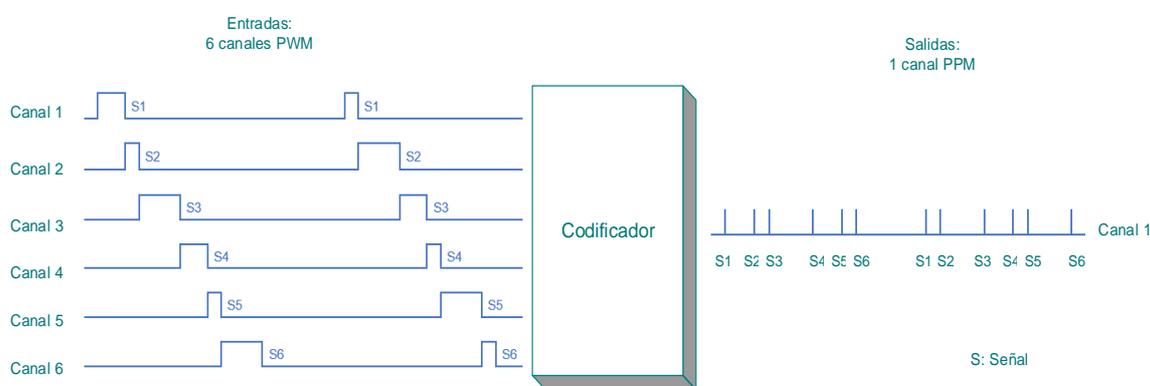


Figura 4. Codificación de señales PWM a PPM.

2.7. Sistema de Radio Control

Los vehículos no tripulados pueden operar de manera autónoma o ser controlados por un operador de forma manual. Para operar de forma manual es necesario un sistema de radio control. Estos sistemas están constituidos por un radio transmisor y un radio receptor, los cuales establecen una comunicación *simplex*, es decir de una sola vía. El radio receptor puede tener dos o más canales por los cuales recibe la señal del radio transmisor. Esta señal es transmitida a través de dos o más interfaces, dependiendo el número de canales (PX4 Dev Team, 2018).

2.8. GPS/Brújula

GPS, (por sus siglas en inglés, *Global Positioning System*) es un sistema de posicionamiento global que permite determinar la posición en tiempo real de un objeto con una precisión y exactitud de metros o centímetros dependiendo del tipo de GPS. El GPS proporciona datos como latitud, longitud, altura, entre otros. El sistema funciona mediante una red de satélites que se encuentran orbitando la Tierra. Se miden distancias mediante triangulación desde el satélite al receptor (Zurdo, 2005).

La brújula determina la orientación, tomando como punto de referencia el Norte magnético. Este dispositivo permite el correcto desplazamiento del vehículo, orientando su trayectoria.

2.9. Sensores de proximidad

Los sensores de proximidad son dispositivos que miden la proximidad de un objeto respecto al sensor.

Existen diferentes tipos de sensores de proximidad dependiendo de su principio de su funcionamiento:

- Sensores fotoeléctricos (emiten un haz de luz) y ultrasónicos (emiten una onda sonora) que miden la distancia del objeto al sensor, para ello emiten una onda que se refleja en el objeto y se determina la distancia.
- Sensores capacitivos, inductivos y magnéticos que detectan la presencia de objetos (Corona Ramírez, Abarca Jiménez y Mares Carreño, 2014).

Los sensores de proximidad pueden ser utilizados en múltiples aplicaciones como: detectar obstáculos, determinar posiciones, crear trayectorias, resolver laberintos.

2.10. Controlador o Driver de actuadores

Es un dispositivo de hardware electrónico que provee de voltaje y corriente a actuadores en base a señales. Existen diferentes tipos de controladores, en los

más sencillos la interfaz de entrada acepta señales PWM, mientras que en controladores más complejos se integran microcontroladores que les permiten procesar comandos a través de interfaces seriales (CAN, I2C, UART) (Pololu, 2018).

Para el movimiento del vehículo no tripulado, señales o comandos son enviados por el hardware de control de navegación hacia los actuadores, previo a esto es necesario un circuito de amplificación y es aquí donde el controlador de actuadores ahorra la tarea de diseñar y construir dicho hardware.

2.11. Batería

Es un dispositivo electroquímico, conformado por un conjunto de dos o más celdas capaces de almacenar energía en estado químico y convertirla en energía eléctrica. Las baterías tienen tres principales elementos: un ánodo (placa negativa), un cátodo (placa positiva) y un electrolito que permite el paso de iones (Arrow Electronics, 2018).

Existen dos grupos de baterías: 1) baterías primarias, que se desechan una vez que ya no pueden suministrar de suficiente energía a un dispositivo y 2) baterías secundarias o de almacenamiento, que pueden ser recargadas después de su descarga completa o parcial (Encyclopædia Britannica, 2018).

2.12. Módulo de distribución de energía

El módulo de distribución de energía (*Power Distribution Board*, PDB) es un dispositivo electrónico encargado de suministrar de energía a los diferentes componentes electrónicos que conforman el UGV, además que cumple la función de regular el voltaje.

El módulo de distribución regula el voltaje de salida, logrando que se encuentre dentro de los límites aceptables tolerados por el dispositivo al que se suministra dicho voltaje, además ayuda a que no existan variaciones de voltajes que podrían ser perjudiciales para los dispositivos (Encyclopædia Britannica, 2018).

2.13. Actuadores

Los actuadores son dispositivos que generan movimientos o cambios de posición según las órdenes dadas por el controlador. Un actuador tiene la capacidad de ejercer una fuerza a partir de la transformación de energía.

Los actuadores se clasifican en dos grupos: 1) por el tipo de energía que emplean: neumática, hidráulica y eléctrica, y 2) por el tipo de movimiento que generan: lineal y rotatoria (Corona Ramírez, Abarca Jiménez y Mares Carreño, 2014).

2.14. Chasis

Estructura o armazón cuyo propósito es dar soporte a los diferentes componentes que forman parte del vehículo como: sensores, sistemas de suspensión, actuadores, sistemas electrónicos, entre otros. El chasis generalmente es metálico o de plástico.

3. Diseño

En este capítulo se realiza un análisis de los diferentes componentes que integra el UGV tanto en dispositivos de hardware como en sistemas de software, se exponen los motivos por los cuales se decide trabajar con uno u otro sistema. Además, se realiza una comparativa de ciertos dispositivos de hardware y sistemas de software, entre los cuales se selecciona el que mejor cumpla los requerimientos o el que mejor se adapta a la implementación para cumplir los objetivos planteados.

3.1. Diseño de Hardware

En esta sección se realiza el análisis para la selección de los dispositivos de hardware. Además, se realiza el diseño de la arquitectura de hardware, es decir las conexiones entre los dispositivos que componen el UGV en base a las diferentes interfaces y protocolos de comunicación.

3.1.1. Chasis

El diseño del vehículo parte con un chasis *Dagu Wild Thumper 6WD*, sobre el cual se integran los diferentes componentes. El chasis cuenta con seis ruedas, cajas de engranajes con una relación de 75:1, un motor de corriente continua con un nivel de voltaje de 7.2 voltios y con una suspensión independiente para cada una de las ruedas (Pololu, 2017), como se ilustra en la Figura 5.

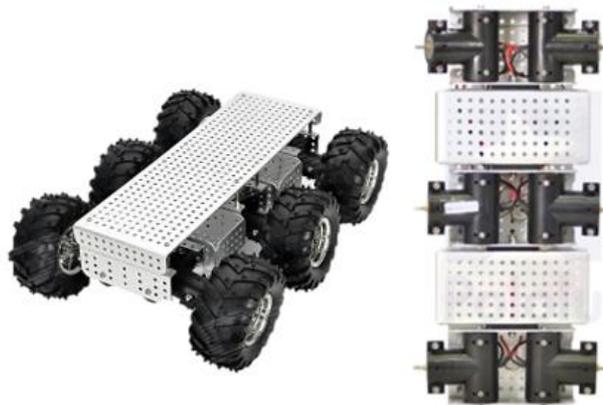


Figura 5. Chasis Daggu Wild Thumper 6WD.

Tomado de Pololu, 2017

La estructura es de aluminio y tiene orificios para el anclaje de los diferentes elementos: sensores, actuadores, sistemas electrónicos, entre otros, que integran el vehículo. El sistema de tracción de 6 ruedas brinda al vehículo la capacidad de desplazarse en terrenos irregulares y pendientes (Pololu, 2017). En la Tabla 1, se describen las especificaciones técnicas del chasis *Dagu Wild Thumper 6WD*.

Tabla 1
Especificaciones chasis *Dagu Wild Thumper 6WD*

Especificaciones	Descripción
Dimensiones	420x300x130 mm
Peso	2.7 kg

Número Ruedas	6
Diámetro Ruedas	120 mm
Carga Útil Recomendada	5 kg
Dimensiones	420x300x130 mm

Tomado de Pololu, 2017.

Para este trabajo de titulación se usa el chasis *Dagu Wild Thumper 6WD (6-wheel drive)*, debido a que es un chasis ideal para exteriores al contar con tracción en las seis ruedas, suspensión independiente y un gran torque gracias a su caja de relación. Además, gracias al sistema de tracción independiente es posible tener un radio de giro igual cero permitiéndole una gran flexibilidad en su desplazamiento para la evasión de obstáculos. Las características técnicas son adecuadas para la implementación planteada.

3.1.2. Controlador de navegación

A continuación, se realizó un análisis entre dos controladores de navegación de la familia Pixhawk disponibles, tomando en cuenta su estabilidad y confiabilidad en la lectura de los sensores. Además, se analiza las interfaces de comunicación necesarias para el desarrollo del tema de titulación.

3.1.2.1. Pixhawk Mini

El controlador de navegación Pixhawk mini fue diseñado por 3DR y es la evolución de Pixhawk 1. El Pixhawk Mini en referencia a su nombre es 1/3 del tamaño del Pixhawk original. Este controlador es recomendado para todo tipo de aplicaciones, sin embargo, no está optimizado para aplicaciones relacionados con video, procesamiento de imágenes o donde se requiera grandes capacidades de cómputo (PX4 Dev Team, 2017).

El controlador internamente cuenta con dos procesadores: un procesador principal (STM32F427 Rev 3) y un procesador I/O (STM32F103). Integra los

siguientes sensores: 1) acelerómetro, giroscopio, y magnetómetro en un solo integrado (MPU9250), 2) acelerómetro y giroscopio en un segundo integrado (ICM20608) y 3) barómetro en un integrado (MS5611) (PX4 Dev Team, 2017), como se ilustra en la Figura 6.

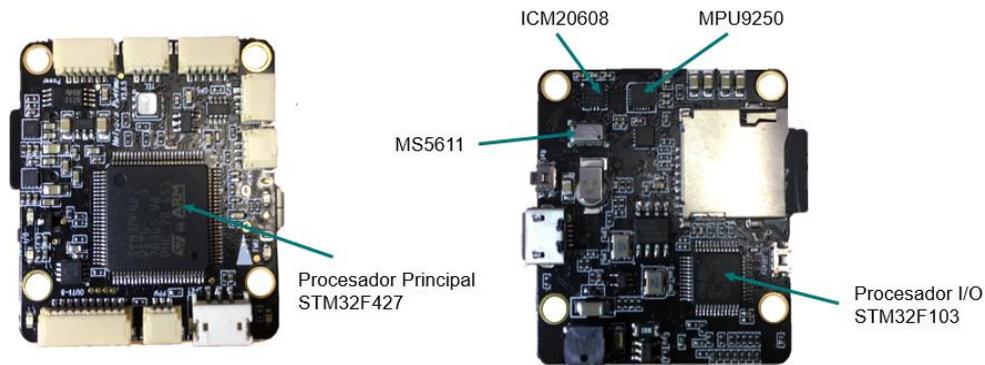


Figura 6. Placa Pixhawk.

Para la comunicación el controlador de navegación integra las siguientes interfaces seriales:

- UART Serial Port (módulo GPS) / I2C (sensores digitales)

Tabla 2
Asignación de pines GPS e I2C

Pin	Señal	Voltaje
1 rojo	SCL (I2C)	+3.3 V
2 negro	SDA (I2C)	+3.3 V
3 negro	VCC	+5 V
4 negro	TX3 (GPS)	+3.3 V
5 negro	RX3 (GPS)	+3.3 V
6 negro	GND	GND

Tomado de PX4 Dev Team, 2017.

- UART Serial Port (Telemetría)

Tabla 3
Asignación de pines Telemetría

Pin	Señal	Voltaje
1 negro	VCC	+5 V
2 negro	TX1 (GPS)	+3.3 V
3 negro	RX1 (GPS)	+3.3 V
4 negro	GND	GND

Tomado de PX4 Dev Team, 2017

- Micro USB Port

Para la operación por radio control el controlador integra dos interfaces de comunicación.

- Spektrum DSM/DSM2/DSM-X

Tabla 4
Asignación de pines Spektrum DSM

Pin	Señal	Voltaje
1 anaranjado	VCC	+3.3 V
2 negro	GND	GND
3 plomo	Datos	+3.3 V

Tomado de PX4 Dev Team, 2017

- Futaba S BUS / PPM Sum Signal

Tabla 5
Asignación de pines PPM Sum

Pin	Señal	Voltaje
1 rojo	VCC	+5 V
2 amarillo	Señal	+3.3 V
3 negro	GND	GND

Tomado de PX4 Dev Team, 2017

Para el control de los actuadores se puede utilizar dos interfaces:

- 8 Channels PWM

Tabla 6
Asignación de pines Canales PWM

Pin	Señal	Voltaje
1 negro	GND	GND
2 rojo	VCC	+5 V
Pin 1	Throttle	--
Pin 2	--	--
Pin 3	Steering	--
Pin 4	--	--
Pin 5	--	--
Pin 6	--	--
Pin 7	--	--

Tomado de PX4 Dev Team, 2017

- CAN (for digital motor control with compatible controllers)

Tabla 7

Asignación de pines puerto CAN

Pin	Señal	Voltaje
1 rojo	VCC	+5 V
2 negro	CAN-H	+3.3 V
3 negro	CAN-L	+3.3 V
4 negro	GND	GND

Tomado de PX4 Dev Team, 2017

El controlador es energizado mediante el puerto Power input desde la PDB.

Tabla 8

Asignación de pines interfaz de alimentación

Pin	Señal	Voltaje
1 rojo	VCC	+5 V
2 negro	VCC	+5 V
3 negro	Corriente	+3.3 V
4 negro	Voltaje	+3.3 V
5 negro	GND	GND
6 negro	GND	GND

Tomado de PX4 Dev Team, 2017

El controlador utiliza un interruptor de seguridad el cual impide armar el vehículo hasta que este no sea accionado, este interruptor se conecta con el controlador mediante el Safety switch port.

Tabla 9
Asignación de pines switch de seguridad

Pin	Señal	Voltaje
1 rojo	VCC	+3.3 V
2 negro	IO_LED_SAFETY	GND
3 negro	SAFETY	GND

Tomado de PX4 Dev Team, 2017

3.1.2.1.1. GPS/Brújula

El kit del controlador de navegación Pixhawk Mini incluye un módulo GPS externo uBlox Neo M8N, con una brújula internamente incorporada HMC5983, que se observa en la Figura 7.



Figura 7. Módulo GPS.

Tomado de PX4 Dev Team, 2017

En la Tabla 10, que se muestra a continuación se describen las especificaciones técnicas del módulo GPS.

Tabla 10
Especificaciones módulo GPS

Recepción simultánea de GNSS	GPS, Galileo, GLONASS, BeiDou
-------------------------------------	-------------------------------

Sensibilidad de navegación	-167 dBm
Seguridad y protección	Detección avanzada de atascos y spoofing

Tomado de PX4 Dev Team, 2017

3.1.2.1.2. **Módulo de distribución de energía**

El kit del controlador de navegación Pixhawk Mini incluye una placa de distribución de energía PM06 V1.0, que se muestra en la Figura 8, para suministrar de corriente y voltaje al Pixhawk, actuadores y otros componentes del UGV desde la batería (PX4 Dev Team, 2017).



Figura 8. PDB PM06.

Tomado de PX4 Dev Team, 2017

3.1.2.1.3. **Limitaciones**

El Pixhawk mini presenta un defecto de hardware, específicamente en la IMU interna principal IMU9250 que integra los sensores: acelerómetro, giroscopio, y magnetómetro, este problema afecta al correcto funcionamiento del vehículo (PX4 Dev Team, 2017).

Debido a que el controlador Pixhawk mini se encontraba disponible se pudo realizar pruebas donde se pudo comprobar que este se calibra correctamente, sin embargo, debido a que la IMU principal está afectada, tan solo se cuenta con un segundo integrado ICM20608 y al no tener redundancia de los sensores se pueden obtener datos incorrectos provocando que en ocasiones el vehículo no funcione de manera adecuada.

El Pixhawk Mini cuenta únicamente con dos interfaces seriales, una interfaz para la conexión del GPS y otra interfaz para telemetría y se requiere de una interfaz más para la conexión del sensor de distancia.

3.1.2.2. Cube

El controlador de vuelo *Cube* anteriormente conocido como Pixhawk 2.1 fue desarrollado por Hex Technology. El controlador se integra de dos estructuras: 1) un cubo metálico en el que se encuentra la tarjeta del controlador de navegación y 2) una base cuadrada de 4 cm y altura de 2 cm, donde se encuentran las diferentes interfaces, como se observa en la Figura 9. Además, cuenta con un solo puerto DF17 el cual permite una conexión sencilla a las distintas placas portadoras comerciales, esto simplifica su estructura reduciendo el cableado. Este controlador está destinado principalmente para sistemas comerciales debido a su pequeño tamaño y flexibilidad de conexión. (Hex Technology Limited, 2016).

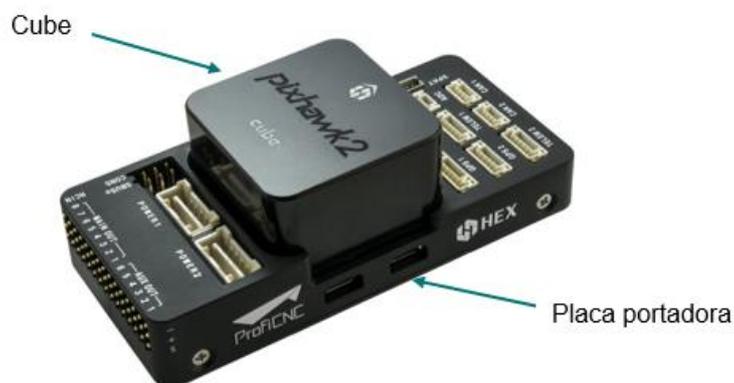


Figura 9. Pixhawk 2.1 Standard Carrier Board.

Adaptado de ProfiCNC, 2017

Este controlador de vuelo está basado en el proyecto Pixhawk FMUv3, cuenta con tres sensores IMU para una mejor estabilidad y confiabilidad, como se puede observar en la Figura 10, dos de estos sensores se encuentran aislados mecánicamente para una lectura de mayor fiabilidad ante vibraciones de alta frecuencia, además cuenta con resistencias térmicas para el control de temperatura permitiendo un óptimo funcionamiento de los sensores IMU (Hex Technology Limited, 2016).

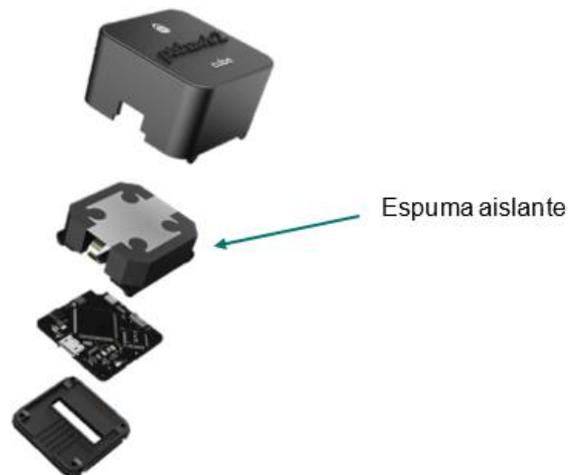


Figura 10. Componentes cube.

Adaptado de ProfiCNC, 2017

En la Figura 11, corresponde a la tarjeta portadora estándar del cube, se puede observar las diferentes interfaces de comunicación para los distintos componentes del UGV.

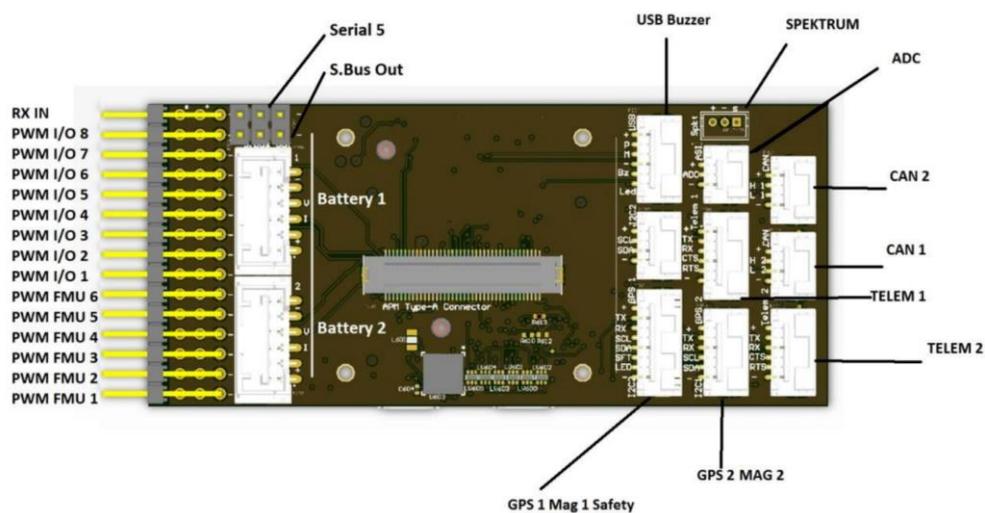


Figura 11. PCB Carrier Board.

Tomado de Hex Technology Limited, 2016

A continuación, se describe la asignación de pines de cada una de las interfaces que integran el Pixhawk cube.

- Battery 1

Tabla 11
Asignación de pines batería

Pin	Nombre	Voltaje
1 rojo	VDD Brick	+5 V
2 rojo	VDD Brick	+5 V
3 negro	BATT_VOLTAGE_SENS_PROT	+5 V
4 negro	BATT_CURRENT_SENS_PROT	+5 V
5 negro	GND	-
6 negro	GND	-

Tomado de Hex Technology Limited, 2016

- Battery 2

Tabla 12
Asignación de pines batería

Pin	Nombre	Voltaje
1 rojo	VDD Brick	+5 V
2 rojo	VDD Brick	+5 V
3 negro	AUX_BATT_VOLTAGE_SENS_PROT	+5 V
4 negro	AUX_BATT_CURRENT_SENS_PROT	+5 V
5 negro	GND	-
6 negro	GND	-

Tomado de Hex Technology Limited, 2016

- I2C

Tabla 13
Asignación de pines puerto I2C

Pin	Nombre	Voltaje
1 rojo	VCC	+5 V
2 negro	SCL (I2C)	+5 V
3 negro	SDA (I2C)	+5 V
4 negro	GND	-

Tomado de Hex Technology Limited, 2016

- CAN

Tabla 14
Asignación de pines puerto CAN

Pin	Nombre	Voltaje
1 rojo	VCC	+5 V
2 negro	CAN-H	+3.3 V
3 negro	CAN-L	+3.3 V
4 negro	GND	-

Tomado de Hex Technology Limited, 2016

- UART Generic

Tabla 15
Asignación de pines puerto UART

Pin	Nombre	Voltaje
1 rojo	VCC	+5 V

2 negro	MCU_TX	+3.3-5 V
3 negro	MCU_RX	+3.3-5 V
4 negro	MCU_CTS (TX)	+3.3-5 V
5 negro	MCU_RTS (RX)	+3.3-5 V
6 negro	GND	-

Tomado de Hex Technology Limited, 2016

- UART GPS

Tabla 16
Asignación de pines GPS

Pin	Nombre	Voltaje
1 rojo	VCC	+5 V
2 negro	GPS_TX	+3.3-5 V
3 negro	GPS_RX	+3.3-5 V
4 negro	SCL	+3.3-5 V
5 negro	SDA	+3.3-5 V
6 negro	BUTTON	-
7 negro	BUTTON_LED	-
8 negro	GND	-

Tomado de Hex Technology Limited, 2016

- UART 4

Tabla 17
Asignación de pines puerto UART

Pin	Nombre	Voltaje
1 rojo	VCC	+5 V
2 negro	MCU_TX	+3.3-5 V
3 negro	MCU_RX	+3.3-5 V
4 negro	SCL	+3.3-5 V
5 negro	SDA	+3.3-5 V
6 negro	GND	GND

Tomado de Hex Technology Limited, 2016

- UART 5

Tabla 18
Asignación de pines puerto UART

Pin	Nombre	Voltaje
1 rojo	S.Bus Out	+5 V
2 negro	MCU_TX	+3.3-5 V
3 negro	VDD_Servo	
4 negro	MCU_RX	+3.3-5 V
5 negro	GND	-
6 negro	GND	-

Tomado de Hex Technology Limited, 2016

- Análogo

Tabla 19
Asignación de pines análogos

Pin	Nombre	Voltaje
1 rojo	VDD	+5 V
2 negro	Pressure sense in	+0-5 V
3 negro	GND	-

Tomado de Hex Technology Limited, 2016

- Spektrum

Tabla 20
Asignación de pines Spektrum

Pin	Nombre	Voltaje
1	VDD	+3.3 V
2 negro	Pressure sense in	+0-5 V
3 negro	GND	-

Tomado de Hex Technology Limited, 2016

- HMI

Tabla 21
Asignación de pines buzzer y USB

Pin	Nombre	Voltaje
1 rojo	VCC	+5 V
2 negro	D_PLUS	+3.3 V

3 negro	D_MINUS	+3.3 V
4 negro	GND	-
5 negro	BE_LED	-
6 negro	BUZZER	VBAT

Tomado de Hex Technology Limited, 2016

- SERVO HEADER

Tabla 22
Asignación de pines servos

Posición	Nombre	Voltaje
15	RC / SBUS IN	+3.3 V
14	MAIN_OUT_8	+3.3 V
13	MAIN_OUT_7	+3.3 V
12	MAIN_OUT_6	+3.3 V
11	MAIN_OUT_5	+3.3 V
10	MAIN_OUT_4	+3.3 V
9	MAIN_OUT_3	+3.3 V
8	MAIN_OUT_2	+3.3 V
7	MAIN_OUT_1	+3.3 V
6	AUX_OUT_1	+3.3 V
5	AUX_OUT_5	+3.3 V
4	AUX_OUT_4	+3.3 V
3	AUX_OUT_3	+3.3 V

2	AUX_OUT_2	+3.3 V
1	AUX_OUT_1	+3.3 V

Tomado de Hex Technology Limited, 2016

3.1.2.2.1. **HERE 2**

El here 2 es un módulo GPS, el cual cuenta internamente con un sensor IMU ICM20948, un barómetro MS5611, y además integra un procesador STM32F302. Gracias al procesador integrado el módulo es capaz de ejecutar ChibiOS basado en Real Time Operating System, estas cualidades le permiten ser configurado según las necesidades del usuario. Además, cuenta con interfaces UART, I2C, y CAN (ProfiCNC, 2017).

En la Figura 12, se puede observar que el módulo dispone de un botón el cual es utilizado como *safety switch*, con el objetivo de reducir el cableado y para mayor confiabilidad.



Figura 12. Here 2.

Adaptado de ProfiCNC, 2017

3.1.2.2.2. **Módulo de energía**

El kit del controlador de navegación Pixhawk Cube incluye un *power brick mini*, que se observa en la Figura 13, el módulo proporciona un suministro confiable de voltaje/corriente a la placa y permite suministrar alimentación por separado a

los ESC. El módulo admite hasta baterías de 8 celdas y una corriente máxima de 30 Amperios (PX4 Dev Team, 2019).



Figura 13. Power Brick Mini.

Tomado de ProfiCNC, 2017

3.1.2.3. Selección del Controlador de Navegación

Luego de las limitaciones mencionadas en el Pixhawk Mini se opta por adquirir el Pixhawk Cube, debido a que el controlador es el hardware de piloto automático más recomendado en el mercado, no registra problemas encontrados en el funcionamiento, cuenta con redundancia en sus sensores, integra interfaces seriales necesarias para las conexiones, está optimizado para ArduPilot y está orientado para el desarrollo de aplicaciones (Hex Technology Limited, 2016).

3.1.3. Sistema de Radio Control

El sistema de radio control está conformado por un radio transmisor y un radio receptor, los cuales establecen una comunicación.

- Radio Receptor

El radio receptor usado es el AR6210 DSMX de 6 canales, como se muestra en la Figura 14. El receptor cuenta con tecnología Spektrum DSM (*Digital Spectrum Modulation*) y es compatible con todos los radios transmisor Spektrum y JR que admiten la tecnología DSM2 (*second-generation Digital Spread Spectrum Modulation*) y DSMX (*Digital Spectrum Modulation 'X'-tra*) (Spektrum & Horizon Hobby, 2010).



Figura 14. Canales Radio Receptor.

Tomado de Spektrum & Horizon Hobby, 2010

En la Tabla 23, se describen las especificaciones técnicas del radio receptor AR6210 DSMX.

Tabla 23
Especificaciones AR6210 DSMX

Tipo	Receptor de largo alcance
Canales	6
Modulación	DSM2 - DSMX
Banda de Frecuencia	2.4 - 2.4835 GHz
Rango de voltaje	3.5–9.6V

Tomado de Spektrum & Horizon Hobby, 2010

La modulación DSM se basa en la modulación DSSS (*Direct Sequencing Spread Spectrum*), desarrollada y optimizada para sistemas de radio control. DSM proporciona un enlace seguro e inmune a interferencias de radio.

DSM escanea la banda de frecuencia de 2.4 - 2.4835 GHz en busca de un canal libre. Cuando encuentra un canal libre reserva este canal para la comunicación, de esta manera si otro control de radio entra en operación en la misma frecuencia, estará en un diferente canal. DSM tiene disponible 79 canales, con un ancho de banda por canal de 1MHz (Horizon Hobby, 2004).

DSM2 es la segunda generación de la tecnología DSM. DSM2 utiliza la tecnología doble enlace (*DuaLink*), reserva dos canales libres para la comunicación en la banda de frecuencia de 2.4 - 2.4835 GHz (Spektrum & Horizon Hobby, 2005).

DSMX funciona de similar forma que DSM2, adicional brinda una protección contra interferencias en el canal y tiempos de reconexión más rápidos (Spektrum & Horizon Hobby, 2005).

- Radio Transmisor

El Transmisor DX5e es un transmisor de largo alcance de 5 canales que opera en la frecuencia de 2.4 GHz, con modulación DSM2 y DSMX.

Este trasmisor cuenta con dos palancas (*sticks*) de dos ejes. La palanca derecha maneja la inclinación en los ejes (*roll, pitch*), mientras que la palanca izquierda maneja la aceleración (*throttle*) y la rotación (*yaw*). Además, cuenta con interruptores auxiliares para ser configurados, por ejemplo, cambiar modos de navegación (manual o autónomo) (Spektrum & Horizon Hobby, 2011). En la Figura 15, se puede observar un Transmisor DX5e.



Figura 15. Transmisor DX5e.

Tomado de Spektrum & Horizon Hobby, 2011

El transmisor Dx5e según la distribución de las palancas está diseñado para el control de aviones y cuadricópteros. Sin embargo, para la aplicación del tema de

titulación este cumple con todas las necesidades, ya que solo es necesario dos canales. En la Figura 16, se puede observar los movimientos de rotación sobre los tres ejes.

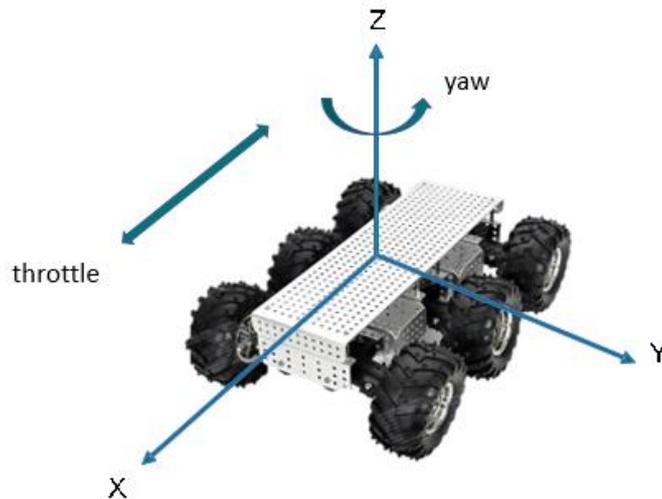


Figura 16. Ejes de rotación.

Para el control manual del UGV se necesita únicamente de aceleración y dirección. Sin embargo, también es necesario de interruptores auxiliares para funciones, como: la selección del modo de navegación, guardar la posición en tiempo real, y otras funciones adicionales que se pueden configurar. El receptor AR6210 DSMX y transmisor DX5e seleccionados cuentan con los canales e interruptores necesarios para la implementación del vehículo terrestre autónomo.

- Enlace entre receptor y Transmisor

Antes de usar el sistema de radio control, es necesario enlazar el receptor y el transmisor. El enlace consiste en que el receptor enviará su código de identificación global GUID al transmisor, código necesario para la modulación DSSS.

Los pasos para vincular el transmisor y receptor son: 1) apagar ambos dispositivos, 2) en el receptor conectar a tierra el pin *bind*, 3) encender el receptor, en este momento un led naranja parpadeara indicando que se encuentra en modo *bind*, 4) en el transmisor activar el interruptor *bind*, 4) encender el transmisor, en este momento los cuatro leds del transmisor

parpadean simultáneamente indicando que se encuentra en modo *bind*, para saber que se ha establecido el enlace correctamente, en el receptor el led deja de parpadear y se torna sólido, 5) ahora es necesario volver a apagar ambos dispositivos, desconectar el pin *bind* del receptor y soltar el interruptor *bind*, 6) volver a encender nuevamente ambos dispositivos (Spektrum & Horizon Hobby, 2010).

3.1.4. Telemetría

Para la comunicación entre la estación terrena y el UGV existen diferentes tecnologías como Bluetooth, Wi-Fi, Zigbee, y SikRadio. En el mercado se puede encontrar una gran variedad de dispositivos para cada una de estas tecnologías.

A continuación, en la Tabla 24 se realiza una comparación de las diferentes tecnologías, analizando módulos de un similar costo en el mercado.

Tabla 24
Tecnologías de inalámbricas

Tecnología	Módulo	Estándar	Frecuencia	Alcance línea de vista (m)	Ancho de Banda	Corriente (mA)
Bluetooth	HC-05	IEEE 802.15.1	2.4 GHz	20 (clase 2)	2 Mbps (v2)	25
Wi-Fi	ESP8266 ESP-01	IEEE 802.11 b/g/n	2.4 GHz	100 - 450	54 Mbps	80
Zigbee	Xbee S2C	IEEE 802.15.4	2.4 GHz	1200	1Mbps	33

SiK Telemetry Radio	3DR Radio	--	915 MHz	300	250 kbps	35
----------------------------	-----------	----	---------	-----	----------	----

Tomado de Digi International, 2018; Guangzhou HC Information Technology, 2011; Shenzhen Anxinke Technology, 2015

Luego del análisis realizado se concluye usar los módulos Xbee S2C, debido a que sus especificaciones técnicas cumplen con lo requerido para la comunicación del vehículo. Una de sus principales ventajas es que tienen un mayor alcance en línea de vista directa, mayor ancho de banda y un bajo consumo de corriente.

3.1.5. Batería

La elección de la batería a usar para el UGV depende del consumo total de corriente de todos los componentes que integran el mismo, y además del tiempo de navegación requerido. A continuación, en la Tabla 25 se realiza un análisis del consumo de corriente.

Tabla 25
Consumo de corriente de componentes del UGV

Componentes	Número	Voltaje	Corriente	Uso	Total
Motores	6	7.2 V	6.6 A	30%	11.88 A
Xbee S2C	1	5 V	33 mA		33 mA
GPS	1	3.6 V	67 mA		67 mA
Servo Motor	1	5 V	600 mA		600 mA
Arduino Nano (ATMEGA328P)	1	5 V	280 mA (7 pines activos)		280 mA

PPM Encoder (ATMEGA328P)	1	5 V	360 mA (9 pines activos)	360 mA
TFmini LiDAR	1	5 V	140 mA	140 mA
Pixhawk	1	5 V	280 mA	280 mA
	Máximo	12 V		Total 13,64 A

A continuación, se realiza el cálculo del tiempo de navegación del vehículo tomando en consideración el consumo total de corriente de todos los componentes que integran el UGV.

$$\text{Consumo Total por hora} = 13,64 \frac{A}{h}$$

$$\text{Voltaje Necesario} = 12 V$$

$$\text{Tiempo de Navegacion} = 15 \text{ min}$$

$$\text{Tiempo de Navegacion} = 0,25 h$$

$$\text{Bateria} = 13,64 \frac{A}{h} * 0,25 h$$

$$\text{Bateria} = 3,41A$$

Luego del análisis realizado se concluye que para un tiempo de navegación de 15 minutos y un consumo de corriente de 13.64 A/hora es necesario una batería de 3.41 A., con una diferencia de potencial de 12 V.

Existe una gran variedad de baterías, las cuales se diferencian en su funcionamiento, composición química, tamaños, aplicaciones, duración entre otros.

Como se menciona en el Capítulo 2, Sección 2.11 existen dos grupos de baterías: primarias y secundarias o de almacenamiento. Para este trabajo de titulación se considera el uso de baterías secundarias debido a que al ser recargables son una mejor alternativa, tienen una alta duración, se necesita una inversión inicial, pero a largo plazo representan un importante ahorro de dinero.

En la Tabla 26, que se muestra a continuación se realiza una comparación de las baterías recargables disponibles.

Tabla 26
Tipos de baterías recargables

Tipo	Composición química	Aplicaciones comunes	Características
Plomo-Ácido	Ánodo: plomo Cátodo: dióxido de plomo con electrolito de ácido sulfúrico	Utilizadas en automóviles, sillas de ruedas, vehículos eléctricos para niños y fuentes de alimentación de emergencia.	Amplia gama de tamaños, batería más barata y más pesada, larga vida, sin efecto memoria, amplio rango de tasas de descarga.
Alcalino			
Níquel - Cadmio	Ánodo: cadmio Cátodo: dióxido de níquel con electrolito de hidróxido de potasio	Utilizado en herramientas eléctricas, teléfonos inalámbricos y equipos biomédicos.	Excelente desempeño, voltaje casi constante, mejor ciclo de vida recargable, efecto de memoria en algunos, cadmio altamente tóxico y cancerígeno si no se recicla correctamente.

Níquel - Hidruro de metal	Ánodo:		
	aleación de níquel.	Utilizado en alarmas de humo, herramientas eléctricas y celulares.	Alta densidad de energía, descarga casi constante de 1.2 voltios, sin efecto memoria.
	Cátodo: dióxido de níquel con electrolito de hidróxido de potasio		

Litio

Ion Litio	Ánodo:	Utilizado en	Mayor densidad de energía y vida más corta que el níquel-cadmio; costoso; sin efecto memoria.
	carbono	celulares y computadoras portátiles.	
	Cátodo: dióxido de litio cobalto con electrolito orgánico		

Polímero Litio	Ánodo: litio	Sistemas de radio control	Baterías ligeras; tamaño reducido; alta densidad de carga; alta tasa de carga/descarga.
	Cátodo: electrolito sólido	aviones y cuadricópteros.	

Tomado de Encyclopædia Britannica, 2019a

Luego del análisis realizado se decide usar la batería de litio polímero Li-Po, que se muestra en la Figura 17, ya que posee una alta capacidad de carga, alta tasa de carga/descarga, todo esto con un tamaño y peso reducido.



Figura 17. Batería Li-Po.

Tomado de ArdoBot Robótica, 2018

A continuación, en la Tabla 27 se realiza una descripción de los principales factores en la selección de una batería Li-Po.

Tabla 27
Factores Batería Li-Po

Número de celdas “S”	El numero “S” indica el número de celdas que la batería tiene conectada en serie. Las batería Li-Po tiene un mayor voltaje en cada celda frente a otras, alcanzando 4.2 V por celda en su máxima capacidad.
Capacidad “mAh”	Este valor nos indica la capacidad de la batería, además que esta puede tener una tasa de descarga de “mA” en una hora.
Tasa de descarga “C”	Tasa de descarga = Capacidad mAh * C. Esto quiere decir: 1 C inidca que la batería tiene la misma tasa de descarga inidcada en la capacidad. 10 C, idica que la batería tiene la capacidad de descargar diez veces su capacidad en una hora.

En la Tabla 28, se describen las especificaciones técnicas de la batería seleccionada MultiStar.

Tabla 28
Especificaciones batería Li-Po MultiStar

Número de celdas	3s
Voltaje	12.6 V
Amperaje	4000 mA
Tasa de carga/descarga	10c

Tomado de Ardoobot Robótica, 2018

3.1.6. Codificador

Para la codificación de señales de servo PWM a señales PPM se analizan dos posibles soluciones:

- Implementar un codificar PPM utilizando la placa de desarrollo Arduino Nano, que se muestra en la Figura 18, que integra un procesador ATMEGA 328p, sobre la cual se carga un firmware desarrollado en lenguaje de programación C.

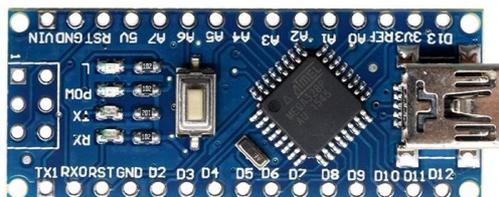


Figura 18. Arduino Nano.

Tomado de Arduino, 2018

- Adquirir un módulo comercial. El módulo PPM Encoder, que se muestra en la Figura 19, es un codificador PPM disponible en el mercado que integra un procesador ATMEGA 328p preconfigurado con un firmware para 8 canales, además cuenta con una interfaz que permite la conexión directa de los canales PWM con el receptor y otra interfaz para la conexión directa del canal PPM con el controlador de navegación Pixhawk (3D Robotics, 2014).

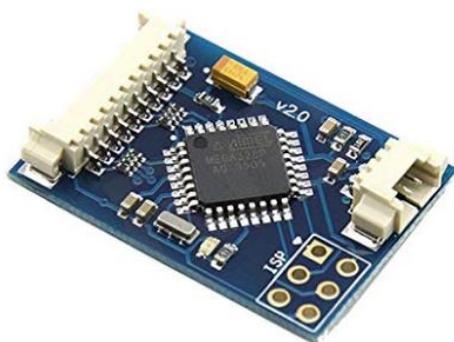


Figura 19. PPM Encoder.

Tomado de 3D Robotics, 2014

Para este trabajo de titulación se opta por usar el módulo comercial debido a que es una solución más compacta que integra interfaces que permiten una conexión directa de los canales y tiene un firmware preconfigurado para la codificación de señales.

3.1.7. Controlador o Driver de actuadores

Para proveer de voltaje y corriente a los actuadores es necesario el uso de un controlador o driver. En la Tabla 29, se realiza una comparación de los diferentes controladores disponibles en el mercado.

Tabla 29
Controladores de actuadores

Módulo	Canales de Motor	Interfaz de Control	Rango de voltaje	
			Entrada (V)	Salida por canal (A)
Pololu Simple High-Power Motor Controller 18v15	1	USB, TTL serial, servo pulses; analog voltage	5.5 - 30	15

Pololu Qik 2s12v10 Dual Serial Motor Controller	2	TTL serial, RS-232 serial	6 -16	13
Pololu TReX Dual Motor Controller DMC01	3	TTL serial, RS-232 serial, RC servo pulses, analog voltage	6 -16	13
Pololu Dual VNH5019 Motor Driver Shield for Arduino	2	Arduino Shield	5.5 - 24	12

Tomado de Pololu, 2017

Luego del análisis realizado se concluye que el controlador Pololu TReX Dual Motor Controller DMC01, es la solución más adecuada debido a que integra una interfaz de pulsos de servo necesarios para la conexión con el controlador de navegación Pixhawk. Además, cuenta con 3 canales de motor lo que permite accionar los actuadores con un solo controlador.

El controlador TReX Dual DMC01 permite controlar los actuadores mediante 4 interfaces. Las interfaces del Pixhawk para el control de motores son CAN y pulsos de servo PWM, por lo que en el controlador de motores se utiliza la interfaz para pulsos PWM. Para esto es necesario situar el jumper 3 en el pin RC.

Como cualquier receptor RC la interfaz de pulsos PWM del Pixhawk no suministra corriente, obligando así a crear un circuito de eliminación de batería conocido como BEC con el objetivo de no sobrecargar la interfaz. Para solucionarlo se debe situar el *jumper 2* en el Pololu TRex DMC, este provee de corriente a dicha interfaz.

El Pololu TRex DMC tiene dos modos para accionar los diferentes tipos de vehículos. En el caso del Chasis *Dagu Wild Thumper 6WD* que tiene un sistema de tracción diferencial es necesario remover el *jumper 1*. Mientras que para un vehículo con tracción y dirección el *jumper 1* debe ser situado (Pololu Corporation, 2015).

En la Figura 20, se puede observar la distribución de los jumpers en el controlador de actuadores Pololu TRex DMC.

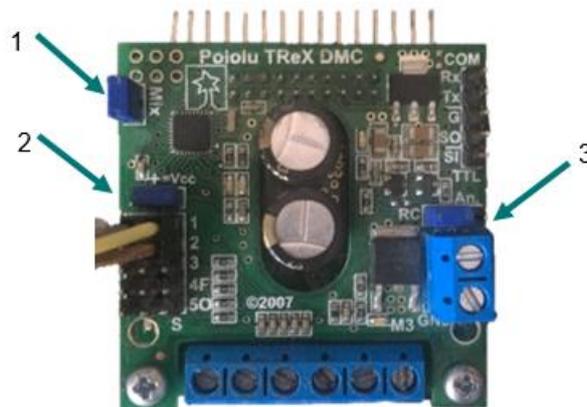


Figura 20. Jumpers Pololu TRex DMC.

3.1.8. Sensor de proximidad

Para el sistema de evasión de obstáculos es necesario un sensor que mida la distancia de un objeto (obstáculo) respecto al sensor. Como se menciona en el Capítulo 2, Sección 2.9, con un sensor fotoeléctrico o ultrasónico se puede lograr medir distancias.

Los sensores fotoeléctricos presentan ciertas características que los hacen mejores en comparación a los sensores ultrasónicos. Los sensores fotoeléctricos son más precisos, al tener frecuencias de lecturas por encima de los 100 Hz se puede lograr un análisis del entorno más confiable, pueden cubrir un rango de operación mayor a los sensores ultrasónicos y permiten realizar lecturas tridimensionales.

3.1.8.1. Análisis de los sensores de proximidad.

A continuación, en la Tabla 30, se realiza un análisis de los sensores fotoeléctricos de bajo costo que se pueden encontrar en el mercado.

Tabla 30
Sensores Fotoeléctricos de bajo costo

Características	TF Mini LiDAR	RPLiDAR A1
Rango de operación	0.3 – 12m	0.15 – 12 m
Rango Angular	0 ° (single point)	0 – 360 °
Frecuencia de Lectura	100 Hz	8000 – 8010 Hz
Rango de resolución	1 cm	0.5 mm
Consumo energía	140 mA	300mA

Tomado de Benewake (Beijing), 2017; Slamtec, 2018

Luego del análisis realizado se concluye que el sensor a usar es el sensor TF mini LiDAR, que se observa en la Figura 21, debido a su bajo costo, bajo consumo de energía y de que el rango de operación es óptimo para la lectura de datos.



Figura 21. TF Mini LiDAR.

Tomado de Benewake (Beijing), 2017

3.1.8.2. TFmini LiDAR

El sensor TFmini opera tanto en exteriores como en interiores, sin embargo, su confiabilidad disminuye conforme aumenta la intensidad de luz del entorno. A continuación, en la Figura 22 se puede apreciar el rango de operación. 1) rango de 0-30 cm corresponde al punto ciego del sensor, 2) de 30-300 cm corresponde al rango de operación en condiciones adversas con una intensidad luminosa que alcanza los 100klux, 3) de 300-700 cm corresponde al rango de operación en condiciones normales de luz alrededor de los 70klux, 4) de 700-1200 cm corresponde al rango de operación para interiores o entornos con una intensidad de luz pobre.

Además, en la Figura 22 la variable *Effective side length* corresponde al ancho del objeto mínimo para obtener una lectura sin errores.

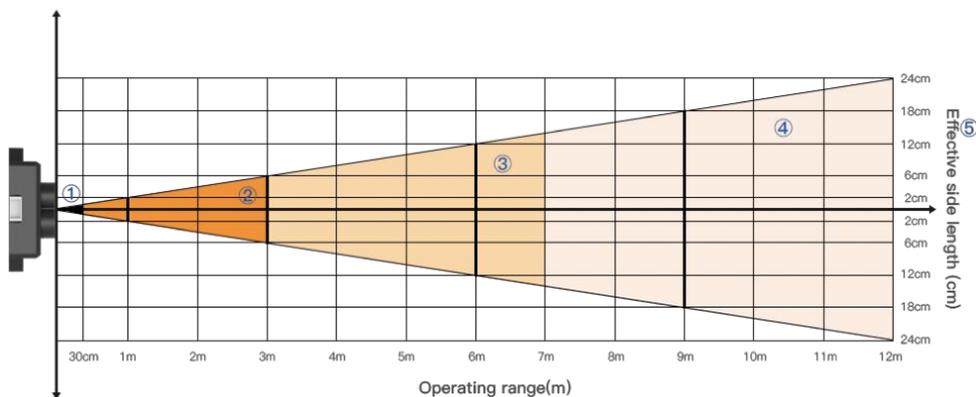


Figura 22. Rango de operación LiDAR.

Tomado de Benewake (Beijing), 2017.

3.1.8.3. Rango de operación y rango de evasión

Debido a que en este tema de titulación se plantea el uso del UGV en exteriores, se utiliza el rango de operación de 30-300 cm para una máxima confiabilidad en el sistema con condiciones de luz adversas.

En base a la velocidad del vehículo y la aplicación de este se plantea un rango de evasión entre 30-100 cm, configuración que se realiza en el Capítulo 3,

sección 3.2.6.4. El rango de evasión indica al UGV que si detecta un objeto dentro del rango definido este debe evadirlo.

3.1.9. Actuador del mecanismo de rotación del sensor de proximidad

Como se menciona anteriormente en la Tabla 30 el sensor TFmini tiene un rango angular de 0° , es decir que únicamente detecta objetos que se encuentran en línea recta hacia el sensor. En la Figura 23, se puede observar que el sensor detecta únicamente el obstáculo b, los obstáculos a y c no son detectados.

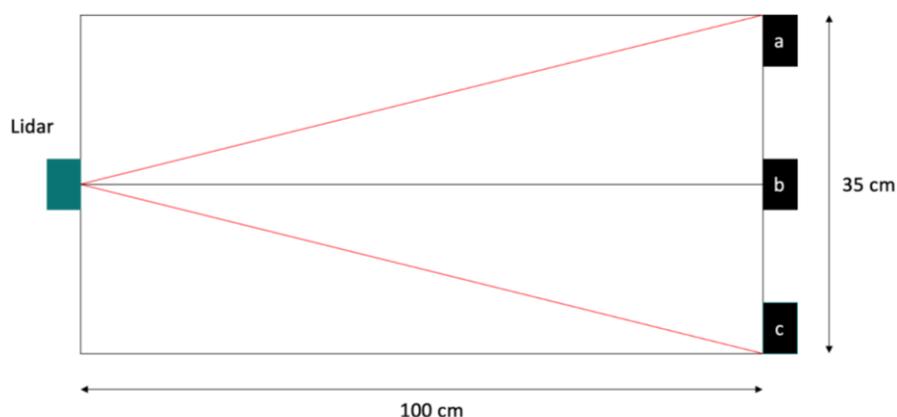


Figura 23. Detección de obstáculos.

Para ampliar el rango angular de dicho sensor se plantea el uso de un actuador, el cual permita realizar un movimiento rotativo del sensor LiDAR.

A continuación, en la Tabla 31, se realiza un análisis de dos posibles soluciones que permiten rotaciones con incrementos angulares precisos y movimientos exactos.

Tabla 31
Actuadores que ejercen movimientos rotacionales

Tipo	Modelo	Ángulo de rotación	Voltaje (V)	Velocidad (s)	Torque (N/m)	Tipo de señal
Motor paso a paso	28BYJ-48	360°	5	1/64	34	Pulsos

Servomotor	SG90	180°	4.8 - 6	0.1	0.25	PWM
-------------------	------	------	---------	-----	------	-----

Tomado de Cheung, 2018; Welten Holdings, 2016

Luego del análisis realizado se opta por usar el servomotor SG90, que se observa en la Figura 24, considerando que el motor paso a paso requiere de un driver para su funcionamiento y previo a este driver es necesario usar un microcontrolador para accionar el motor, mientras que en el servomotor tan solo se necesita de una salida PWM que se tiene integrado en el controlador de navegación Pixhawk Cube.



Figura 24. Servo Motor SG90.

Tomado de Cheung, 2018

Además, se selecciona usar el servomotor SG90 debido a su a su pequeño tamaño, y a que el torque y la velocidad del motor son suficiente para desplazar el sensor TFmini en un ángulo de 25 grados.

3.1.10. Piezas de montaje

Se diseñaron las diferentes estructuras que alojan los componentes que integran el UGV, tales como: módulos de comunicación, sensores, hardware de piloto automático, microcontroladores y módulo GPS.

En la Figura 25, se puede observar la estructura que está montada sobre el servomotor, y da soporte al sensor de distancia LiDAR TFmini. El diseño de la estructura permite la rotación del sensor.



Figura 25. Estructura de soporte de LiDAR y servomotor

En la Figura 26, se observa la estructura de soporte para el sistema de radio control y telemetría, los cuales deben estar alejados del GPS para no causar interferencias.

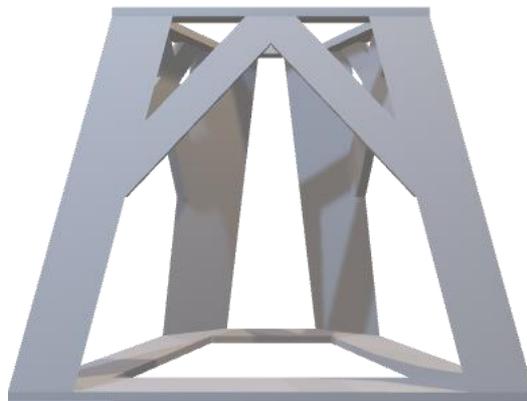


Figura 26. Estructura de soporte de radio receptor y telemetría

En la Figura 27, se observa la estructura de soporte donde se ubica el controlador de navegación Pixhawk, y en la cavidad circular se aloja el módulo GPS. El módulo GPS se encuentra a una altura superior respecto a los demás componentes. Además, en la estructura se aloja el PPM Encoder y el módulo de distribución de energía (PDB).

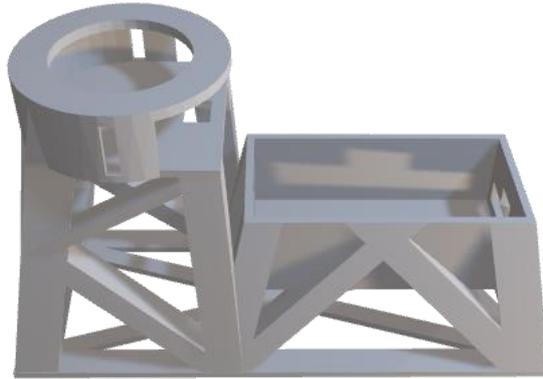


Figura 27. Estructura de soporte de Pixhawk, PPM Encoder, PDB y GPS.

3.2. Diseño de software

En esta sección se analiza el software necesario para los componentes de hardware que lo requieren, se selecciona las opciones que mejor se adaptan al UGV. Se analiza las diferentes soluciones de software de navegación, estación terrena y entornos de desarrollo.

3.2.1. Piloto Automático

El controlador de navegación Pixhawk Cube admite la ejecución de múltiples pilotos automáticos como: PX4 y ArduPilot.

3.2.1.1. Análisis piloto automático

- PX4

PX4 es un software de piloto automático de código abierto para todo tipo de vehículos aéreos, terrestres y sumergibles. El proyecto PX4 comenzó en el año 2009, pero no fue hasta marzo del 2012 que se lanzó la versión inicial. El piloto automático trabaja bajo una licencia tipo BSD, es desarrollado en el lenguaje de programación C++ y se ejecuta en sistemas operativos como: NuttX, Linux y MacOS, sin embargo, esta optimizado para el sistema operativo NuttX.

PX4 es un piloto automático generalizado para todo tipo de vehículos, donde es necesario la configuración de parámetros para establecer el tipo de vehículo a

utilizar. El código de PX4 al ser un código abierto puede ser modificado para crear soluciones personalizadas (PX4 Dev Team., 2018).

- ArduPilot

ArduPilot es un proyecto de piloto automático que comenzó en 2007, sin embargo, el lanzamiento de la versión inicial fue en el 2009. El piloto automático ArduPilot trabaja bajo una licencia GLP y un código abierto para múltiples vehículos al igual que PX4.

ArduPilot es un software multiplataforma, desarrollado en el lenguaje de programación C++ y Python. El código de ArduPilot cuenta con un código fuente dedicado para cada tipo de vehículo (ArduPilot Dev Team, 2018).

Para este trabajo de titulación se usa el piloto automático ArduPilot tomando en consideración que se cuenta con un código fuente propio para un vehículo terrestre autónomo. El código fuente al ser desarrollado específicamente para vehículos terrestres tiene librerías dedicadas y un código más ordenado lo que permite una más fácil comprensión.

Adicional, ArduPilot cuenta en su página oficial con documentación completa y detallada tanto para usuarios como para desarrolladores y al ser un software multiplataforma se puede ejecutar en diferentes sistemas operativos.

3.2.1.2. Piloto Automático ArduPilot

El código de ArduPilot es un software realmente extenso con más de 700.000 líneas de código C++. Y para comprenderlo de manera sencilla es necesario conocer su estructura básica. En la Figura 28, se puede observar un diagrama con la estructura de ArduPilot, el cual esta dividido en tres secciones principales (ArduPilot Dev Team, 2018).

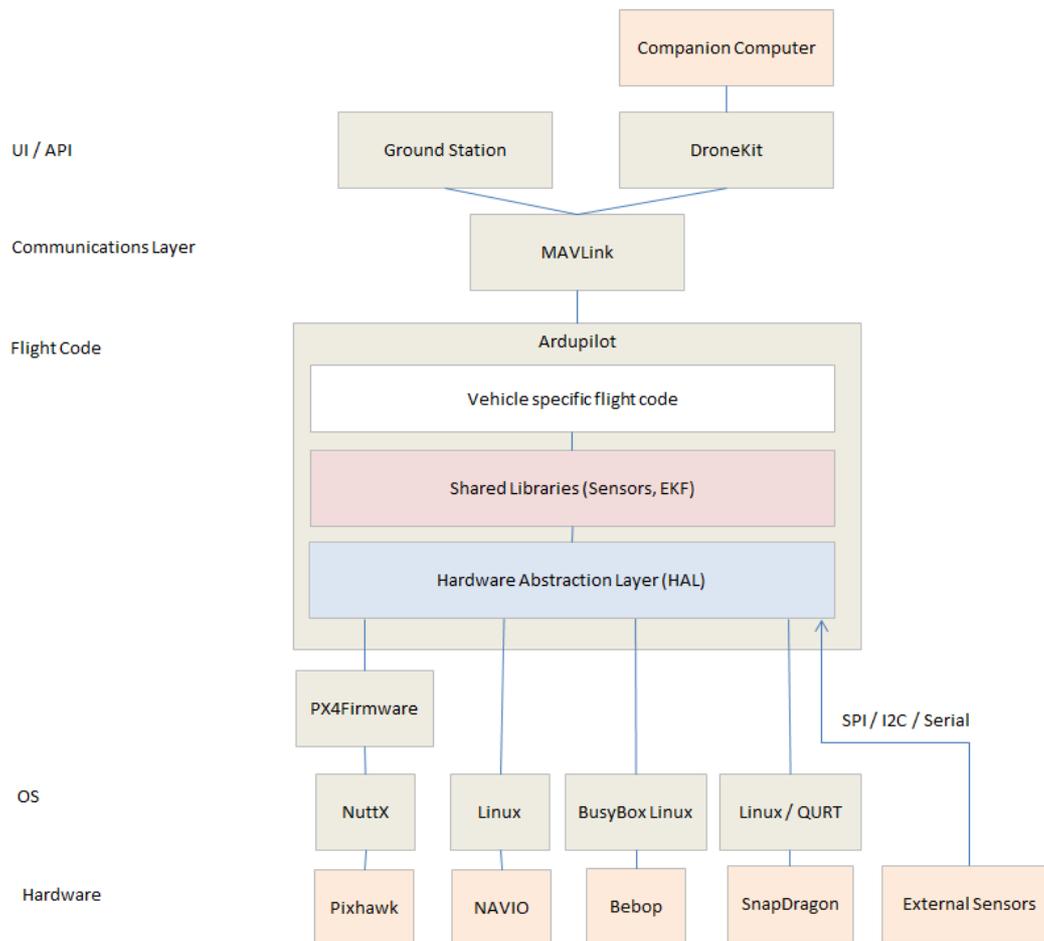


Figura 28. Estructura ArduPilot.

Tomado de ArduPilot Dev Team, 2018

El código del vehículo es la capa superior de ArduPilot, el cual hace referencia a un directorio donde se agrupan un conjunto de elementos específicos para cada tipo de vehículo. En la segunda capa se encuentran las librerías compartidas, las cuales son compatibles con cualquier vehículo. Por último, la capa de abstracción de Hardware brinda a ArduPilot la capacidad de ser cargado sobre un sin número de plataformas (ArduPilot Dev Team, 2018).

3.2.1.3. Entorno de Desarrollo

El entorno de desarrollo más recomendable para ArduPilot es Linux ya que este es el sistema operativo principal en la comunidad, por consiguiente, es el mayor optimizado para dicha tarea. Por todo lo mencionado para este proyecto de

titulación se usa Linux Ubuntu como entorno de desarrollo para la construcción del código.

3.2.1.4. Descarga Código Fuente

El proyecto de ArduPilot está basada en una licencia GPL, la cual propone compartir con la comunidad los nuevos avances y mejoras. Esto no es obligatorio para los usuarios, a menos que se pretenda vender copias de este. Sin embargo, para los desarrolladores que se encuentran continuamente realizando actualizaciones del código se utiliza la aplicación Git para la administración y GitHub como repositorio (ArduPilot Dev Team, 2018).

El primer paso para obtener una copia del código es la instalación de Git, que es la aplicación cliente que nos permitirá administrar los avances y mantener actualizado el código. Sin embargo, debido a que esta es una aplicación por consola para mayor comodidad es posible instalar una aplicación con GUI como GitKraken para Ubuntu. Si se desea un mayor detalle de la instalación de Git se puede visitar la sección de desarrollo de la página oficial de ArduPilot.

Fork the Repository, este paso consiste en crear un nuevo repositorio bajo el id de un usuario de GitHub, es decir que este nuevo repositorio está basado en el repositorio maestro, con el objetivo de compartir y mantener actualizado el código de ArduPilot. El nuevo repositorio creado se encuentra en la nube por lo que es necesario clonarlo en el pc de desarrollo.

3.2.1.5. Preparación del entorno de desarrollo

Una vez clonado el código de ArduPilot, es necesario ciertos paquetes para poder compilar el código. Estos paquetes se instalan ejecutando un script que se encuentra el repositorio de ArduPilot, por lo que es necesario únicamente ejecutar (ArduPilot Dev Team, 2018).

Tools/scripts/install-prereqs-ubuntu.sh -y

Ahora para recargar el path de instalación se debe ejecutar la siguiente línea de comando.

`./~/profile`

Es recomendable cerrar y abrir sesión.

3.2.1.6. Compilación del Código fuente

Anteriormente para compilar el código ArduPilot se utilizaba *make-based Build System*, sin embargo, el sistema de compilación a migrado lentamente, y actualmente se utiliza *waf Build System*. Para realizar la compilación del código ArduPilot, *waf Build System* debe ser siempre llamado desde la carpeta raíz del repositorio de ArduPilot (De Marchi, Mackay, Tridgell y De Sousa, 2018).

Para compilar el código es necesario configurar la tarjeta sobre la cual se ejecutará el código, además de la versión del firmware de esta. (PX4 Dev Team, 2018). Para este tema de titulación se usa un Pixhawk Cube (PX4) y una versión del firmware 3 (v3), se ejecuta el siguiente comando:

```
./waf configure --board px4-v3
```

Por último, para compilar simplemente se debe seleccionar el tipo de vehículo ya sea *antennatracker*, *copter*, *plane*, *rover*. En el caso del UGV el tipo de vehículo es *rover*.

```
./waf rover
```

Una vez compilado el código fuente su resultado es ubicado en el repositorio de ArduPilot en la ruta "*build/<board>/bin/*", en donde *board* representa la tarjeta que se configuro previamente. En el caso del UGV el archivo compilado se encuentra en la ruta "*ArduPilot/build/px4-v3/bin/ArduRover*".

3.2.2. Estación terrena

Para la configuración de parámetros, control del vehículo, planificación de misiones y calibración de sensores existen múltiples aplicaciones de estaciones terrenas.

3.2.2.1. Análisis Estación Terrena

En la Tabla 32, que se muestra a continuación, se realiza una comparación de las diferentes estaciones terrenas.

Tabla 32
Estaciones Terrenas

Estación Terrena	Plataforma	Licencia	Descripción
Mission Planner	Windows	Código abierto (GPLv3)	Estación terrena completa y ampliamente utilizada.
APM Planner	Windows, Mac OS X, Linux	Código abierto (GPLv3)	Tiene un conjunto de funciones reducidas en comparación con Mission Planner. Optimizado para el controlador de navegación APM.
MAVProxy	Linux	Código abierto (GPLv3)	Usado principalmente por desarrolladores de aviones. Tiene una interfaz de línea de comandos con módulos gráficos para la edición de mapas y misiones.

QGroundControl	Windows, Mac OS X, Linux, Android y iOS	Código abierto (GPLv3)	Funciona con pilotos automáticos que admiten el protocolo de comunicación MAVLink. Es multiplataforma, se ejecuta en todas las plataformas de escritorio y móviles.
-----------------------	--	------------------------------	--

Tomado de ArduPilot Dev Team, 2018

Luego del análisis realizado para la implementación se utiliza la estación terrena QGroundControl y Mission Planner, tomando en consideración que Mission Planner esta optimizado para el piloto automático ArduPilot y que QGroundControl presenta una interfaz gráfica amigable tanto para usuarios principiantes como para desarrolladores, además de ser un sistema multiplataforma.

3.2.2.2. Instalación de estación terrena

La estación terrena QGroundControl se ejecuta en sistemas operativos Windows, Android, MacOS y Linux; mientras que Mission Planner se ejecuta tan solo en el sistema operativo Windows.

Para instalar las aplicaciones tan solo es necesario descargar el instalador y seguir las instrucciones del asistente de instalación para completar el proceso de configuración. En el proceso se instalarán automáticamente los controladores de software necesarios.

3.2.3. Configuración de Telemetría

Para la configuración de los módulos de telemetría Xbee S2C es necesario de la aplicación XCTU, la cual se conecta a los Xbee mediante comunicación serial.

Una vez conectado el Xbee se debe cambiar el firmware a la versión ZIGBEE TH Reg, utilizando la opción de Update en la interfaz.

3.2.3.1. Configuración del módulo GroundStation

Para la configuración del módulo Xbee el cual se conecta a la estación terrena, se debe cambiar los parámetros expuestos en la Tabla 33, como se muestra en la Figura 29.

Tabla 33
Parámetros Módulo Xbee GroundStation

Parámetro	Valor
PAN ID	105
Coordinador Enable	1
Node Identifier	GROUNDSTATION
Baud Rate	57600

Id de Red

Read Write Default Update Profile

Parameter

ID PAN ID	105	
SC Scan Channels	7FFF	Bitfield
SD Scan Duration	3	exponent
ZS ZigBee Stack Profile	0	
NJ Node Join Time	FF	x 1 sec
NW Network Watchdog Timeout	0	x 1 minute
JV Channel Verification	Disabled [0]	
JN Join Notification	Disabled [0]	
OP Operating PAN ID	0	
OI Operating 16-bit PAN ID	FFFF	
CH Operating Channel	0	
NC Number of Remaining Children	14	
CE Coordinator Enable	Enabled [1]	
DO Device Options	0	Bitfield
DC Device Controls	0	Bitfield
SH Serial Number High	13A200	
SL Serial Number Low	41665719	
MY 16-bit Network Address	0	
MP 16-bit Parent Address	FFFE	
DH Destination Address High	0	
DL Destination Address Low	FFFF	
NI Node Identifier	GROUNDSTATION	
NH Maximum Hops	1E	
BH Broadcast Radius	0	
AR Many-to-One Route Broadcast Time	FF	x 10 sec
DD Device Type Identifier	A0000	
NT Node Discovery Backoff	3C	x 100 ms
NO Node Discovery Options	0	
Serial Interfacing Change modem interfacing options		
BD Baud Rate	57600 [6]	
NB Parity	No Parity [0]	
SB Stop Bits	One stop bit [0]	
RO Packetization Timeout	3	x character times
D6 Pin 16 - DIO6/nRTS Configuration	Disable [0]	
D7 Pin 12 - DIO7/nCTS Configuration	nCTS flow control [1]	
AP API Enable	Transparent mode [0]	
AO API Output Mode	Native [0]	

Coordinador

Identificador

Baudrate

Figura 29. Configuración Xbee GroundStation

3.2.3.2. Configuración del módulo Onboard

De igual manera para la configuración del módulo Xbee, el cual está a bordo del UGV se debe cambiar los parámetros expuestos en la Tabla 34, como se muestra en la Figura 30.

Tabla 34
Parámetros Módulo Xbee Onboard

Parámetro	Valor
PAN ID	105
Node Identifier	ONBOARD
Baud Rate	57600

Read Write Default Update Profile

Parameter + -

Id de Red

i	ID PAN ID	105			
i	SC Scan Channels	7FFF	Bitfield		
i	SD Scan Duration	3	exponent		
i	ZS ZigBee Stack Profile	0			
i	NJ Node Join Time	FF	x 1 sec		
i	NW Network Watchdog Timeout	0	x 1 minute		
i	JV Channel Verification	Disabled [0]			
i	JN Join Notification	Disabled [0]			
i	OP Operating PAN ID	105			
i	OI Operating 16-bit PAN ID	D2F7			
i	CH Operating Channel	16			
i	NC Number of Remaining Children	14			
i	CE Coordinator Enable	Disabled [0]			
i	DO Device Options	0	Bitfield		
i	DC Device Controls	0	Bitfield		
i	SL Serial Number Low	416656B4			
i	MY 16-bit Network Address	C313			
i	MP 16-bit Parent Address	FFFE			
i	DH Destination Address High	0			
i	DL Destination Address Low	0			
i	NI Node Identifier	ONBOARD			
i	NH Maximum Hops	1E			
i	BH Broadcast Radius	0			
i	AR Many-to-One Route Broadcast Time	FF	x 10 sec		
i	DD Device Type Identifier	A0000			
i	NT Node Discovery Backoff	3C	x 100 ms		
i	NO Node Discovery Options	0			
i	NP Maximum Number of Transmission Bytes	54			
i	BD Baud Rate	57600 [6]			
i	NB Parity	No Parity [0]			
i	SB Stop Bits	One stop bit [0]			
i	RO Packetization Timeout	3	x character times		
i	D7 DIO7 Configuration	CTS flow control [1]			
i	D6 DIO6 Configuration	Disable [0]			
i	AP API Enable	Transparent mode [0]			
i	AO API Output Mode	Native [0]			

Identificador

Baudrate

Figura 30. Configuración Xbee Onboard

3.2.4. Configuración Inicial del control de navegación

Para la configuración inicial del controlador de navegación Pixhawk se establece la comunicación con la estación terrena QGroundControl por conexión USB. La comunicación también puede establecerse mediante telemetría, sin embargo, al tratarse de ajustes iniciales es recomendable hacerlo por conexión USB para garantizar que no se presente pérdida de datos en la comunicación lo que ocasiona errores en la configuración que afecta al correcto funcionamiento del vehículo.

Una vez establecida la comunicación, en la estación terrena se muestra una pantalla de configuración inicial, que se puede observar en la Figura 31, donde se realiza la calibración de sensores, configuración de modos de navegación, configuración y calibración de radio control, calibración de batería, todo esto con la ayuda de un asistente de configuración. Además de la calibración de sensores y configuración de parámetros, en esta configuración inicial se puede cargar el firmware o Piloto automático (QGroundControl, 2019).

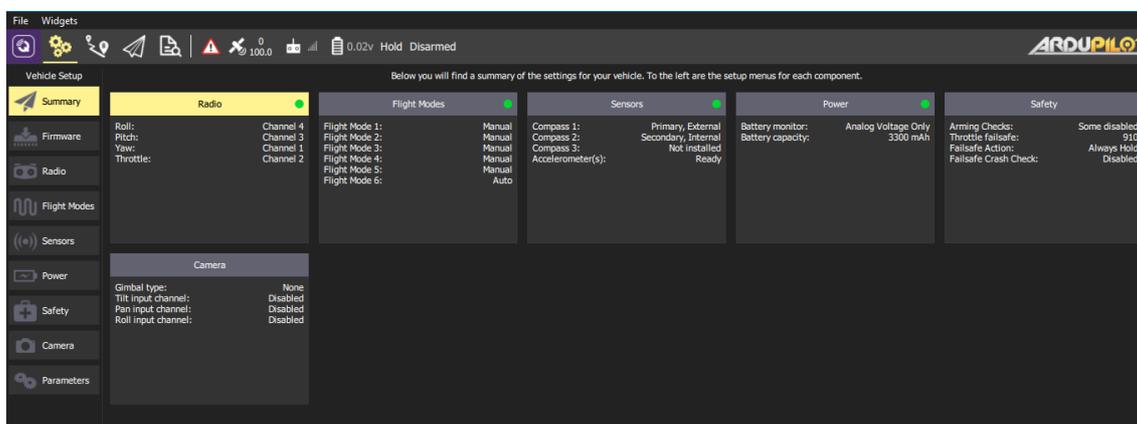


Figura 31. Parámetros de Configuración del Vehículo.

3.2.4.1. Firmware

Para entrar en modo de configuración y cargar un nuevo piloto automático, se selecciona la sección Firmware, y reiniciar el controlador de navegación Pixhawk. La estación terrena QGroundControl detecta automáticamente el controlador de navegación, muestra una lista de opciones donde es posible

elegir que firmware cargar. Como se menciona en el Capítulo 3, Sección 3.2.1.1, se trabaja con el piloto automático ArduPilot. Al seleccionar la opción de ArduPilot, se debe especificar el tipo de vehículo (*rover*).

De manera predeterminada, QGroundControl instala la versión de firmware más estable, pero también es posible instalar una versión beta, versión desarrollador o una versión personalizada (QGroundControl, 2019). En la Figura 32, se puede observar el entorno de configuración del Firmware.

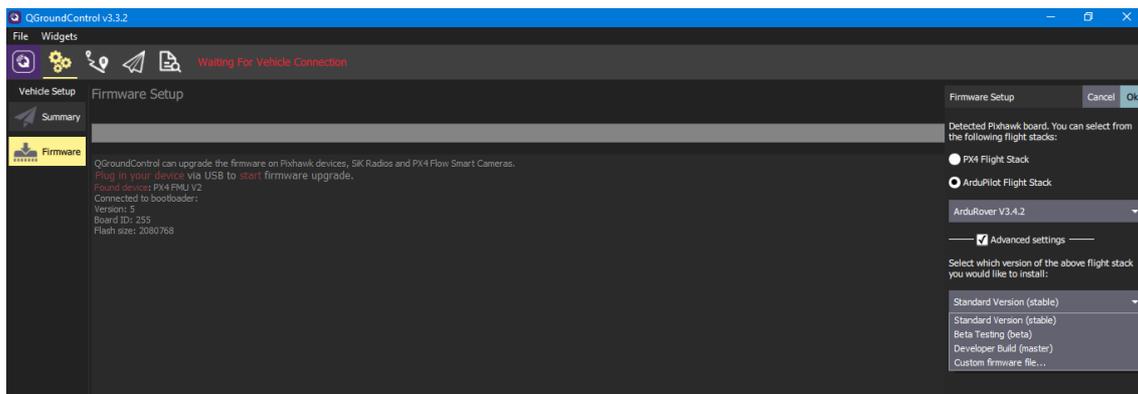


Figura 32. Carga de firmware.

3.2.4.2. Calibración de sensores

En la sección de calibración de sensores es posible calibrar la brújula, el giroscopio y el acelerómetro del vehículo, para ello tan solo basta con seleccionar en la lista que se muestra en la Figura 33, cada uno de los sensores para iniciar con la calibración y seguir las instrucciones del asistente.

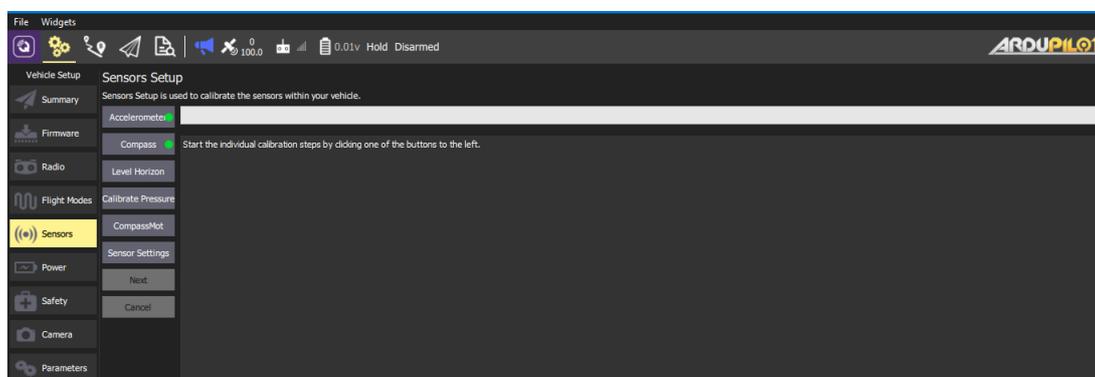


Figura 33. Entorno de calibración de sensores.

3.2.4.2.1. Acelerómetro

Para la calibración del acelerómetro se debe colocar el vehículo en las posiciones: *level*, *left side*, *right side*, *nose down*, *nose up*, *on its back*, (ArduPilot Dev Team, 2018) cómo se observa en la Figura 34, instrucciones que se indican en el asistente de calibración Figura 35.

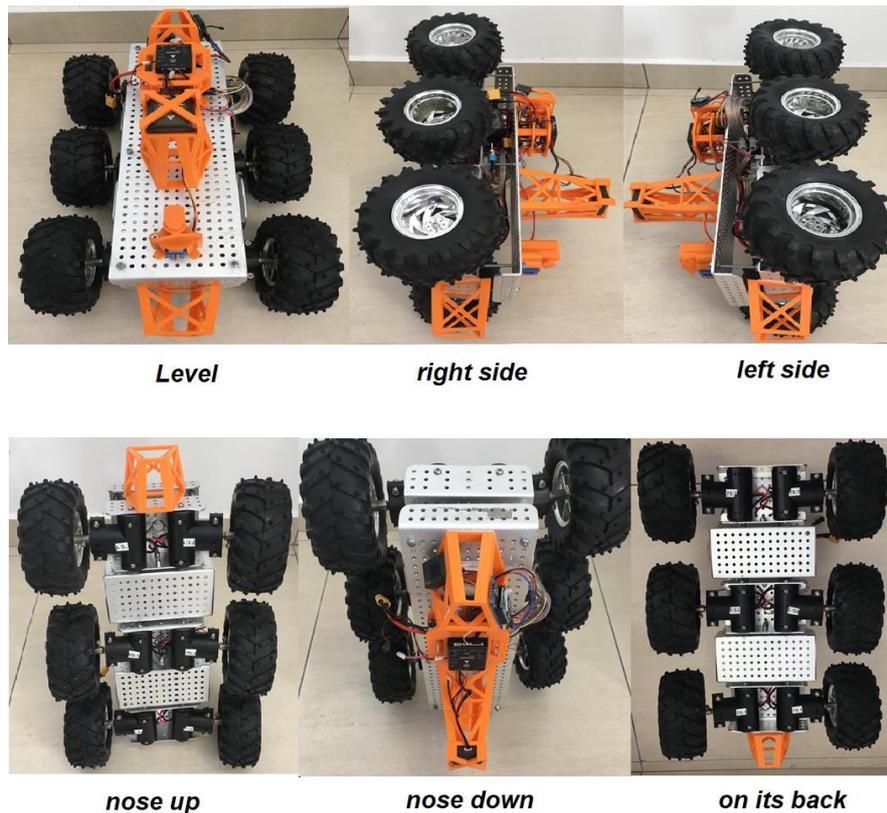


Figura 34. Posiciones de calibración acelerómetro

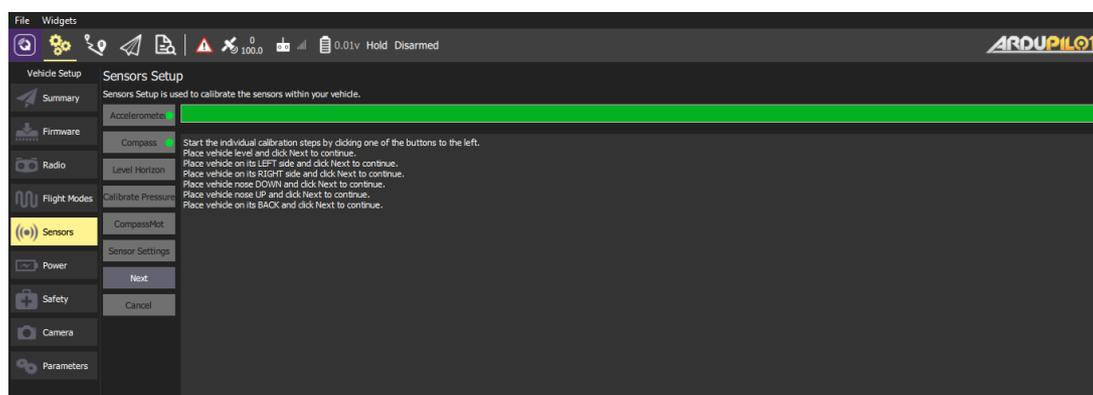


Figura 35. Asistente de calibración acelerómetro.

3.2.4.2.2. Brújula

Para la calibración de la brújula se procede a girar el vehículo alrededor de todos los ejes en todas las orientaciones hasta que se complete el proceso (ArduPilot Dev Team, 2018), como se indica en el asistente Figura 36.

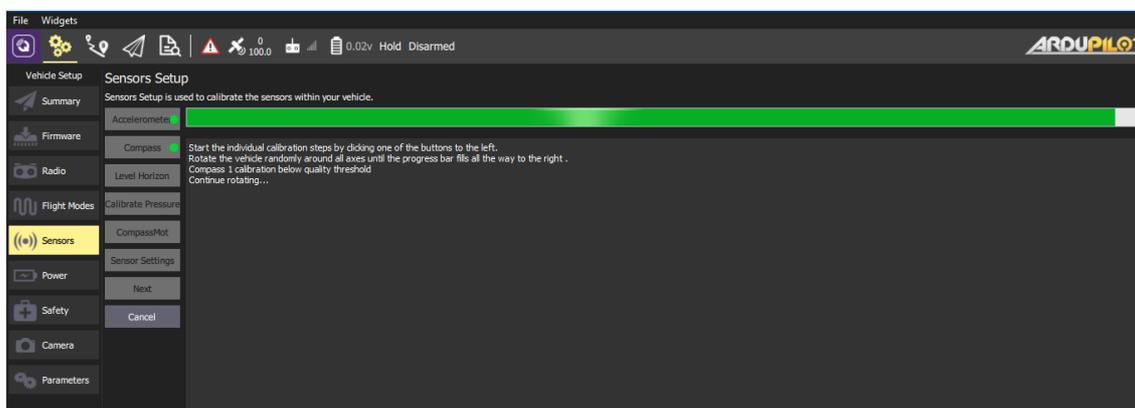


Figura 36. Asistente de calibración brújula

3.2.4.3. Radio control

Para el correcto control manual del vehículo es necesario la asignación de los canales de control y calibración de las palancas del radio transmisor.

3.2.4.3.1. Configuración

El radio transmisor se usa para controlar los movimientos del vehículo, en el *rover* se controla únicamente la aceleración y la dirección. En la Tabla 35, se describen los movimientos sobre los ejes correspondientes a cada palanca del radio transmisor.

Tabla 35

Movimientos de los ejes en palancas del radio transmisor

	Palanca Izquierda	Palanca Derecha
Modo 1	Elevación (pitch)	Aceleración (throttle)
	Dirección (yaw)	Rotación (roll)

Modo 2	Aceleración (throttle)	Elevación (Pitch)
	Dirección (yaw)	Rotación (roll)

Tomado de ArduPilot Dev Team, 2018

El transmisor puede ser configurado en dos modos de operación como se muestra en la Figura 37.

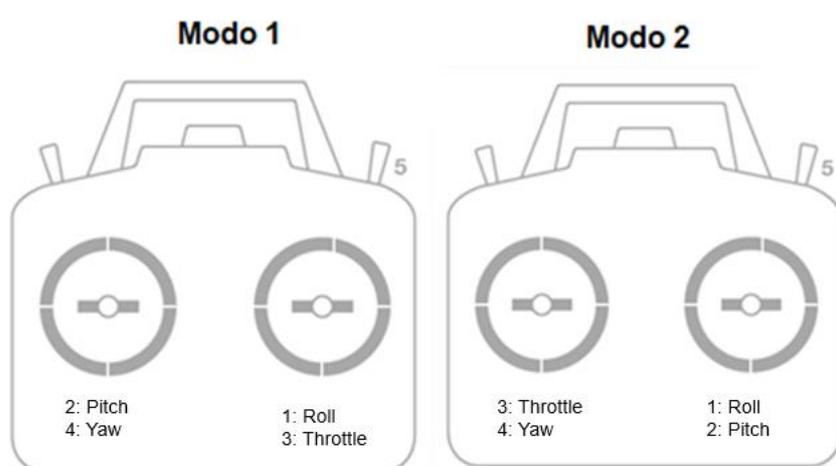


Figura 37. Modos de operación radio transmisor.

Tomado de ArduPilot Dev Team, 2018

El radio transmisor utilizado en la implementación está configurado en el modo de operación dos.

3.2.4.3.2. Calibración

En la sección de Radio se puede calibrar los canales del radio, para iniciar el proceso se debe tomar en cuenta que todos los *trim* deben estar centrados.

Para calibrar se debe mover cada una de las palancas en su rango completo para registrar las posiciones máxima y mínima como se indica en el asistente de configuración. En la Figura 38 y Figura 39, se muestra la calibración del canal throttle, mismo proceso que se realiza para la calibración de los demás canales.

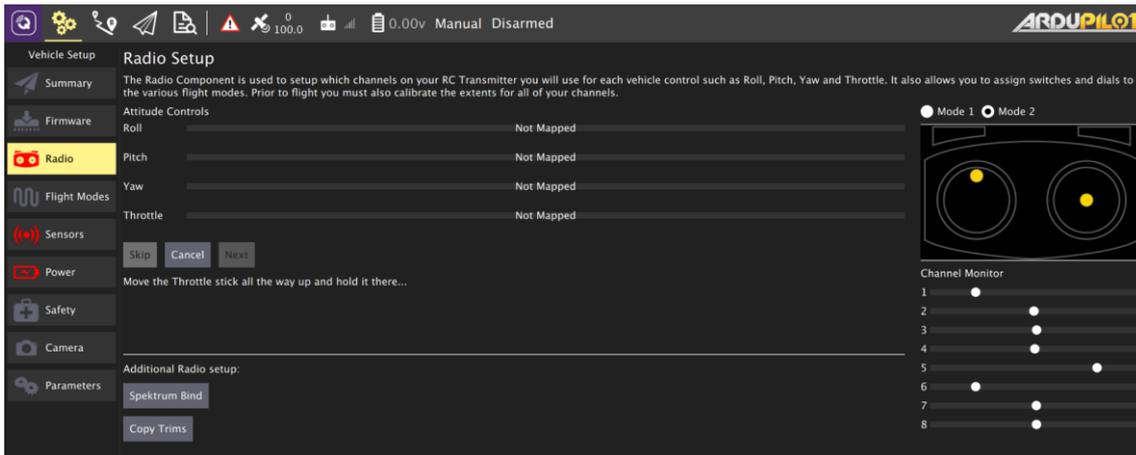


Figura 38. Calibración del canal throttle máximo

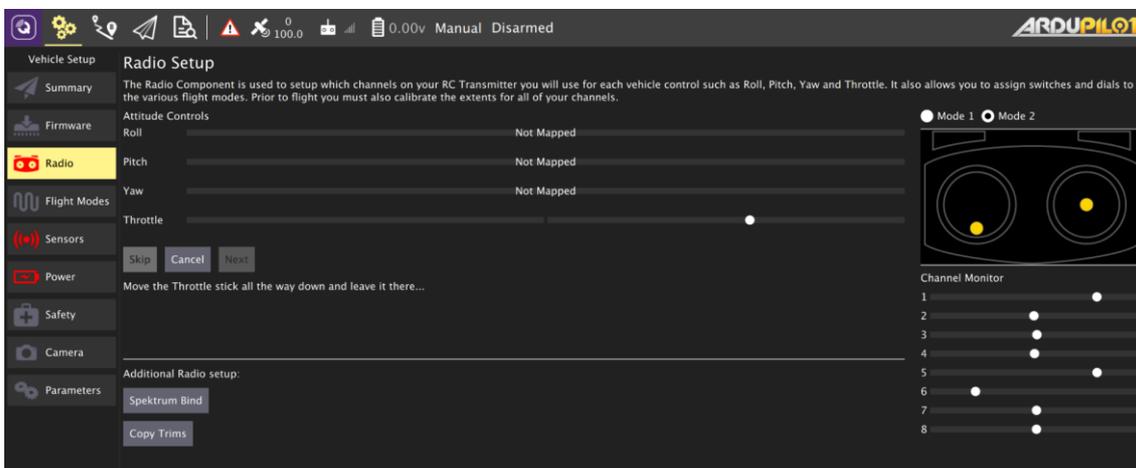


Figura 39. Calibración del canal throttle mínimo

3.2.4.4. Configuración Modos de navegación

Los modos de navegación son usados para proporcionar al vehículo diferentes tipos de navegación. La configuración de modos de navegación permite asignar hasta 6 modos de navegación en un solo canal, cambiando el modo según el valor PWM del canal.

Para la configuración tan solo basta con seleccionar el número de canal, en nuestro caso es el canal 5 el cual corresponde a un interruptor, y se le asigna el modo de navegación manual y automático que se activa según el valor PWM del canal. Los rangos de PWM para cada modo de navegación se muestran junto a la selección del modo de navegación.

En la Figura 40, se puede observar la configuración de los modos de navegación y los rangos PWM correspondientes.

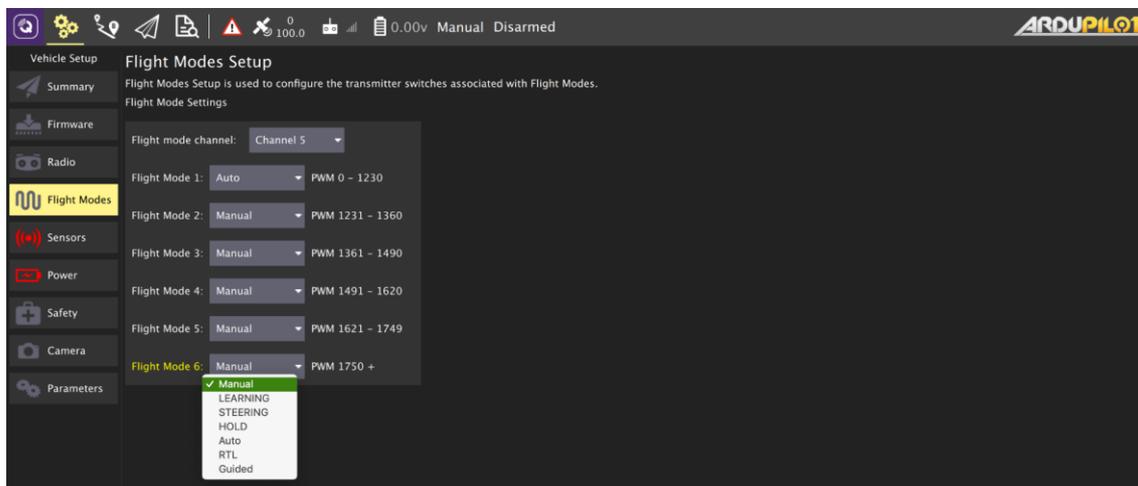


Figura 40. Entorno de configuración modos de navegación

- Modo Manual

La navegación es totalmente asistida, el usuario tiene control del vehículo.

- Modo Automático

En el modo automático el vehículo cumple con misiones programadas en un entorno controlado.

3.2.4.5. Guardar Copia de Parámetros

Una vez que se termine con la configuración y calibraciones se pueden guardar los parámetros, de esta manera no se pierde la configuración y tan solo basta cargar los parámetros del archivo existente. En la Figura 41, se muestra el entorno donde se puede realizar una copia de los parámetros.

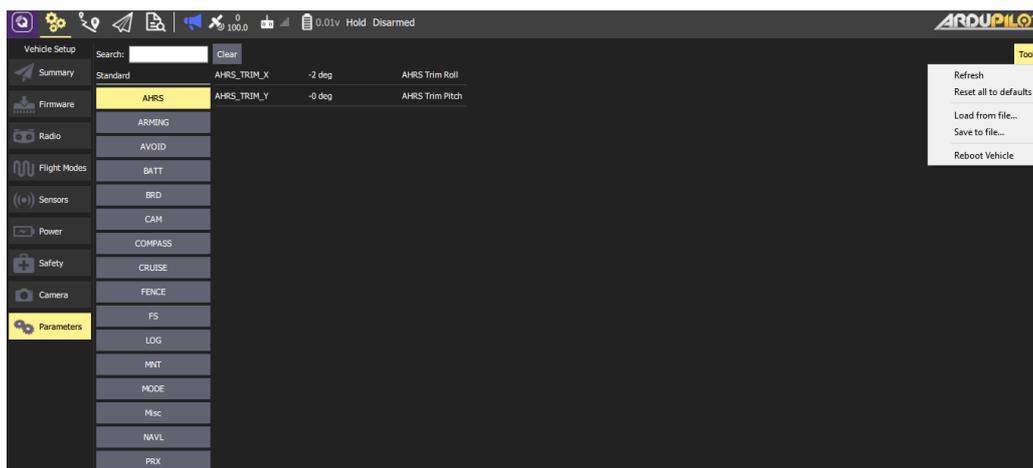


Figura 41. Entorno guardar copia de Parámetros.

3.2.5. Configuración Avanzada

Esta sección presenta opciones de configuración y ajustes más avanzados para controlar el comportamiento del vehículo. Se trata de una lista de parámetros avanzados asociados al funcionamiento del vehículo tanto en modo de navegación manual como en modo de navegación autónoma.

3.2.5.1. Ajustes de parámetros Modo manual

En la Tabla 36, que se muestra a continuación se detallan los parámetros que pueden ser configurados para cambiar el comportamiento del vehículo en el modo de navegación manual.

Tabla 36

Lista de parámetros modo manual

Parámetro	Descripción	Valor por defecto	Rango
RCMAP	Configurar la asignación del canal de entrada RC.	RCMAP_PITCH 2 RCMAP_ROLL 1 RCMAP_THROTTLE 3 RCMAP_YAW 4	1 - 8

SERVO_	Invertir el canal de	SERVO2_REVERSED	0 normal
REVERSED	salida.	0	1 reversed
SERVO_	Asignar la función	SERVO2_FUNCTION	0 disabled
FUNCTION	del canal de salida.	0	26 Ground
		SERVO4_FUNCTION	0 Steering
			70 Throttle

Tomado de ArduPilot Dev Team, 2019

3.2.5.2. Ajustes de parámetros Modo autónomo

En la Tabla 37, que se muestra a continuación se detallan los parámetros que pueden ser configurados para cambiar el comportamiento del vehículo en el modo de navegación autónoma.

Tabla 37
Lista de parámetros modo autónomo

Parámetro	Descripción	Valor por defecto	Rango
CRUISE_ SPEED	Configurar velocidad de navegación del vehículo.	2 m/s	0 - 100
CRUISE_ THROTTLE	Configurar la aceleración necesaria para alcanzar la velocidad de navegación del vehículo.	50 %	0 - 100
WP_RADIUS	Establecer el radio del punto de la trayectoria.	2 m	0 - 1000

WP_ OVERSHOOT	Una vez que el vehículo alcance el waypoint este se mantendrá dentro del waypoint y accionara los motores cuando se supere el valor definido.	2 m	0 - 10
TURN_ RADIUS	Configurar radio de giro del vehículo	0.9 m	0 - 10
TURN_ MAX_G	Configurar la aceleración máxima de giro.	0.6 gravities	0.1 - 10

Tomado de ArduPilot Dev Team, 2019

3.2.5.3. Ajustes de parámetros evasión de obstáculos

En la Tabla 38, que se muestra a continuación se detallan los parámetros que pueden ser configurados para establecer el comportamiento del sensor para la evasión de obstáculos.

Tabla 38
Lista de parámetros evasión de obstáculos

Parámetro	Descripción	Valor por defecto	Rango
SERIAL_ PROTOCOL	Configurar que tipo de protocolo se usa en el puerto.	SERIAL2_ PROTOCOL 1	-1: ninguno 1: Mavlink1 9: rangefinder 11 lidar360

SERIAL_ BAUD	La velocidad en baudios del puerto.	SERIAL2_ PROTOCOL 57	1: 1200 19: 19200 57: 57600 115: 115200 921: 921600
-------------------------	-------------------------------------	---------------------------------	---

RNGFND_ TYPE	Establecer el tipo de sensor.	0	1: Analógico 8: LightWare Serial 10: MAVLink 19: Benewake TF02 20: Benewake TFmini
-------------------------	-------------------------------	---	--

RNGFND_ MIN_CM	Establecer la distancia mínima que el sensor puede leer de manera confiable.	20 cm	--
---------------------------	--	-------	----

RNGFND_ MAX_CM	Establecer la distancia máxima que el sensor puede leer de manera confiable.	700 cm	--
---------------------------	--	--------	----

RNGFND_ TRIGGR_CM	Configurar la distancia respecto al obstáculo a la que el vehículo girara para evitar el obstáculo.	100 cm	0 - 1000
------------------------------	---	--------	----------

RNGFND_ TURN_ANGL	Configurar la dirección y el ángulo de giro para evitar un obstáculo.	45 grados	-450 – 450 Negativo giro izquierda Positivo giro derecha
RNGFND_ TURN_TIME	Configurar tiempo de giro para evitar el obstáculo.	1 s	0 - 100
SERVO_ FUNCTION	Asignar la función del canal de salida.	SERVO5_ FUNCTION 0	0: disabled 90: liddarservo

Tomado de ArduPilot Dev Team, 2019

3.2.6. Evasión de obstáculos

En esta sección se configuran todos los componentes que integran el sistema de evasión de obstáculos. Además, se realiza la modificación del código fuente ArduPilot para accionar el servomotor el cual conjunto a la estructura de la Figura 25 permite la rotación del sensor de distancia.

3.2.6.1. Selección del método de evasión de obstáculos.

El piloto automático ArduRover integra en su código dos métodos de evasión de obstáculos los cuales se analiza en la Tabla 39 a continuación.

Tabla 39
Métodos de evasión de obstáculos.

Método de evasión de obstáculos	Dodge	Simple
Funcionamiento	Detección y evasión	Detección y detención

Modos de navegación compatibles	Steering, SmartRTL, Auto.	RTL, Guided.	Acro, Steering, Loiter, Guided.
Sensores compatibles	Lidar single point, sonar ultrasónico.	Lidar single point, Lidar 2D, sonar ultrasónico	

Tomado de ArduPilot Dev Team, 2019

3.2.6.2. Configuración del Sensor de Distancia

El sensor de proximidad TFmini envía en la trama de datos, la distancia censada en unidades de centímetros. Mientras que el piloto automático ArduPilot está diseñado para recibir dicho valor en metros.

Por lo que es necesario configurar el sensor TFmini, y esto se lo realiza enviando el comando “42 57 02 00 00 00 04 06” por medio de la consola BENEWAKE TF, como se muestra en la Figura 42.

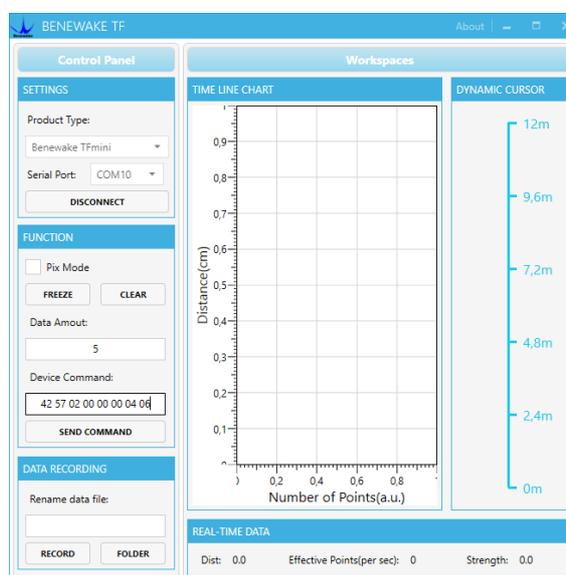


Figura 42. GUI de consola de Benewake.

3.2.6.3. Barrido del sensor TFmini LiDAR

Como se menciona en el Capítulo 3, Sección 3.1.9. Se aumenta el rango angular del sensor TFmini LiDAR utilizando el servomotor SG90.

Para accionar el servomotor SG90 se realiza la modificación del código fuente ArduPilot. A continuación, en la Figura 43, se puede apreciar el diagrama de flujo del código agregado.

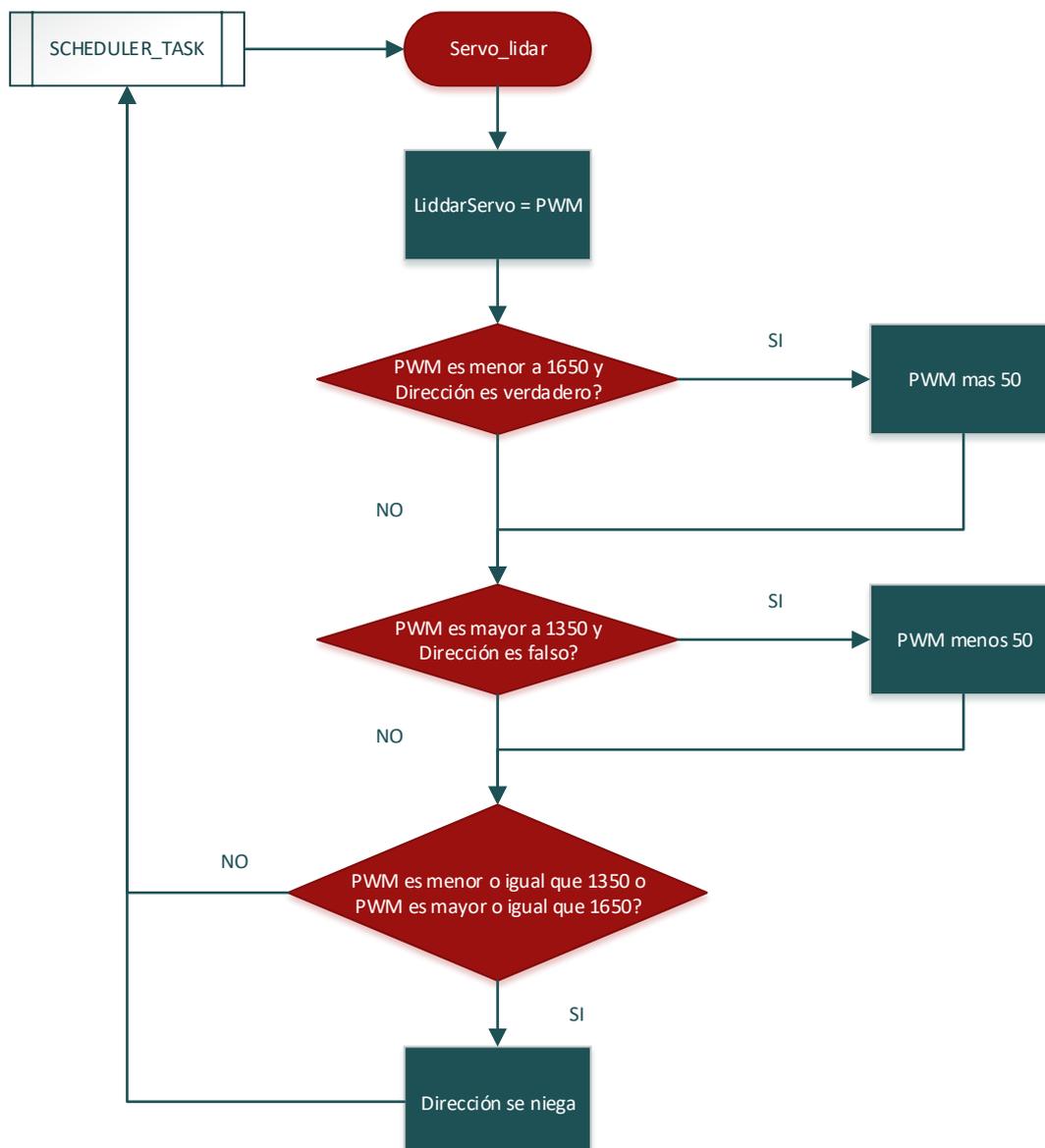


Figura 43. Diagrama de flujo servomotor

Dentro del directorio Ardupilot/libraries/SRV_Channel/ se modifica el archivo SRVChannel.h como se puede ver en la Figura 44.

```
C SRV_Channel.h x
122     k_dspoilerRight2      = 87,
123     k_winch                = 88,
124     k_mainsail_sheet      = 89,
125     k_liddar_servo         = 90,
126     k_nr_aux_servo_functions
127 } Aux_servo_function_t;
```

Figura 44. Parametro Lidar Servo.

Para accionar el servomotor es necesario el uso de variables globales con el objetivo de almacenar la posición de cada instancia llamada por el SCHEDULER_TASK, para esto se modifica el archivo header rover.h del código principal APMrover.cpp ubicado en la carpeta Ardupilot/APMrover/. A continuación, en la Figura 45.

```
C Rover.h
390 //pwm auxiliar lidar servomotor
391 uint16_t last_liddar_pwm = 1500;
392
393 //direccion auxiliar lidar servomotor
394 bool direction_lidar = true;
```

Figura 45. Variables Globales.

En el archivo APMrover.cpp a continuación, se realizaran dos cambios, 1) se crea la función servo_lidar como se muestra en la Figura 46, código encargado de accionar el servomotor.

```

APMrover2.cpp
326 void Rover::servo_lidar(void)
327
328
329
330
331     SRV_Channels::set_output_pwm(SRV_Channel::k_lidar_servo, last_lidar_pwm);
332
333     if (last_lidar_pwm < 1650 && direction_lidar) {
334         last_lidar_pwm += 50;
335     }
336
337
338     if (last_lidar_pwm > 1350 && !direction_lidar) {
339         last_lidar_pwm -= 50;
340     }
341
342
343     if (last_lidar_pwm >= 1650 || last_lidar_pwm <= 1350)
344     {
345         direction_lidar = !direction_lidar;
346     }
347

```

Figura 46. Función servo_lidar.

Y 2) como se muestra en la Figura 47, se agrega al SCHEDULER_TASK, la llamada a la función servo_lidar, además se configura la frecuencia a la que se ejecuta la función servo_lidar, y el tiempo estimado para la función.

```

APMrover2.cpp
32 #include "Rover.h"
33
34 const AP_HAL::HAL& hal = AP_HAL::get_HAL();
35
36 Rover rover;
37
38 #define SCHED_TASK(func, _interval_ticks, _max_time_micros) SCHED_TASK_CLASS(Rover, &rover, func, _interval_ticks, _max_time_micros)
39
40 /*
41  scheduler table - all regular tasks are listed here, along with how
42  often they should be called (in Hz) and the maximum time
43  they are expected to take (in microseconds)
44  */
45 const AP_Scheduler::Task Rover::scheduler_tasks[] = {
46     // Function name, Hz, us,
47     SCHED_TASK(servo_lidar, 10, 200),

```

Figura 47. Organizador de tareas.

3.2.6.4. Configuración de parámetros para el método de evasión Dodge.

A continuación, en la Tabla 40 se especifica los parámetros modificados que permiten la habilitación del sistema de evasión de obstáculos.

Tabla 40

Configuración de parámetros iniciales para la evasión de obstáculos

Parámetro	Descripción	Valor modificado
SERIAL2_ PROTOCOL	Comunicación entre el sensor TFmini LiDAR y el controlador Cube	9: rangefinder
SERIAL2_ BAUD	El baud rate del sensor TFmini LiDAR es de 115200	115: 115200
RNGFND_ TYPE	Según el Datasheet del sensor TFmini LiDAR este valor debe estar configurado en 8: LightWare Serial, y no en 20: Benewake TFmini LiDAR. Esto es debido a que las librerías de este se encuentran en versión Beta por lo que se tienen ciertos errores.	8: LightWare
RNGFND_ MIN_CM	Según el rango de operación de 30-700 cm mencionando en la sección 3.1.8.3	30
RNGFND_ MAX_CM	Según el rango de operación de 30-700 cm mencionando en la sección 3.1.8.3	700
RNGFND_ TRIGGR_CM	Según el rango de evasión de 30-100 cm mencionando en la sección 3.1.8.2	100
SERVO5_ FUNCTION	El valor 90 liddarservo es una de las modificaciones que se realizó en el código de ArduPilot para realizar el barrido del sensor TFmini.	90: liddarservo

4. Pruebas y Resultados

Una vez integrados todos los componentes mecánicos y electrónicos, software de navegación, estación terrena, sistema de radio control y sistema de telemetría que componen el UGV, como se muestra en la Figura 48, se procede a realizar pruebas para verificar el correcto funcionamiento del vehículo.



Figura 48. UGV

Antes de iniciar con la navegación se debe tomar en cuenta que la interfaz de salida de los servos no se habilita mientras el UGV no se encuentre armado. El estado predeterminado cuando se alimenta el Pixhawk es desarmado para evitar que el UGV accione los actuadores antes de encontrarse en una zona segura y así prevenir posibles accidentes.

4.1. Armar el UGV

Previo a realizar el armado del vehículo se debe verificar que, 1) no existan conexiones físicas al vehículo, 2) el receptor debe estar encendido, 3) los

controles de aceleración y dirección estén en posición neutral, y 4) que el interruptor del modo de navegación este en modo manual.

Una vez encendido el UGV, el controlador de navegación internamente realiza una serie de comprobaciones de seguridad, para verificar que no haya problemas en la configuración de parámetros y calibración de sensores. Como se había mencionado en el Capítulo 3, Sección 3.1.2.2.1, el módulo Here 2 integra el switch de seguridad, y leds que indican el estado del controlador de navegación. Los leds parpadean de color amarillo, indicando que no existen errores que impida que el vehículo se arme.

Después de verificar la configuración, se procede a presionar el botón de seguridad por tres segundos. Los leds se tornan de color verde indicando que se han habilitado las salidas de los servos, pero únicamente se tendrá control de la dirección del vehículo.

Para tener el control de la aceleración, se deberá armar el vehículo desde la estación terrena para ello se presiona el botón de armar/desarmar que se muestra en la Figura 49.



Figura 49. Control Armar Vehículo Mission Planner

4.2. Control Manual

Se procede a realizar las pruebas del control manual para observar el comportamiento de los actuadores en respuesta al radio transmisor, se verifica que los sentidos de dirección sean correctos.

Se observa que se tiene control de los motores al mover las palancas de throttle y roll con la configuración por defecto. A pesar de que se tiene control manual con la configuración por defecto, esta no facilita el control del vehículo, además no se tiene aceleración negativa debido a que el radio transmisor está diseñado únicamente para aviones y la dirección y aceleración está invertida.

Tomando en cuenta que la configuración por defecto no es la adecuada, se procede a realizar cambios en la configuración avanzada, como se observa en la Tabla 41.

Tabla 41
Modificación de parámetros modo manual

Parámetro	Descripción	Valor modificado
RCMAP_PITCH	Al canal de radio correspondiente a la elevación se le asigna la aceleración del vehículo.	3
RCMAP_THROTTLE	Al canal de radio correspondiente a la aceleración se le asigna la elevación, sin embargo, este canal no es utilizado.	2
SERVO2_FUNCTION	En esta salida se configura el <i>steering</i> o dirección del UGV	26
SERVO2_REVERSED	Para corregir el sentido de la dirección.	1

SERVO4_FUNCTION	En esta salida se configura la aceleración del UGV.	70
------------------------	---	----

Luego de los cambios realizados se comprueba que la aceleración ahora se la controla desde la palanca de pitch y el sentido de las direcciones son correctas.

4.3. Navegación Autónoma

El sistema de navegación autónoma en el piloto automático ArduPilot es conocido como auto mission, y para ejecutarla el primer paso es crear una misión, esta consiste en marcar una serie de coordenadas geográficas que corresponden a la trayectoria y además marcar la coordenada donde se encuentra el UGV o *home* a través de Mission Planner. Una vez marcados los puntos (waypoints), se crea un archivo de tipo *misión.waypoints* el cual es cargado al piloto automático. Las misiones se activan cuando se cambia el vehículo al modo de navegación AUTO.

Para registrar los puntos de trayectoria de la misión es posible realizar de dos maneras: 1) conducir el vehículo hacia el punto y registrar la posición del vehículo o 2) marcar los puntos en el mapa satelital que se presenta en la interfaz de Mission Planner.

Se procede a realizar las pruebas de navegación autónoma, marcando los puntos de la trayectoria en la estación terrena Mission Planner. En la Figura 50, se muestran los puntos de la trayectoria a seguir.



Figura 50. Puntos de trayectoria

Una vez marcada la trayectoria se procede a cargar la misión en el controlador de navegación y se activa el interruptor del radio transmisor en modo automático. El UGV inicia con la misión. En la Figura 51, se puede observar los waypoints en el área a recorrer. Una vez que el vehículo llegue al punto 4, se detiene porque ha terminado con la misión planificada.



Figura 51. Trayectoria de la misión

En la Figura 52, se puede observar la trayectoria que ha cumplido el vehículo en la misión.



Figura 52. Trayectoria realizada en la misión.

Luego de las pruebas realizadas es necesario realizar ajustes en la configuración para mejorar el recorrido del vehículo. En la Tabla 42, se procede a realizar cambios en la configuración avanzada.

Tabla 42

Modificación de parámetros modo autónomo

Parámetro	Descripción	Valor modificado
CRUISE_SPEED	En las pruebas realizadas se pudo observar por consola que el vehículo estaba navegando a su máxima velocidad de 0.7 m/s. Esta velocidad no es recomendable debido a que el vehículo está trabajando a su máxima capacidad por lo que se	0.5 m/s

	reduce la velocidad de navegación del vehículo.	
CRUISE_THROTTLE	Al momento de ejecutar la misión se pudo observar que el vehículo acelera drásticamente por lo que para reducir esta aceleración se configura este parámetro.	10 %
WP_RADIUS	Se reduce el radio del waypoint para mejorar la precisión en espacios pequeños, de esta manera se puede apreciar de mejor manera la trayectoria de la misión.	0.5 m
WP_OVERSHOOT	Se reduce el valor debido a que se reduce el valor de WP_RADIUS.	1.5 m
TURN_RADIUS	Se reduce el radio de giro del vehículo para tener giros más pronunciados.	0.1 m
TURN_MAX_G	Al realizar los giros se observa que el vehículo gira bruscamente por lo que se pierde estabilidad, para evitar esto se reduce este parámetro.	1 gravities

4.4. Precisión GPS

Para analizar la precisión del módulo de GPS HERE 2 se realiza una prueba conduciendo el vehículo hacia los puntos que se desea registrar como waypoints, de esta manera se elimina el error de las imágenes satelitales. En la Figura 53, se puede observar los puntos marcados.



Figura 53. Puntos de la trayectoria marcados desde posición rover

Una vez marcada la trayectoria se activa el modo auto e inicia la misión. Y al culminar la misión en la Figura 54, se puede apreciar a través de la estación terrena la trayectoria trazada.



Figura 54. Trayectoria de navegación de la precisión GPS.

A continuación, en la Figura 55 y Figura 56 correspondientes al Waypoint 1 y Waypoint 2 respectivamente, se puede apreciar un error de +/-1.15 metros.



Figura 55. Distancia del rover respecto a waypoint 1



Figura 56. Distancia del rover respecto a waypoint 2

Para analizar el error de las imágenes satelitales se procede a registrar los puntos de la trayectoria desde la interfaz de Mission Planner, dando clic sobre el punto que se quiere marcar como waypoint. En la Figura 57, se puede observar los puntos marcados desde la estación terrena utilizando imágenes satelitales.

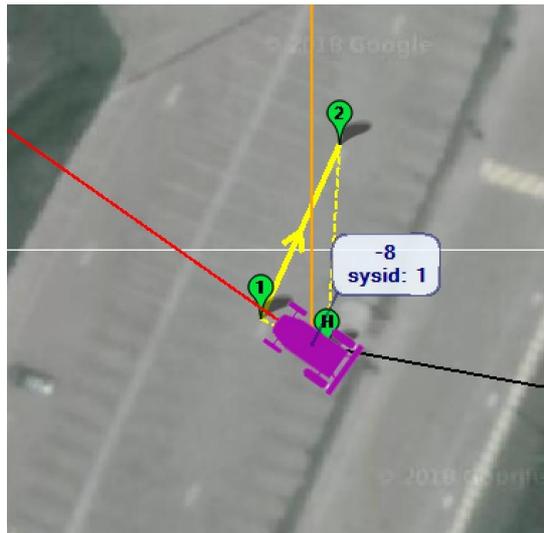


Figura 57. Puntos de la trayectoria marcados desde GUI

Una vez marcada la trayectoria se activa el modo auto e inicia la misión. En la Figura 58, se observa el recorrido que realiza el vehículo.



Figura 58. Trayectoria con precisión de imágenes satelitales

En la Figura 59, se observa la distancia que existe del rover al waypoint 1. Se puede apreciar que la distancia a la que pasa el rover del waypoint 1 marcado en la interfaz es de 1.43 m.

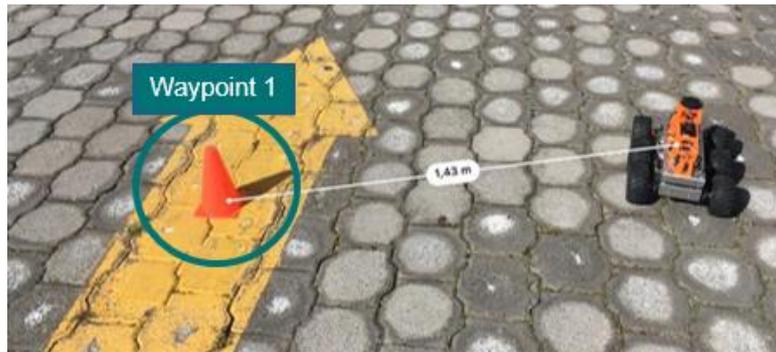


Figura 59. Distancia del rover respecto a waypoint 1

En la Figura 60, se observa la distancia que existe del rover al waypoint 2. Se puede apreciar que la distancia a la que pasa el rover del waypoint 2 marcado en la interfaz es de 2.49 m.

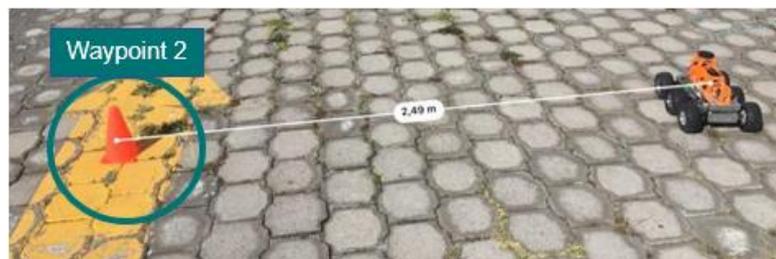


Figura 60. Distancia del rover respecto a waypoint 2

Como se pudo apreciar en los puntos de trayectoria marcados desde la estación terrena se obtuvo un error de +/- 2 metros debido al error en las imágenes satelitales.

4.5. Evasión de obstáculos

En esta sección se realiza misiones de navegación autónoma en el campo de pruebas con obstáculos. El objetivo es analizar y realizar los afinamientos necesarios para un correcto funcionamiento del sistema de evasión de obstáculos implementado.

En la Figura 61, se puede apreciar el campo de pruebas con obstáculos.

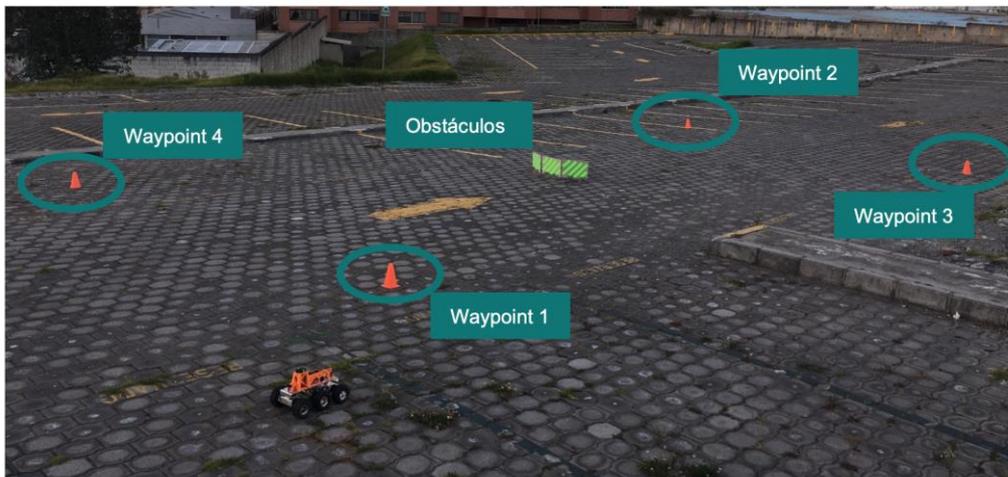


Figura 61. Campo de Pruebas con obstáculos.

Como se puede apreciar en la Figura 62, al ejecutar la misión se obtiene el siguiente trazo de trayectoria, en el recorrido entre el waypoint 1 y waypoint 2 el vehículo detecta el obstáculo y realiza un giro a la derecha sin embargo este no es suficiente, detecta nuevamente el obstáculo y realiza el giro nuevamente. En el recorrido entre el waypoint 3 y waypoint 4 el vehículo detecta el obstáculo y realiza un giro a la derecha, pero debido a que el radio de giro es corto este impacta con el obstáculo.



Figura 62. Prueba Obstáculos 1

A continuación, en la Tabla 43 se especifica los parámetros modificados con el objetivo de corregir el problema mencionado.

Tabla 43
Parámetros para la calibración del sistema de evasión.

Parámetro	Descripción	Valor modificado
RNGFND_TURN_ANGL	Se aumenta el radio de giro en el momento de detectar un obstáculo.	100
RNGFND_TURN_TIME	Tiempo durante el cual el vehículo realiza el giro una vez detecta un obstáculo.	0.5

Como se puede observar en la Figura 63, se realiza nuevamente la misión con los parámetros modificados y se obtiene la siguiente trayectoria, en donde se puede apreciar giros más pronunciados en la detección de obstáculos. En este caso el UGV completo el campo de pruebas exitosamente.

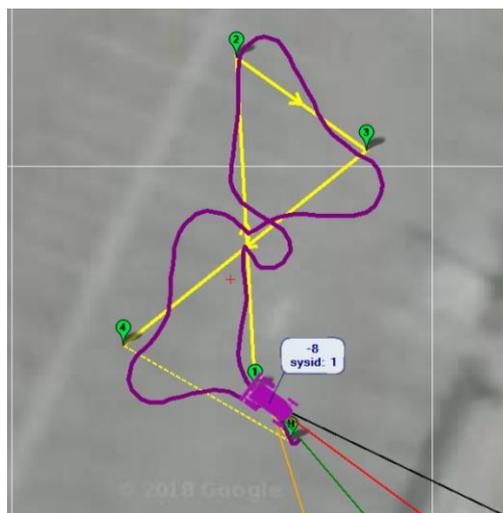


Figura 63. Prueba Obstáculos 2

5. CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones

En la página oficial de Pixhawk mini se menciona un problema de hardware en la IMU9250 (acelerómetro, giroscopio, y magnetómetro), esto no afecta a todos los Pixhawk mini, sin embargo, en el Capítulo 3, Sección 3.1.2 se pudo comprobar que la placa que se analizó estaba afectada. Además, en la página se indica que el software de navegación PX4 deshabilita dicho sensor, por lo que se procedió a deshabilitarlo de igual manera en los parámetros de ArduPilot con el objetivo de lograr una correcta calibración. Luego de las pruebas realizadas se observó lecturas incorrectas lo que ocasionó un funcionamiento erróneo en la navegación autónoma.

El controlador de navegación Pixhawk Mini, tiene un tamaño reducido respecto a su predecesor y cuenta con un número limitado de interfaces, lo que lo hace un controlador de navegación óptimo para UV pequeños, donde no se requiere el uso de varios sensores externos.

El software de navegación PX4 no está orientado para vehículos terrestres, en el asistente de configuración no es posible seleccionar el airframe rover. Esta configuración es posible realizarla accediendo a los parámetros de configuración avanzada es posible establecer el vehículo como rover, sin embargo, esto ocasiona errores en todo su funcionamiento. Por lo mencionado para la implementación del UGV se utilizó el software de navegación ArduPilot, el cual es superior a PX4. ArduPilot es un software que lleva más tiempo en desarrollo por lo que sus algoritmos son más robustos, se compone de un código dedicado para cada tipo de vehículo, y además cuenta con documentación más detallada para desarrollo debido a que tiene una licencia GPL.

Sobre el controlador de navegación Pixhawk Cube se ejecuta el software de navegación ArduRover, el cual es uno de los cuatro tipos de software de ArduPilot. ArduRover está optimizado para UGV, el software permite realizar misiones como el seguimiento de puntos de una trayectoria fijada por

coordenadas geográficas, a través de una estación terrena conectada inalámbricamente.

En las pruebas realizadas con el controlador de navegación Pixhawk Cube se pudo observar que existe un error de aproximadamente 1.15 metros con respecto a los puntos que el vehículo registro en el seguimiento de la trayectoria indicada.

Luego del análisis realizado en base a los sistemas de radio comunicación se seleccionó la tecnología Zigbee debido a que presentaba mejores características según sus especificaciones. Sin embargo, luego de las pruebas realizadas se pudo apreciar inconvenientes debido a que utiliza la misma frecuencia de operación que el wifi de 2.4 GHz, esto produce interferencias en el sistema de telemetría ocasionando una baja velocidad de transmisión.

5.2. Recomendaciones

Como se mencionó anteriormente el controlador de navegación Pixhawk Mini no es el controlador más adecuado en el desarrollo de aplicaciones, además que presenta fallas de hardware. Para el desarrollo de aplicaciones en vehículos terrestres se recomienda el uso de otros controladores de navegación, que presentan mejores características en diseño y funcionamiento como el Pixhawk 2.1 desarrollado por HEX Technology y el Pixhawk 4 desarrollado por Holybro.

Para la selección de los diferentes sensores y componentes del vehículo es recomendable tener en cuenta el tipo de interfaces disponibles en el controlador de navegación. Con el objetivo de no utilizar componentes intermedios en la comunicación.

Es recomendable el uso de más de dos o más sensores inerciales, de esta manera se crea una redundancia en el sistema, se mejora la confiabilidad de la lectura de datos y esto garantiza una mejor estabilidad del vehículo.

Luego de los inconvenientes presentados con la tecnología Zigbee. Es recomendable utilizar sistemas de telemetría que trabajan a una frecuencia de

900 MHz o 400 MHz ya que estos brindan un gran alcance y similares velocidades de transmisión sin interferencia.

REFERENCIAS

- 3D Robotics. (2014). *PPM Encoder User Manual*. Recuperado el 26 de noviembre de 2018 de http://download.ardupilot.org/downloads/wiki/pdf_guides/PPM-Encoder-V3-Manual.pdf
- Alke. (2016). UGV – Vehículos autónomos controlados en forma remota. Recuperado el 26 de mayo de 2018 de <https://www.alke.eu/es/ugv-vehiculos-autonomos>
- Ardobot Robótica. (2018). Bateria *Multistar High Capacity* 4000mAh 3S 10C. Recuperado el 20 de noviembre de 2018 de <https://www.ardobot.co/bateria-11-1v-multistar-high-capacity-4000mah-3s-10c-multi-rotor-lipo-pack.html>
- Arduino. (2018). Arduino Nano. Recuperado el 25 de noviembre de 2018 de <https://store.arduino.cc/usa/arduino-nano>
- ArduPilot. (2016). *ArduPilot :: About*. Recuperado el 13 de septiembre de 2018 de <http://ardupilot.org/about>
- ArduPilot Dev Team. (2018). *Accelerometer Calibration in Mission Planner — Rover documentation*. Recuperado el 10 de diciembre de 2018 de <http://ardupilot.org/rover/docs/common-accelerometer-calibration.html>
- ArduPilot Dev Team. (2018). *Compass Calibration — Rover documentation*. Recuperado el 10 de diciembre de 2018 de <http://ardupilot.org/rover/docs/common-compass-calibration-in-mission-planner.html>
- ArduPilot Dev Team. (2018). *Choosing a Ground Station — Rover documentation*. Recuperado el 16 de septiembre de 2018 de <http://ardupilot.org/rover/docs/common-choosing-a-ground-station.html>

ArduPilot Dev Team. (2018). *Pixhawk Overview — Rover documentation*. Recuperado el 10 de septiembre de 2018 de <http://ardupilot.org/rover/docs/common-pixhawk-overview.html#common-pixhawk-overview>

ArduPilot Dev Team. (2018). *Telemetry — Rover documentation*. Recuperado el 15 de septiembre de 2018 de <http://ardupilot.org/rover/docs/common-telemetry-landingpage.html>

ArduPilot Dev Team. (2018). *Compatible RC Transmitter and Receiver Systems — Copter documentation*. Recuperado el 15 de octubre de 2018 de <http://ardupilot.org/copter/docs/common-pixhawk-and-px4-compatible-rc-transmitter-and-receiver-systems.html>

ArduPilot Dev Team. (2018). *Downloading the Code / Using Git — Dev documentation*. Recuperado el 28 de octubre de 2018 de <http://ardupilot.org/dev/docs/where-to-get-the-code.html>

ArduPilot Dev Team. (2018). *Learning ArduPilot — Introduction — Dev documentation*. Recuperado el 4 de noviembre de 2018 de <http://ardupilot.org/dev/docs/learning-ardupilot-introduction.html>

ArduPilot Dev Team. (2018). *Learning the ArduPilot Codebase — Dev documentation*. Recuperado el 4 de noviembre de 2018 de <http://ardupilot.org/dev/docs/learning-the-ardupilot-codebase.html>

ArduPilot Dev Team. (2018). *Radio Control Calibration in Mission Planner — Rover documentation*. Recuperado el 20 de diciembre de 2018 de <http://ardupilot.org/rover/docs/common-radio-control-calibration.html>

ArduPilot Dev Team. (2018). *Setting up the Build Environment (Linux/Ubuntu) — Dev documentation*. Recuperado el 13 de septiembre de 2018 de <http://ardupilot.org/dev/docs/building-setup-linux.html#building-setup-linux>

ArduPilot Dev Team. (2018). *Community: — ArduPilot documentation*.

Recuperado el 3 de diciembre de 2018 de <http://ardupilot.org/ardupilot/>

ArduPilot Dev Team. (2019). *Complete Parameter List — Rover documentation*. Recuperado el 31 de enero de 2019 de <http://ardupilot.org/rover/docs/parameters.html>

ArduPilot Dev Team. (2019). *Object Avoidance — Rover documentation*. Recuperado el 4 de enero de 2019 de <http://ardupilot.org/rover/docs/rover-object-avoidance.html>

Arrow Electronics. (2018). Bateria primaria y secundaria. Recuperado el 17 de octubre de 2018 de <https://www.arrow.com/es-mx/categories/batteries-and-accessories/batteries>

Benewake (Beijing). (2017). *Official Product Information LiDAR*. Recuperado el 3 de diciembre de 2018 de <http://www.benewake.com/en/down.html>

Benewake (Beijing). (2017). *TFmini LiDAR Module*. Recuperado el 3 de diciembre de 2018 de <http://en.benewake.com/download>

Bloom, J. (2015). *Serial Communication*. Recuperado el 11 de septiembre de 2018 de <https://learn.sparkfun.com/tutorials/serial-communication>

Bunge, J. (2016). *Farmers Reap New Tools From Their Own High-Tech Tinkering - WSJ. The Wall Street Journal*. Recuperado el 22 de abril de 2018 de <https://www.wsj.com/articles/farmers-reap-new-tools-from-high-tech-tinkering-1461004688?shareToken=stfba6c579c48849e98e9e165da2129bf0>

Corona Ramírez, L., Abarca Jiménez, G., y Mares Carreño, J. (2014). Sensores y actuadores aplicaciones con Arduino. Recuperado el 10 de octubre de 2018 de <https://ebookcentral.proquest.com/lib/udlasp/reader.action?docID=4569609&query=actuadores>

De Marchi, L., Mackay, R., Tridgell, A., y De Sousa, G. (2018). *Building ArduPilot*. Recuperado el 6 de noviembre de 2018 de <https://github.com/ArduPilot/ardupilot/blob/master/BUILD.md>

Delgado, V. (2016). Historia de los drones - El Drone. Recuperado el 22 de abril de 2018 de <http://eldrone.es/historia-de-los-drones/>

Digi International. (2018). *XBee/XBee-PRO S2C 802.15.4 Module User Guide*. Recuperado el 12 de diciembre de 2018 de <https://www.digi.com/resources/documentation/digidocs/pdfs/90001500.pdf>

Encyclopædia Britannica. (2018). *Battery | electronics*. Recuperado el 17 de octubre de 2018 de <https://www.britannica.com/technology/battery-electronics>

Encyclopædia Britannica. (2018). *Voltage regulator*. Recuperado el 14 de noviembre de 2018 de <https://www.britannica.com/technology/voltage-regulator>

Guangzhou HC Information Technology. (2011). *HC Serial Bluetooth Products User Instructional Manual*. Recuperado el 12 de diciembre de 2018 de https://cdn.makezine.com/uploads/2014/03/hc_hc-05-user-instructions-bluetooth.pdf

Hex Technology Limited. (2016). *Pixhawk v2 Feature Overview*. Recuperado el 25 de enero de 2019 de http://www.hex.aero/wp-content/uploads/2016/07/DRS_Pixhawk-2-17th-march-2016.pdf

Hord, M. (2016). I2C. Recuperado el 11 de septiembre de 2018 de <https://learn.sparkfun.com/tutorials/i2c>

Horizon Hobby. (2004). *Introducing the New DSM System from Spektrum*. Recuperado el 7 de noviembre de 2018 de <http://www.spektrumrc.com/Articles/Article.aspx?ArticleID=1423&Page=1>

- Huang, H. (2009). *The HCS12/9S12: An Introduction to Software and Hardware Interfacing*. Recuperado el 25 de noviembre de 2018 de https://books.google.com.ec/books?id=5atwJG7D_HMC&pg=PR1&hl=es&source=gbs_selected_pages&cad=2#v=onepage&q&f=false
- Meier, L. (2018). *The History of Pixhawk – Auterion*. Recuperado el 26 de mayo de 2018 de <https://auterion.com/the-history-of-pixhawk/>
- Cheung, P. (2018). *Servo Motor SG90*. Recuperado el 30 de enero de 2019 de http://www.ee.ic.ac.uk/pcheung/teaching/DE1_EE/stores/sg90_datasheet.pdf
- Pololu. (2017). *Pololu - Dagu Wild Thumper 6WD All-Terrain Chassis, Silver, 75:1*. Recuperado el 7 de octubre de 2018 de <https://www.pololu.com/product/1561>
- Pololu. (2018). *Pololu - Motion Control Modules*. Recuperado el 10 de octubre de 2018 de <https://www.pololu.com/category/9/motion-control-modules>
- Pololu Corporation. (2015). *Pololu TReX User's Guide*. Recuperado el 28 de noviembre de 2018 de <https://www.pololu.com/product/777/resources>
- ProfiCNC. (2017). *Here 2 - Pixhawk2*. Recuperado el 30 de enero de 2019 de <http://www.proficnc.com/all-products/152-gps-module.html>
- ProfiCNC. (2017). *Pixhawk2.1 - Pixhawk2*. Recuperado el 25 de enero de 2019, de <http://www.proficnc.com/content/13-pixhawk2>
- ProfiCNC. (2017). *Pixhawk2.1 Standard Set - Pixhawk2*. Recuperado el 25 de enero de 2019 de <http://www.proficnc.com/all-products/31-pixhawk2-suite.html>
- ProfiCNC. (2017). *Power Brick Mini - Pixhawk2*. Recuperado el 26 de enero de 2019 de <http://www.proficnc.com/all-products/80-power-module.html>
- PX4 Dev Team. (2018). *User Guide - PX4 Open Source Autopilot*. Recuperado

el 2 de diciembre de 2018 de <https://px4.io/user-guide-2/>

PX4 Dev Team. (2017). *Pixhawk Mini · PX4 User Guide*. Recuperado el 5 de septiembre de 2019, de https://docs.px4.io/en/flight_controller/pixhawk_mini.html

PX4 Dev Team. (2018). *Basic Concepts · PX4 User Guide*. Recuperado el 26 de mayo de 2018 de https://docs.px4.io/en/getting_started/px4_basic_concepts.html

PX4 Dev Team. (2018). *Flight Controller Reference Design · PX4 Developer Guide*. Recuperado el 4 de noviembre de 2018 de <https://dev.px4.io/en/debug/reference-design.html>

PX4 Dev Team. (2018). *Radio Systems · PX4 User Guide*. Recuperado el 22 de septiembre de 2018 de https://docs.px4.io/en/getting_started/rc_transmitter_receiver.html

PX4 Dev Team. (2018). *Telemetry Radios · PX4 User Guide*. Recuperado el 15 de septiembre de 2018 de <https://docs.px4.io/en/telemetry/>

PX4 Dev Team. (2018). *Toolchain Installation · PX4 Developer Guide*. Recuperado el 13 de septiembre de 2018 de http://dev.px4.io/en/setup/dev_env.html

PX4 Dev Team. (2019). *Cube Wiring Quickstart · PX4 User Guide*. Recuperado el 26 de enero de 2019 de https://docs.px4.io/en/assembly/quick_start_cube.html

QGroundControl. (2019). *Firmware · QGroundControl User Guide*. Recuperado el 10 de diciembre de 2018 de <https://docs.qgroundcontrol.com/en/SetupView/Firmware.html#connect-device-for-firmware-update>
<https://docs.qgroundcontrol.com/en/SetupView/Firmware.html%23connect-device-for-firmware-update>

QGroundControl. (2019). *Setup · QGroundControl User Guide*. Recuperado el 10 de diciembre de 2018 de <https://docs.qgroundcontrol.com/en/SetupView/SetupView.html>

Rodriguez, M., Trejo, F., y Espinoza, F. (2017). Instrumentación y desarrollo de una plataforma de vehículo terrestre para la obtención de posicionamiento RTK. *Revista del Diseño Innovativo*. Recuperado el 26 de mayo de 2018 de http://www.ecorfan.org/taiwan/research_journals/Diseno_Innovativo/vol1num1/Revista_Dise%C3%B1o_Innovativo_V1_N1_4.pdf

Slamtec. (2018). *RPLIDAR A1*. Recuperado el 2 de diciembre de 2018 de http://bucket.download.slamtec.com/b90ae0a89feba3756bc5aaa0654c296dc76ba3ff/LD108_SLAMTEC_rplidar_datasheet_A1M8_v2.2_en.pdf

Shenzhen Anxinke Technology. (2015). *ESP-01 WiFi Module*. Recuperado el 12 de diciembre de 2018 de <https://ecksteinimg.de/Datasheet/Ai-thinker%20ESP-01%20EN.pdf>

Spektrum & Horizon Hobby. (2005). *DSM2 Technology : Spektrum*. Recuperado el 7 de noviembre de 2018 de <http://www.spektrumrc.com/Technology/DSM2.aspx>

Spektrum & Horizon Hobby. (2005). *DSMX Technology: Spektrum*. Recuperado el 7 de noviembre de 2018 de <https://www.spektrumrc.com/Technology/DSMX.aspx>

Spektrum & Horizon Hobby. (2010). *AR6210 Manual*. Recuperado el 4 de noviembre de 2018 de <https://www.spektrumrc.com/Products/Default.aspx?ProdID=SPMAR6210>

Spektrum & Horizon Hobby. (2011). *Spektrum DX5e*. Recuperado el 15 de noviembre de 2018 de

https://www.spektrumrc.com/ProdInfo/Files/SPMR5510-Manual_EN.pdf

Welten Holdings. (2016). *28BYJ-48 Stepper Motor*. Recuperado el 5 de enero de 2019 de <http://robocraft.ru/files/datasheet/28BYJ-48.pdf>

Zurdo, D. (2005). El sistema GPS y sus competidores. Recuperado el 2 de octubre de 2018 de https://www.acta.es/medios/articulos/ciencias_y_tecnologia/038055.pdf

