



**UNIVERSIDAD POLITÉCNICA SALESIANA**  
**SEDE GUAYAQUIL**  
**CARRERA DE INGENIERÍA ELECTRÓNICA**

**DISEÑO DE UN PROTOTIPO DE ROBOT CON  
NAVEGACIÓN AUTOMÁTICA GPS PARA INSPECCIÓN DE  
RUTAS CON VISIÓN ARTIFICIAL**

Trabajo de titulación previo a la obtención del  
Título de Ingeniero Electrónico

**AUTORES:**

Katherine Nahomy Remache Luque

Héctor Alfonso Matovelle Rentería

**TUTOR:**

Ing. Orlando Barcia

Guayaquil - Ecuador

2021

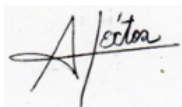
**CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE  
TITULACIÓN**

Nosotros, **KATHERINE NAHOMY REMACHE LUQUE** con documento de identificación N° 0921886891 y **HÉCTOR ALFONSO MATOVELLE RENTERÍA** con documento de identificación N° 0929565034; manifestamos que:

Somos los autores y responsables del presente trabajo; y, autorizamos a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Guayaquil, 9 de febrero del año 2022

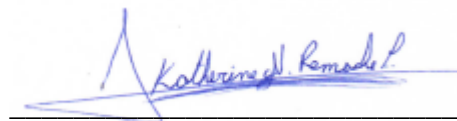
Atentamente,



---

Héctor Alfonso Matovelle Rentería

0929565034



---

Katherine Nahomy Remache Luque

0921886891

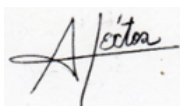
**CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITECNICA SALESIANA**

Nosotros, **KATHERINE NAHOMY REMACHE LUQUE** con documento de identificación No. 0921886891 y **HÉCTOR ALFONSO MATOVELLE RENTERÍA**, con documento de identificación N° 0929565034, expresamos nuestra voluntad y por medio del presente documento cedemos a la **UNIVERSIDAD POLITECNICA SALESIANA** la titularidad sobre los derechos patrimoniales en virtud de que somos autores del trabajo de grado titulado: “**DISEÑO DE UN PROTOTIPO DE ROBOT CON NAVEGACIÓN AUTOMÁTICA GPS PARA INSPECCIÓN DE RUTAS CON VISIÓN ARTIFICIAL**”, el cual ha sido desarrollado para optar por el título de: **INGENIERO ELECTRÓNICO**, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribimos este documento en el momento que hacemos la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Guayaquil, 9 de febrero del año 2022

Atentamente,



---

Héctor Alfonso Matovelle Rentería

0929565034



---

Katherine Nahomy Remache Luque

0921886891

## CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, ORLANDO GIOVANNI BARCIA AYALA con documento de identificación N° 1309445714, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: “**DISEÑO DE UN PROTOTIPO DE ROBOT CON NAVEGACIÓN AUTOMÁTICA GPS PARA INSPECCIÓN DE RUTAS CON VISIÓN ARTIFICIAL**”, realizado **REMACHE LUQUE KATHERINE NAHOMY** con documento de identificación N° 0921886891 y **HÉCTOR ALFONSO MATOVELLE RENTERÍA** con documento de identificación N° 0929565034, obteniendo como resultado final el trabajo de titulación bajo la opción trabajo de grado que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Guayaquil, 9 de febrero del año 2022

Atentamente,



---

Ing. Orlando Giovanni Barcia Ayala, MSc

1309445714

## **DEDICATORIA**

Este proyecto se lo dedico a tres personas que han sido mi apoyo, impulso y motor para culminar mi carrera y ser una profesional, a mi madre **LUISA KATINA LUQUE AGUILERA** por su amor infinito, por ser incondicional y pese a la distancia me demuestra día a día que no hay barreras ni distancia para limitar el amor de madre; a mi abuelo **LUIS EDUARDO LUQUE BRIONES** por ser el mejor ejemplo de padre y amigo que pese a ya no tenerlo a mi lado cumplo uno de nuestros sueños, a mi hijo **DANIEL MATHIAS BOBADILLA REMACHE** porque esta es una prueba para demostrarle que no hay impedimento alguno para no cumplir tus sueños y ser ejemplo de lucha, perseverancia y superación.

**Katherine Nahomy Remache Luque**

## **DEDICATORIA**

Dedico esta tesis a DIOS, por permitirme llegar hasta donde me encuentro, por darme fuerza y salud para con esto llevar a cabo mis metas y objetivos.

A mi familia en especial a mis padres. A mi madre **LAURA ARACELY RENTERÍA MORAN** por su infinito amor y paciencia conmigo, a mi padre **HÉCTOR JOSÉ MATOVELLE VILLAMAR** por su sacrificio y entrega hacia mí, siempre preocupado de que no me faltase nada.

**Héctor Alfonso Matovelle Rentería**

## **AGRADECIMIENTO**

Quiero agradecer a Dios por darme salud, fortaleza y forjar mi camino.

A mi entrañable familia por el apoyo constante, por sus palabras de aliento, por creer en mí y en mis capacidades de seguir adelante pese a toda adversidad porque todo se puede lograr con sacrificio, amor, dedicación y esfuerzo.

A mi papá **RICARDO GONZÁLEZ CORONEL** por siempre apoyarme, creer en mí y ser un ejemplo a seguir; a mi novio **CARLOS IVÁN MENENDEZ CHÁVEZ** y a toda su familia por todo el cariño, la atención y la confianza depositada en mí de ser una gran profesional en la vida.

**Katherine Nahomy Remache Luque**

## **AGRADECIMIENTO**

Comienzo por agradecerle a Dios, padre celestial, por guiar siempre mi camino y por forjar mi espíritu, por estar conmigo y darme las fuerzas para seguir adelante.

A mis padres por su infinito cariño y amor sin medida hacia mí, por siempre velar mis pasos y cuidarme. A mi madre **LAURA ARACELY RENTERÍA MORÁN** que con su inmenso corazón siempre está a mí lado para escucharme y, a mi padre **HÉCTOR JOSÉ MATOVELLE VILLAMAR** por sus innumerables enseñanzas y preocupación. Me encontraré siempre agradecido porque gracias a ellos he encontrado la voluntad y las fuerzas para seguir el camino correcto, a esforzarme, a luchar por mis sueños, a ver la vida de otra manera y que pese a las dificultades puedo siempre contar con ellos, siempre han sido mi pilar fundamental y les debo todo en la vida.

A mis hermanos y familia, por siempre ser un apoyo a seguir fuertemente trabajando a alcanzar cualquier meta que me plantee.

**Héctor Alfonso Matovelle Rentería**



## RESUMEN

Ecuador, es un país lleno de bondades por sus características geológicas, topográficas y climáticas, pero también es un país donde no estamos exentos a la manifestación de eventos que puedan poner en riesgo la vida de nuestra población. [1]

En la actualidad convivimos con eventos que pueden desencadenar desastres naturales, el cual nos dificulta el acceso a lugares y terrenos poco estables en donde una persona no puede ingresar o nos enfrentamos a situaciones en donde tenemos la necesidad de plantear ciertas rutas y explorar en ellas; es por ello que se realizó un Robot terrestre que cuente con navegación automática vía GPS y visión artificial.

Para el desarrollo y cumplimiento de este proyecto se levantó la construcción del robot todoterreno, la adquisición de la placa de programación Pixhawk o APM y la instalación de la Raspberry Pi y visión Artificial para que el usuario a través de la cámara pueda observar en tiempo real en su dispositivo la ruta que está desarrollando en el momento el prototipo.

El usuario también tiene la opción de manejar tanto manual como automático, siendo este último el ingreso de rutas por medio del Mission Planner, esto se logró al comunicar el sistema de inspección con la estación base donde se pueda receptor la información de ubicaciones más vulnerables, posibles misiones de rescate u objetivos específicos que se desee visualizar o explorar.

El robot notificó la ubicación en tiempo real sobre posibles eventos de rescate, problemas en el sitio y donde el ser humano pudo evidenciar posibles riesgos o peligros.

Con ello también se buscó incentivar a compañeros en conocer más de la carrera y se incentivó a que se brinde este tipo de aportes que aún se encuentran en ideas para hacer

un llamado a la iniciativa de inversión y que se comience a apostar por productos tecnológicos propios de la Universidad Politécnica Salesiana. [2]

## **ABSTRACT**

Ecuador is a country full of goodness due to its geological, topographic, climate and weather characteristics, but it is also a country which it is not exempt from circumstances that may endanger the lives of its population.

Nowadays we are faced with events, that can trigger natural disasters, making it difficult for anybody to access unstable places and terrain. Therefore, there will be situations where we have the need create alternative routes and explore them. That is the reason why a terrestrial Robot with an automatic navigation system via GPS and artificial vision was created.

For the development and fulfilment of this project was necessary the assembly of the all-terrain robot, the acquisition of the Pixhawk or APM programming board , the installation of the Raspberry Pi and Artificial vision, to be able to observe -in real time through the camera on the device -the route that the prototype is developing at the moment.

The user has also the option to control the robot either manually or automatically, the latter being the entry of routes through the Mission Planner . This was achieved by communicating the inspection system with the base station, where information from more vulnerable locations, possible rescue missions or specific targets which one wants to view or explore can be received.

The robot notifies the location in real time about possible rescue events, problems on site and wherever the human being could evidence possible risks or dangers.

This proposal also sought to encourage fellow students to learn more about the career and inspire them to provide this type of contributions, in order to make a call to the

investment initiative to bet on Universidad Politécnica Salesiana's technological products.

## ÍNDICE GENERAL

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITECNICA SALESIANA .....	II
CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN .....	III
DEDICATORIA .....	IV
DEDICATORIA .....	V
AGRADECIMIENTO .....	VI
AGRADECIMIENTO .....	VII
RESUMEN.....	VIII
ABSTRACT.....	X
INTRODUCCIÓN .....	1
PLANTEAMIENTO DEL PROBLEMA .....	3
1.2. ANTECEDENTES.....	3
1.3. IMPORTANCIA Y ALCANCES .....	4
1.4. DELIMITACIÓN.....	4
1.4.1. Delimitación Temporal: .....	4
1.4.2. Delimitación Espacial: .....	4
1.4.3. Delimitación Académica: .....	5
1.5. EXPLICACIÓN DEL PROBLEMA.....	5
1.6. OBJETIVOS .....	6
1.6.1. Objetivo General: .....	6
1.6.2. Objetivo Específico: .....	6
ESTADO DEL ARTE.....	7
2.1. ESTADO DEL ARTE.....	7

2.1.1.	Chasis DJI F450 .....	7
2.1.2.	Motores DC 108RPM engranaje alto torque 3-18 VDC .....	7
2.1.3.	Ruedas .....	8
2.1.4.	Monster shield 30 Amp VNH3ASP30 .....	9
2.1.5.	Shield de expansión Arduino nano.....	10
2.1.6.	Arduino nano.....	11
2.1.7.	Sensores ultrasónicos .....	12
2.1.8.	Placa pixhawk .....	12
2.1.9.	Buzzer y pulsador.....	14
2.1.10.	Módulo GPS.....	16
2.1.11.	FPV radio telemetría .....	16
2.1.12.	Módulo de poder APM 2.8 2.6 BEC 5V 3A para pixhawk.....	16
2.1.13.	Tarjeta de expansión de puerto pixhawk I2C.....	17
2.1.14.	Codificador PPM.....	18
2.1.15.	Raspberry PI.....	18
2.1.16.	Cámara de video conferencia GENIUS 1000X HD/720P/USB.....	19
2.1.17.	LiPo battery .....	20
2.1.18.	Cargador IMAX .....	21
2.1.19.	Cargador Turnigy 12v mini charger 2S/3S para batería LiPo.....	22
2.1.20.	Cables.....	23
2.1.21.	Conectores.....	23
2.1.22.	Switch de 9 pines.....	24
2.1.23.	6+ channel RC transmitter and receiver.....	25
2.1.24.	Protocolo MAVLink .....	27

2.1.25.	Paquetes MAVLink.....	27
2.2.	SOFTWARES UTILITARIOS .....	29
2.2.1.	SITL .....	29
2.2.2.	MISSION PLANNER .....	29
	MARCO METODOLÓGICO.....	31
3.1.	FASE INICIAL .....	34
3.1.1.	Estructura y enlaces.....	34
3.1.2.	Enlaces para la navegación.....	36
3.2.	CONFIGURACIÓN.....	37
3.2.1.	Controlador PIXHAWK.....	37
3.3.	ACTUACIÓN .....	53
3.4.	NAVEGACIÓN .....	55
3.4.1.	Navegación Manual.....	55
3.4.2.	Navegación autónoma .....	59
3.5.	VISIÓN ARTIFICIAL .....	63
3.5.1.	Vision.py .....	65
3.5.2.	gpsfinal.py .....	65
3.5.3.	GMAP.py .....	66
3.6.	LIBRERÍAS:.....	66
3.7.	CREACIÓN DE CLAVE API KEY .....	68
3.8.	COMANDOS E INSTRUCCIONES .....	73
	VERIFICACIÓN Y FUNCIONAMIENTO .....	76
4.1.	RESULTADOS.....	76
4.2.	ANÁLISIS DEL RESULTADO .....	79

4.3. CONCLUSIONES .....	84
4.4. RECOMENDACIONES .....	86
REFERENCIAS BIBLIOGRÁFICAS.....	87
ANEXOS A: CÓDIGOS DE PROGRAMACIÓN .....	90
ANEXOS B: DATASHEET DE EQUIPOS .....	98
ANEXOS C: IMÁGENES .....	130



## ÍNDICE DE FIGURAS

Fig. 1 Estructura del chasis de metal .....	7
Fig. 2 Motor DC de alto torque.....	8
Fig. 3 Rueda puntada de nylon .....	9
Fig. 4 Módulo de Monster Shield .....	10
Fig. 5 Conexión del Buzzer .....	15
Fig. 6 Módulo de poder APM.....	17
Fig. 7 Tarjeta de expansión de puerto.....	18
Fig. 8 Cámara Web Genius 720p.....	20
Fig. 9 Cargador IMAX.....	22
Fig. 10 Cargador Turnigy 12v .....	23
Fig. 11 Conector XT60 .....	24
Fig. 12 Switch de 9 pines.....	24
Fig. 13 Estructura de un paquete MAVLink.....	27
Fig. 14 Esquema del proyecto.....	31
Fig. 15 Diagrama de flujo principal del funcionamiento del robot .....	33
Fig. 16 Imagen de la estructura externa del robot.....	35
Fig. 17 Elementos del proyecto .....	36
Fig. 18 Enlaces para la navegación.....	37
Fig. 19 Placa controladora Pixhawk .....	37
Fig. 20 Pantalla del ArduPilot.....	38
Fig. 21 Pantalla de calibración del acelerómetro .....	39
Fig. 22 Prototipo colocado al nivel del piso .....	40
Fig. 23 Prototipo colocado al lado izquierdo.....	40
Fig. 24 Prototipo colocado al lado derecho .....	41

Fig. 25 Prototipo colocado nariz arriba .....	41
Fig. 26 Prototipo colocado nariz abajo .....	42
Fig. 27 Prototipo colocado cabeza abajo .....	42
Fig. 28 Pantalla de calibración del acelerómetro .....	43
Fig. 29 Pantalla de configuración de la brújula .....	44
Fig. 30 Pantalla de calibración de la brújula.....	45
Fig. 31 Imagen de los rangos que tienen los controles .....	46
Fig. 32 Numeración de los Canales según su rango mínimo y máximo.....	47
Fig. 33 Pantalla de configuración del Monitor de batería.....	47
Fig. 34 Pantalla de configuración de las afinaciones básicas. ....	48
Fig. 35 Pantalla de configuración de parámetros estándar .....	49
Fig. 36 Pantalla de configuración de todos los parámetros. ....	50
Fig. 37 Sección de la configuración RC .....	51
Fig. 38 Sección de la configuración Waitpoint.....	51
Fig. 39 Sección de la configuración del telémetro.....	53
Fig. 40 Placa Arduino UNO .....	53
Fig. 41 Grados de distancia de detección del robot .....	54
Fig. 42 Diagrama de flujo del funcionamiento del geoposicionamiento para la realizar la misión .....	56
Fig. 43 Imagen del Radio Control RC .....	58
Fig. 44 Pines posteriores de la placa pixhawk .....	59
Fig. 45 Diagrama de flujo del funcionamiento del control automático .....	60
Fig. 46 Pantalla del Mission Planner .....	61
Fig. 47 Pantalla del Mission Planner .....	62
Fig. 48 Mosaico de Bayer .....	63

Fig. 49 Diagrama de flujo del funcionamiento de la visión artificial .....	65
Fig. 50 Selección de proyecto dentro de la página de Google.....	69
Fig. 51 Pantalla del Google Cloud Platform.....	69
Fig. 52 Bibliotecas de API de Google .....	70
Fig. 53 Bibliotecas de API de Google .....	71
Fig. 54 Instalado producto de la biblioteca de Google .....	71
Fig. 55 Sección de credenciales de API.....	71
Fig. 56 Código de la credencial .....	72
Fig. 57 Clave API .....	72
Fig. 58 Puesta de la clave en el código Python.....	73
Fig. 59 Ventana de GUI para generar mapa .....	73
Fig. 60 Ventana de terminal de comandos.....	74
Fig. 61 Muestra de datos del terminal.....	75
Fig. 62 Ruta recorrida en el bloque B1 de la Universidad Politécnica Salesiana.....	76
Fig. 63 Trayectoria realizada dentro del bloque B de la Universidad Politécnica Salesiana .....	77
Fig. 64 Radio de giro del prototipo.....	77
Fig. 65 Medidas de seguridad ante obstáculos .....	78
Fig. 66 Prueba de los sensores ultrasónicos.....	78
Fig. 67 Puesta en marcha del robot.....	79
Fig. 68 Robot recorriendo camino de piedra con desniveles .....	80
Fig. 69 Código de visión en lenguaje Python .....	81
Fig. 70 Detección de señalética de Salida de Emergencia.....	81
Fig. 71 Terminal de comando de la raspberry pi .....	82
Fig. 72 Puesta en marcha del robot en modo automático .....	83

Fig. 73 Montaje de la parte inferior del prototipo, parte interna. ....	130
Fig. 74 Montaje de la parte superior del prototipo, nivel 1. ....	130
Fig. 75 Ensamble y montaje de la parte externa del prototipo. ....	131
Fig. 76 Planos del diseño en 3D de la cubierta del prototipo .....	131
Fig. 77 Planos del diseño sólido en 3D de la cubierta y nivel 1 interno del prototipo .....	132
Fig. 78 Estructura de los controles del control RC. ....	132
Fig. 79 Diagrama del conector de la placa Pixhawk.....	133
Fig. 80 Conectores de la placa programadora Pixhawk.....	134
Fig. 81 Asignación de pines del puerto utilizado TELEM1 o Telem2 de la placa programadora Pixhawk.....	135
Fig. 82 Asignación de pines del puerto utilizado GPS de la placa programadora Pixhawk .....	135
Fig. 83 Asignación de pines del puerto utilizado I2C de la placa programadora Pixhawk .....	136
Fig. 84 Exposición y validación de proyecto de tesis en la Universidad Politécnica Salesiana a los estudiantes del curso de Control Automático. ....	136
Fig. 85 Encuesta de medición de impacto del proyecto de tesis en la Universidad Politécnica Salesiana a los estudiantes del curso de Control Automático.....	137
Fig. 86 Exposición del proyecto de tesis en webinar desarrollado el martes 17 de agosto por la Universidad Politécnica Salesiana. ....	138

## ÍNDICE DE TABLAS

TABLA I PRATÓN DEL BUZZER SEGÚN CADA ESTADO .....	15
TABLA II TABLA DESCRIPTIVA DE RC .....	25
TABLA III FUNCIÓN DE CADA BYTE DE LA CABECERA .....	28
TABLA IV TABLA DE SENSORES INTEGRADOS - PIXHAWK .....	38

## INTRODUCCIÓN

El ser humano con el paso del tiempo ha ido evolucionando y con ello su entorno, al punto de crear objetos que satisfagan sus necesidades. La tecnología es una de ellas, convirtiéndose en un arma ante los desafíos del mundo actual, un recordatorio de todo lo que somos capaces de lograr.

Tanto hemos evolucionado que ya no necesitamos estar presentes para poder realizar ciertas actividades solo monitorearla constantemente por los cambios que se puedan presentar como en la domótica, pero la robótica es el siguiente paso ya que es adaptable a ciertos posibles cambios que puedan ocurrir. [2]

Podemos decir que un robot es un manipulador reprogramable y multifuncional, que está diseñado para mover objetos, piezas, entre otras que se puede variar las trayectorias o programar para realizar diferentes trabajos; en otras palabras, es una máquina que es capaz de reemplazar a una persona en situaciones específicas o en situaciones de alto riesgo.

A pesar de que la tecnología ha avanzado en una velocidad increíble, no estamos preparados tecnológicamente para enfrentar desafíos que la naturaleza le pueda presentar a nuestro país, debido a que les toma mucho tiempo en realizar trabajos que son necesarias realizarlas en un menos tiempo.

El terremoto del 16 de abril del 2016 que dejó muchas víctimas y marcó la historia de todo un pueblo nos demostró que no estamos preparados para este tipo de eventos, pese a que muchos rescatistas de todo el mundo dieron su mayor esfuerzo durante varios días

para salvar la mayor cantidad de víctimas, pero aun así no fue suficiente ante la magnitud de la desgracia.

Es por ello que la NASA asegura sobre los robots que “No sólo puede viajar a ambientes demasiado lejanos o demasiado hostiles para exploradores humanos, también pueden mejorar el horario de trabajo de una misión espacial tripulada.” [3]

De aquí la importancia de la presente elaboración del Prototipo de Robot con navegación automática GPS para estas inspecciones de rutas que nos permite añadirle la visión artificial ya que trataremos de plantear la mejor solución a través de un equipo electrónico para que de situaciones de riesgo como derrumbes causados por un terremoto puedan ser controladas de manera rápida.

La motivación a solucionar la necesidad de este proyecto será descrita estructuralmente en el Capítulo 1, a través del planteamiento del problema, justificación, técnicas empleadas para el desarrollo y beneficio e impacto que generaría el sistema.

Seguido del Capítulo 2, que describe teóricamente los componentes del sistema y de los lineamientos considerados para el desarrollo de este.

En el Capítulo 3, se detalla el uso y proceso programable tanto de las placas controladoras como del aplicativo para cumplir con nuestro objetivo.

En el Capítulo 4, se detallan teórica y gráficamente las pruebas realizadas con respecto a las propuestas en la zona y en la eventualidad establecidas en el anteproyecto.

## **PLANTEAMIENTO DEL PROBLEMA**

### **1.1. PROBLEMA**

En la actualidad la humanidad está conviviendo con eventos que pueden desencadenar desastres naturales; hace 5 años, el sábado 16 de abril, el terremoto de Pedernales tuvo una magnitud de Mw 7.8 y se convirtió en el evento natural más catastrófico de lo que va del presente siglo en el Ecuador. [4]

Es así, como consecuencias del terremoto se contabilizaron más de 7000 heridos y alrededor de 700 personas fallecidas; luego de varios análisis se concluye que uno de los principales problemas a los que se enfrentaron fue muerte por asfixia siendo un gran porcentaje por la falta de maquinaria, de personal y recursos que el proceso de rescate se hizo muy lento; por lo cual, se realizó un Robot terrestre que cuente con navegación automática vía GPS y visión artificial.

### **1.2. ANTECEDENTES**

En caso de algún temblor o terremoto en donde se encuentren en una situación de rescate y de difícil acceso por escombros o terrenos irregulares donde el ser humano no puede moverse, es importante el contar con una herramienta el cual nos permita buscar, mapear y mandar una señal de alerta u aviso de donde poder buscar una vida de una manera más rápida y segura; sin embargo, al ser un prototipo el presente Proyecto de Titulación busca beneficiar en un inicio a los estudiantes de Ingeniería Electrónica y Automatización relacionados al campo de Automatización Industrial, redes de telecomunicaciones y grupos investigativos para el Club de Robótica.



### **1.3. IMPORTANCIA Y ALCANCES**

El desarrollo del proyecto tiene como fundamento principal solventar la necesidad de recopilar información mediante la configuración de la placa Pixhawk para el escaneo de objetivos y para un posterior estudio de mejoras para implementación en desastres naturales o análisis de estudios académicos. Por ello es indispensable contar con una herramienta el cual realice inspecciones en determinadas zonas de prueba en terrenos irregulares con características de suelos distintas; el robot podrá notificar la ubicación en tiempo real sobre posibles eventos de rescate y problemas en el sitio donde el ser humano podría evidenciar posibles riesgos y peligros. Esto trajo como resultado la factibilidad para el desarrollo de plan de acción y prevenir estos eventos. Así mismo el ahorro de recursos necesarios hasta el momento para realizar la inspección como son la mano de obra y tiempo invertido, generando un costo beneficio muy aceptable.

### **1.4. DELIMITACIÓN**

#### **1.4.1. Delimitación Temporal:**

La investigación y desarrollo de este proyecto se realizó en la ciudad de Guayaquil, Ecuador en el periodo 2020-2021.

#### **1.4.2. Delimitación Espacial:**

El proyecto de titulación se desarrolló para el Club de robótica que pertenece a las carreras de ingeniería de la Universidad Politécnica Salesiana, que se encuentra en el Campus Centenario Robles 104 y Chambers. Debido a la Emergencia sanitaria que el Ecuador se encuentra actualmente, se realizó dentro de un domicilio tomando en cuenta todas las medidas preventivas y precauciones del caso, las pruebas se realizaron en un

terrero abierto para cumplir con las medidas de distanciamiento social y en lugares diferentes donde el suelo tenga características irregulares. En la visión artificial la detección de objetos se realiza mediante la captura o censo de un color verde.

#### **1.4.3. Delimitación Académica:**

A través del proyecto de investigación y desarrollo para la titulación de Ingeniería Electrónica se aplicaron conocimientos técnicos además de empíricos adquiridos durante los cursos regulares y seminarios profesionales dictados en el transcurso de la carrera en las materias como; Electiva, Sensores y transductores, Teoría de control, Procesamiento Digital de Señales (DSP), Programación.

### **1.5.EXPLICACIÓN DEL PROBLEMA**

A pesar de que contamos con la tecnología para ayudarnos a enfrentar situaciones de necesidades o urgencias como es en el caso de los desastres naturales, fue demostrado que hace falta prepararse más y darle más asunto al desarrollo de nuestra tecnología.

El terremoto del 16 de abril del 2016 demostró que no estábamos preparados para enfrentarnos con esta realidad es por ello que es importante contar con un robot que nos permite agilizar y tener esa velocidad de mapeo y escaneo para poder salvar vidas; por ello es indispensable contar con un instrumento o una herramienta que nos permita tener ese acceso de inspección a lugares que para un ser humano no le permita.

El proyecto realizado es una apertura a poder realizar esas inspecciones en lugares poco accesibles o en terrenos irregulares, el cual ya sea mediante una primera inspección por RC podamos verificar ir visualizando mediante la cámara los obstáculos que se nos presenten y enviando los posibles accesos para que pueda ir el ser humano.

## **1.6. OBJETIVOS**

### **1.6.1. Objetivo General:**

Diseñar un robot que realice inspección autónoma en búsqueda de señales de rescate, señales de auxilio, donde puedan concurrir muchas personas en caso de desastres naturales y en terrenos vulnerables, gracias a posicionamiento Global GPS y visión Artificial.

### **1.6.2. Objetivo Específico:**

1. Realizar configuración y puesta en marcha de placa programadora Pixhawk.
2. Construir un robot todo terreno que pueda navegar por terrenos accidentados por desastres naturales tales como rocas, piedras, para inspección de lugares complicados o inaccesibles por el usuario. (El usuario es el operador del Robot, la persona encargada de manejarlo por control remoto y crear las rutas).
3. Programar visión artificial utilizando RaspberryPi OpenCV y Python.
4. Interactuar con el medio físico a través de sensores.
5. Desarrollar rutas de inspección autónomas GPS.

## ESTADO DEL ARTE

### 2.1. ESTADO DEL ARTE

#### 2.1.1. Chasis DJI F450

Estructura compuesta por metal o lata que aporta rigidez como se muestra en *Fig. 1* y forma a los carros con apartados físicos para los elementos o accesorios necesarios en el funcionamiento de un vehículo, permitiendo una buena funcionalidad a los mismos.

Características:

- Agujeros de sensores y motores: 2mm
- Altura: 24cm
- Largo: 10cm
- Ancho: 4cm



Fig. 1 Estructura del chasis de metal

#### 2.1.2. Motores DC 108RPM engranaje alto torque 3-18 VDC

Este elemento posee una caja de metal de 125:1 que permite reducir la velocidad y obtener un gran torque, cuyos valores bordean los 77 RPM (9 V.) y los 108 RPM (12 V),

con 5Kg/cm a los 9 volt. Su voltaje de alimentación es de 3 a 18 Vdc, donde su máximo PAR de trabajo o torque, es a 18 Vdc. Como se muestra en **Fig. 2**.

Características:

- Voltaje de funcionamiento (Gran Margen de voltaje): 3 ~ 18 Vdc
- Diámetro eje: 4mm
- Engranajes planetarios.
- Potencia: 1W
- Relación de engranajes: 1: 125
- Velocidad en vacío: 77RPM (9 V.) / 108RPM (12 V.)
- Torque (máx.): 5 Kg/cm



Fig. 2 Motor DC de alto torque [5]

### 2.1.3. Ruedas

Rueda de neumático de coche encaja 4WD / 2WD de 85 mm de diámetro exterior y 38 mm de ancho cuyo bode del coche es de un material, de nylon, como se muestra en **Fig. 3**. Los neumáticos fabricados de caucho con coeficiente de fricción suave, buena capacidad de soporte y buen agarre adecuado para

condiciones complejas de tráfico, tales como arena, laderas de las montañas y otras ocasiones de Cross-country.



Fig. 3 Rueda puntada de nylon [6]

#### 2.1.4. **Monster shield 30 Amp VNH3ASP30**

Es un módulo con dos controladores de motor de DC con una corriente continua máxima de 14 A por canal. En el escudo hay una entrada para PWM, la capacidad de instalar bloques de terminales de 5 mm y la capacidad de medir la corriente de carga, como se muestra en *Fig. 4*.

Además, el controlador tiene una función de bloqueo del motor. El módulo está hecho en el formato de shield, lo que simplifica su conexión con el Arduino. El módulo también tiene salidas analógicas para medir la corriente, con la ayuda de la cual es posible apagar los motores de emergencia en caso de atasco o demasiada carga mecánica. Hay orificios en el tablero para montar almohadillas de salida de 5 mm (no incluidas). Cuando se usa a corrientes cercanas al máximo, se recomienda soldar los cables al módulo y usar enfriamiento activo.

Especificaciones:

- Basado en el chip VN13ASP30 (
- Número de canales: 2
- Voltaje de entrada: 5.5 ... 16 V
- Tensión lógica: 5 V
- Corriente de carga: hasta 6 A sin enfriamiento, hasta 14 A con enfriamiento
- Máxima corriente de sobretensión hasta 30 A

Se puede visualizar estos datos en el *ANEXOS B: DATASHEET DE EQUIPOS - DATASHEET DEL MONSTER SHIELD 30 AMP VN13ASP30*.



Fig. 4 Módulo de Monster Shield [7]

### 2.1.5. Shield de expansión Arduino nano

Placa que facilita las conexiones del Arduino Nano ya que utiliza la configuración 3P, la cual expande a 3 pines (Señal, 5V y GND) cada pin de entrada/salida del micro controlador, facilitando la conexión de servomotores y sensores.

Características:

- Configuración 3P en pines de VCD, GND y E/S tanto digitales como analógicas.
- Compatible con Arduino Nano v2.x y v3.x.
- Compatibilidad para conexión de Servomotores.
- Pines de expansión I2C.
- Salida AREF
- 14 puertos I/O digitales preparados para conexión directa de módulos digitales o servos.
- 6 puertos analógicos preparados para conexión directa de sensores.
- Conector de alimentación (power jack).
- Botón de reset.
- LED indicador de alimentación.
- Peso: 21 g
- Dimensiones: 5.72 cm x 5.35 cm [8]

#### **2.1.6. Arduino nano**

El Arduino Nano es una placa pequeña, completa y compatible con la placa de pruebas basada en el ATmega328 (Arduino Nano 3.x). Tiene más o menos la misma funcionalidad del Arduino Duemilanove, pero en un paquete diferente. Solo carece de un conector de alimentación de CC y funciona con un cable USB Mini-B en lugar de uno estándar.



Arduino es una plataforma de electrónica "open-source" o de código abierto cuyos principios son contar con software y hardware fáciles de usar. Es decir, una forma sencilla de realizar proyectos interactivos para cualquier persona. [9]

### **2.1.7. Sensores ultrasónicos**

El sensor HC-SR04 es un sensor de distancia que utiliza ultrasonido para determinar la distancia de un objeto en un rango de 2 a 450 cm.

Destaca por su pequeño tamaño, bajo consumo energético, buena precisión y excelente precio.

Los sensores ultrasónicos miden la distancia mediante el uso de ondas ultrasónicas. El cabezal emite una onda ultrasónica y recibe la onda reflejada que retorna desde el objeto. Los sensores ultrasónicos miden la distancia al objeto contando el tiempo entre la emisión y la recepción. [10]

### **2.1.8. Placa pixhawk**

Es una plataforma de tiempo real y multihilo con un entorno de programación Unix/Linux que brinda flexibilidad y confiabilidad para manejar cualquier tipo de vehículo autónomo.

Características:

Procesador

- Núcleo ARM Cortex M4 de 32 bits con FPU
- 168 Mhz / 256 KB de RAM / 2 MB de memoria flash
- Coprocesador a prueba de fallos de 32 bits

## Sensores

- MPU6000 como aceleración principal y girocompás
- Giroscopio ST Micro de 16 bits
- Acelerómetro / brújula ST Micro de 14 bits (magnetómetro)
- Barómetro MEAS

## Energía

- Controlador de diodo ideal con conmutación por error automática
- Servo rail de alta potencia (7 V) y listo para alta corriente
- Todas las salidas periféricas protegidas contra sobre corriente, todas las entradas protegidas contra ESD

## Interfaces

- 5 puerto serie UART, 1 con capacidad de alta potencia, 2 con control de flujo de HW
- Entrada de satélite Spektrum DSM / DSM2 / DSM-X
- Entrada Futaba S.BUS (salida aún no implementada)
- Señal de suma de PPM
- Entrada RSSI (PWM o voltaje)
- I2C, SPI, 2x CAN, USB
- Entradas ADC de 3.3V y 6.6V

## Dimensiones

- Peso 38 g (1,3 oz)

- Ancho 50 mm (2,0 ")
- Altura 15,5 mm (0,6 ")
- Longitud 81,5 mm (3,2 ")

Dentro de las características resalta la inclusión de la IMU en la misma plataforma, el MPU 6000 es un sensor que contiene un giroscopio y un acelerómetro en los 3 ejes. El giroscopio incluido tiene una salida de datos digital de 16 bits.

Otra característica importante son los puertos incluidos ya que tiene una enorme cantidad por lo que cubre el requisito de puertos mínimos, y la ventaja de estos puertos es que detectan automáticamente los módulos y los configura por lo que es totalmente compatible con los módulos detallados anteriormente y además en la misma plataforma y con ayuda del software de la estación se puede realizar la calibración de los sensores externos e internos de la plataforma.

De igual manera, si se desea tener mayor información se puede acceder a la página web de ArduPilot. [11]

#### **2.1.9. Buzzer y pulsador**

Se utiliza un zumbador (o alarma de tono) para indicar de forma audible los cambios de estado del vehículo. Dependiendo de las capacidades de la placa, este puede ser un dispositivo activo o un dispositivo piezoeléctrico pasivo impulsado por PWM capaz de producir tonos variables. La conexión del buzzer la puede observar en *Fig. 5*.



Fig. 5 Conexión del Buzzer

Si se utiliza un zumbador activo, como lo puede indicar la siguiente **TABLA I**:

**TABLA I**  
**PRATÓN DEL BUZZER SEGÚN CADA ESTADO [12]**

ESTADO	PATRÓN
<b>Armamento</b>	Pitido largo de 3 segundos
<b>Fallo de Armado</b>	Bip único
<b>Desarmado</b>	Bip único
<b>Batería a prueba de fallos</b>	Un pitido que se repite cada 3 segundos
<b>Fallo EFK</b>	Bip-Bip-Bip-Bip
<b>Vehículo perdido</b>	Bip-Bip que se repite cada 3 segundos

El pulsador por otra parte, sirve para conocer el estado de la placa mediante indicadores visuales de un LED incorporado que según la frecuencia de destello significa una cosa u otra. Principalmente también funciona para pre armar los motores, ya que no se pueden

armar y poner en marcha sin antes haberlo pulsado durante unos segundos hasta ver la luz roja fija.

#### **2.1.10. Módulo GPS**

En la actualidad las innovaciones de los GPS integrado en los drones realizan las mismas funciones que los GPS en automóviles u otros dispositivos, pero tiene algunos otros elementos que agregan datos de ubicación a la telemetría, incluidos los datos de velocidad.

Es importante que los vehículos reciban toda esta información, desde carros profesionales hasta prototipos de vehículos, sin los cuales el uso de estos dispositivos sería muy limitado ya que para estos prototipos pueden controlarse remotamente el GPS. Regrese al punto de partida, registre ubicaciones específicas y agregue detalles sobre su sitio.

#### **2.1.11. FPV radio telemetría**

Este permite realizar la comunicación mediante el UAV y la estación terrestre, se comunica desde el aire mediante el protocolo MAVLink, nos ayuda a que exista interacción de las misiones en tiempo real y adquirir datos de transmisión por parte de la cámara del UAV o cualquier otro componente instalado en él.

Las velocidades de telemetría varían de 56K a 1-2K baudios según el protocolo y, en algunos casos, la distancia. A menudo, el rango de telemetría será menor que el rango de control de radio.

#### **2.1.12. Módulo de poder APM 2.8 2.6 BEC 5V 3A para pixhawk**

Este módulo de potencia, como se muestra en *Fig. 6* Módulo de poder APM *Fig. 6*, tiene la función de que puede proporcionar una fuente de alimentación estable a la placa de control.

- Voltaje máximo de entrada: 18V
- Voltaje de entrada mínimo: 4,5 V
- Medición de voltaje y corriente configurada para ADC de 5V

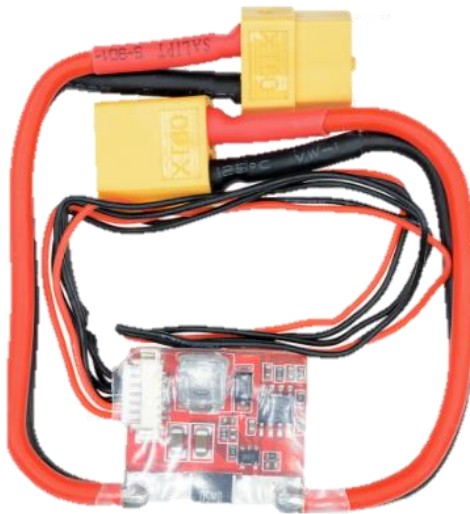


Fig. 6 Módulo de poder APM [13]

### 2.1.13. Tarjeta de expansión de puerto pixhawk I2C

La placa de expansión del puerto Pixhawk I2C con cable divisor I2C expande el puerto I2C para permitir hasta cuatro conexiones periféricas adicionales para Pixhawk.

Tiene cinco zócalos verticales 1.25 4 a completamente paralelos, conexión conveniente en forma de 1 arrastre 4 cuatro periféricos I2C. Como se muestra en la Fig. 7 Tarjeta de expansión de puerto *Fig. 7*. [14]

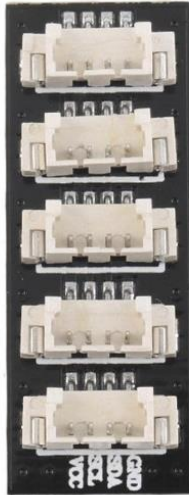


Fig. 7 Tarjeta de expansión de puerto [14]

#### **2.1.14. Codificador PPM**

Lo que hace es agrupar diferentes señales PWM (Pulse Width Modulation) en una sola señal. Una señal PWM es un tren de pulsos con una periodicidad fija, los cuales pueden tener una cierta duración, estando el valor que se desea transmitir codificado en esa duración. Dado que para la actuación se utilizan diversas señales PWM, se pueden combinar todas estas en una sola de PPM, de manera que pueden existir varios canales. Teniendo cada canal una duración diferente en la señal PWM, esto lo que hace es generar un tren de pulsos separados la distancia correspondiente a la duración del PWM de cada canal, generándose así la señal PPM. [15]

#### **2.1.15. Raspberry PI**

La innovación que presenta esta RaspBerry Pi son sus múltiples características, y la principal es que cuenta con las mismas funcionalidades que sus antecesoras como lo es la RaspBerry Pi original. La potencia que genera esta placa es producida por su procesador

Broadcom BCM2835 con un núcleo ARM11 de 1GHz, lo cual se encuentra situado en una placa de circuito de 65x30 milímetros.

Esta placa tan solo cuenta con 512MB de memoria RAM, pero el sistema operativo se ejecuta desde una tarjeta de memoria del tipo microSD, lo que le aporta practicidad, velocidad, sigilo y eficiencia.

Esta RaspBerry viene equipada con un puerto Mini-HDMI de 1080p para realizar la salida de video, en cuanto a la transferencia de datos y alimentación cabe recalcar que vienen dos puertos USB.

#### **2.1.16. Cámara de video conferencia GENIUS 1000X HD/720P/USB**

Conexión en 720p HD en aplicaciones de mensajería. Como se muestra en *Fig. 8*.

Especificaciones:

- Sensor de imagen CMOS de píxeles de alta definición 720p
- tipo de lente objetivo de enfoque manual
- Formato de archivo: AVI/WMV
- Con Micrófono: SÍ
- Resolución (DPI)1MP, 1280 x 720, 640 x 480 pixeles





Fig. 8 Cámara Web Genius 720p [16]

### 2.1.17. LiPo battery

Todos los vehículos pequeños no tripulados requieren de baterías para realizar su funcionamiento la cual son de manera recargable de polímero de litio (LiPo), para esto se debe tener en cuenta que existe una regla general en la cual se usa 1.000 mAH (miliamperios-hora) por cada motor. Esto significa que con una batería LiPo de 4.000 mAH sería suficiente para operar un vehículo.

En este caso se utiliza:

- Tattu de 3 celdas 11.1V- 2700mA
- Zippy compact 2450 mAH, 3 celdas 11.1V
- Turnigy 2200mAH 3S 11.1V

### **2.1.18. Cargador IMAX**

Este cargador como lo muestra en la *Fig. 9* cuenta con un microprocesador y software, dispone con opciones de carga controladas por un sistema de control de 2 vías, esto permite seguridad y minimizar posibles problemas de carga.

Práctico y funcional para el usuario cuenta con las siguientes características:

- Controlado de microprocesador
- Circuito de alta potencia y de alto rendimiento
- Seguridad del máximo
- Sensibilidad del voltaje máximo
- Límite de la capacidad
- Límite del tiempo de procedimiento
- Límite de la temperatura
- Monitor de potencia de entrada
- Monitorea el voltaje de la batería individual
- Detector del voltaje separado para el modo de carga de balance
- Rápido cargador para baterías de nicd/nimh/li-polímero/li-ion/pb
- Li-ion/LiPo/life: 1 a 6 baterías
- Nicd/nimh: 1 a 15 baterías
- Corriente de carga: 0.1 a 5a
- Corriente de descarga: 0.1 a 1a
- Entrada Dc: 11 a 18v



Fig. 9 Cargador IMAX

### **2.1.19. Cargador Turnigy 12v mini charger 2S/3S para batería LiPo**

Cargador de baterías de Litio Polímero (LiPo) de 2 y 3 celdas marca Turnigy tiene un adaptador de 2.1mm para la fuente de alimentación. Puede suministrar hasta 800mA de corriente de carga. Como se muestra en la *Fig. 10*.

Este cargador de balance Turnigy cuenta con una protección de la carga inteligente para garantizar que las baterías no estén sobrecargando y tiene supervisión en tiempo real aumentando el ciclo de vida de sus paquetes de baterías. Puede ser alimentado desde cualquier salida de 12V y cargas a 800mA. La carga es a través del conector JST-XH en su batería. Compatible con todos los Turnigy, Zippy, así como la mayoría de baterías.



Fig. 10 Cargador Turnigy 12v [17]

### 2.1.20. Cables

Elementos que te permite el paso de la corriente formado o utilizado comúnmente de cobre debido a su excelente conductividad de este material.

Según el uso del mismo en donde se desea emplear varía el grosor o numeración, es por ello que para este proyecto se utilizan:

- Cable #12
- Cable #24
- Jumpers
- Micro USB macho a USB hembra

### 2.1.21. Conectores

Dispositivo que tiene como función conectar o unir circuitos para formar una trayectoria continua en la cual fluya la corriente eléctrica, un ejemplo son los conectores XT60, como se muestra en la *Fig. 11*.

En este caso se utiliza:

- Jack DC
- Conectores balanceadores
- Borneras



Fig. 11 Conector XT60 [18]

### 2.1.22. Switch de 9 pines

Switch de bypass de 9 contactos y dos posiciones como se muestra en la **Fig. 12**. Este permite encender 3 circuitos de manera independiente.



Fig. 12 Switch de 9 pines [19]

### 2.1.23. 6+ channel RC transmitter and receiver

Para poder realizar pruebas de los motores en modo manual y modificar su manejo se debe adquirir un transmisor de radio de control, se puede usar distintos tipos de transmisor / receptor C teniendo en cuenta que se puede usar mínimo 6 canales.

ArduPilot requiere al menos 5 canales para la mayoría de los vehículos, sin embargo, de 8 a 16 canales están comúnmente disponibles en la mayoría de los sistemas y son muy convenientes para controlar otras funciones del vehículo, como cámaras u opciones de funciones de vuelo. Muchos vehículos requieren 8 canales solo para el funcionamiento básico, como muchos QuadPlanes.

A continuación, se muestra una **TABLA II** con algunos sistemas comúnmente disponibles que muestran estos elementos. Tenga en cuenta que no todas las versiones de transmisores y / o receptores de un fabricante en particular pueden tener estas características.

TABLA II  
TABLA DESCRIPTIVA DE RC

Fabricante original	Abarcar	Telemetría	Velocidad Telem	Pantalla TX	Protocolo RC	Notas
<b>DragonLink</b>	Largo	Bi-dir	56K	n / A	PPM_SUM / SBUS	1
<b>FLYSKY</b>	Pequeño	No	•	•	IBus	
<b>Serie FRSky X</b>	Pequeño	Bi-dir	Medio	sí	PPM-SUM / SBUS	2
<b>Serie FRSky R9</b>	Medio	Bi-dir	Medio	sí	PPM-SUM / SBUS	2

<b>Futaba</b>	Pequeño	No	•	•	SBus
<b>Graupner</b>	Pequeño	sí	Medio	sí	SUM-D
<b>Multicine</b>	Pequeño	No	•	•	SRXL1
<b>Spektrum</b>	Pequeño	No	•	•	DSM / DSM2 DSM-X / SRXL1

**Nota 1:** DragonLink proporciona un enlace transparente de 56 Kbaudios para telemetría, lo que permite una telemetría MAVLink completa hacia / desde el vehículo desde el módulo transmisor. Dragonlink es un módulo adicional para el transmisor, como un FRSky Taranis o RadioMaster T16. Ver los sistemas DragonLink RC

**Nota 2:** Consulte Script de telemetría Yaapu FrSky para OpenTX. Las futuras versiones de firmware ofrecerán la capacidad de cambiar parámetros a través de la telemetría FRSky desde un transmisor compatible con Open TX además de mostrar los datos de telemetría. La mayoría de los transmisores compatibles con FRSky utilizan OpenTX. Tenga en cuenta que los sistemas R9 no son del todo de largo alcance, sino de mucho más alcance que los sistemas FRSky normales, que se encuentran en el extremo más alto de la categoría de corto alcance con un alcance de 1,6 a 2 km.

Los transmisores RC permiten a la persona configurar el modo de operar, controlar el movimiento y la orientación del vehículo.

El rango de control RC varía mucho según el sistema utilizado, la instalación, las antenas utilizadas, el terreno e incluso las condiciones meteorológicas. Pero en general, para fines de discusión aquí, los sistemas RC se pueden clasificar en corto alcance (2 km y menos), medio (2-10 km) y largo alcance (> 10 km). Además, pueden ofrecer telemetría unidireccional (vehículo a transmisor) o bidireccional (vehículo a / desde transmisor).

### 2.1.24. Protocolo MAVLink

MAVLink es un protocolo especialmente de comunicación y se especializa en el intercambio de información entre la estación base y el vehículo, este protocolo fue diseñado en modo de una librería lo cual es muy ligera y posee la información adecuada y necesaria para realizar el intercambio de mensajes llevado a cabo en la comunicación.

Toda esta información se almacena en ficheros con extensión .xml, lo que nos ayuda a que sean empaquetada y empleados en diferentes lenguajes de programación, para poder facilitar el desarrollo de nuevas soluciones de software.

Este protocolo es muy sencillo de integrar en cualquier sistema ya puede ser en la estación base como en la del prototipo debido a que la definición de los mensajes se puede realizar mediante cabeceras en lenguaje C. Sin embargo, la librería llega a contener una gran variedad de mensajes que suelen ser muy útiles, al mismo tiempo se podrían llegar a crear mensajes personalizados.

### 2.1.25. Paquetes MAVLink

Consiste en una secuencia codificada de bytes, para crear la estructura de los paquetes de este protocolo para drones se sirvieron de los protocolos Controller Area Network bus y SAE AS-4. Como se observa en la **Fig. 13**, la cabecera está conformada por un gran conjunto de 6 bytes lo cual indican diferentes metadatos en referencia al mensaje.

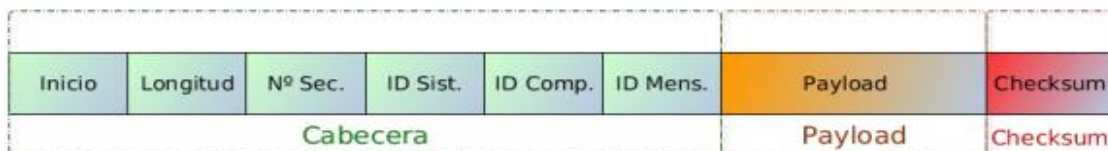


Fig. 13 Estructura de un paquete MAVLink



A continuación, en la **TABLA III** se detalla la funcionalidad de cada byte.

**TABLA III**  
**FUNCIÓN DE CADA BYTE DE LA CABECERA**

BYTES	EXPLICACIÓN
<b>Inicio</b>	Este byte indica el comienzo del paquete MAVLink.
<b>Longitud</b>	Indica el número de byte que se van a enviar como payload del mensaje. Puede tomar un valor entre 0 a 255.
<b>Número de secuencia</b>	Muestra el número de paquete enviando con el objetivo de reordenar o detectar paquetes perdidos. También puede tomar valores entre 0 y 255.
<b>Identificación del sistema</b>	Identifica el sistema, mediante un numero natural, que ha enviado el mensaje permitiendo diferencias entre varios sistemas de la misma red. Puede tomar un valor entre 0 a 255.
<b>Identificación del componente del sistema</b>	Señala, mediante un numero natural, el componente de dicho sistema que ha enviado el mensaje permitiendo de este modo diferencia varios componentes dentro del mismo sistema, toma valores entre 0 y 255.
<b>Identificación del sistema</b>	Indica el tipo de mensaje enviado, señalando como se tiene que leer el payload que viene a continuación. Utilizando un símil, es una leyenda que indica cómo se debe leer y traducir la carga útil del mensaje. Puede tomar un valor entre 0 y 255.

## **2.2.SOFTWARES UTILITARIOS**

### **2.2.1. SITL**

Este software permite básicamente ejecutar ArduPilot en su PC directamente, sin ser necesario utilizar un hardware especial. En esta ocasión se aprovecha el hecho de que ArduPilot es un piloto automático portátil y sencillo al momento de ejecutarse ya que cuenta con una gran variedad de plataformas. Su PC es solo otra plataforma en la que ArduPilot se puede construir y ejecutar. Al momento en que se requiere ejecutar en SITL, los datos que provienen del sensor se los obtiene de forma dinámica de un simulador de vuelo, este software contiene una gran variedad de simuladores de vehículos integrados y a su vez pueden interactuar con varios simuladores externos. Esto ayuda a que el ArduPilot pruebe en gran diversidad los tipos de vehículos que SITL puede simular:

- Aeronave multirroto
- Aeronave de ala fija
- Vehículos terrestres
- Vehículos submarinos
- Cardán de cámara
- Rastreadores de antena
- Una amplia variedad de sensores opcionales, como lidars y sensores de

flujo óptico. [20]

### **2.2.2. MISSION PLANNER**

Este software es considerado como una estación de control de tierra, fue creado por Michael Osborne, lo cual permite la planificación, el uso y calibración de todas aquellas

misiones y entre otras aplicaciones también encontramos la tarjeta de control APM de Arduino. También lleva consigo controladores para drones multirrotores y entre otros. Trabaja con líneas de código libre lo que hace que el desarrollo de un controlador propio o consiga la mejora de datos que ya existen. Cuenta con principales características:

- Descargar y analizar los registros de la misión creados por su piloto automático.
- Configura las opciones de la APM para su ejecución.
- Contiene interface de simulador de vuelo para PC y crea un simulador completo. [21]

## MARCO METODOLÓGICO

El enfoque de este proyecto es diseñar y construir un ROV funcional que se pueda operar tanto autónoma como manual para recoger y enviar datos en latitud como en longitud cuando detecta un objetivo mediante visión artificial. Se necesita establecer enlaces de radio-control y telemetría para establecer comunicación con el ROV desde una estación base. En la **Fig. 14**, se puede observar el esquema funcional del robot.

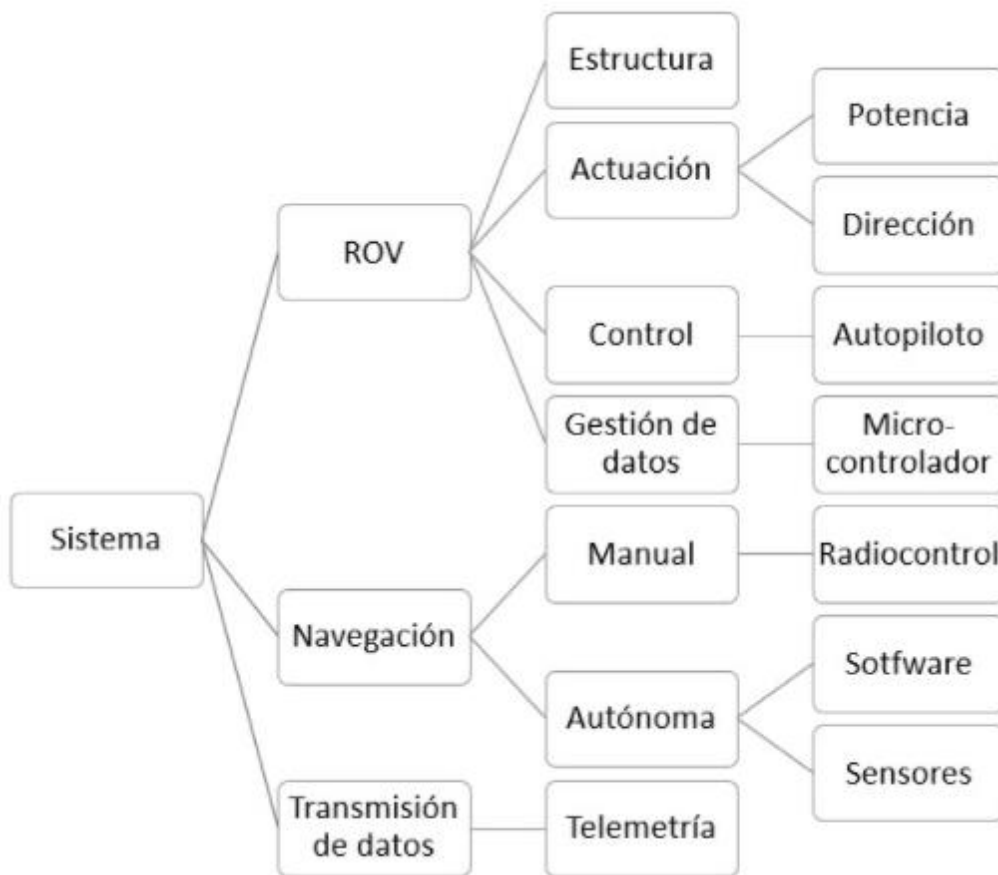
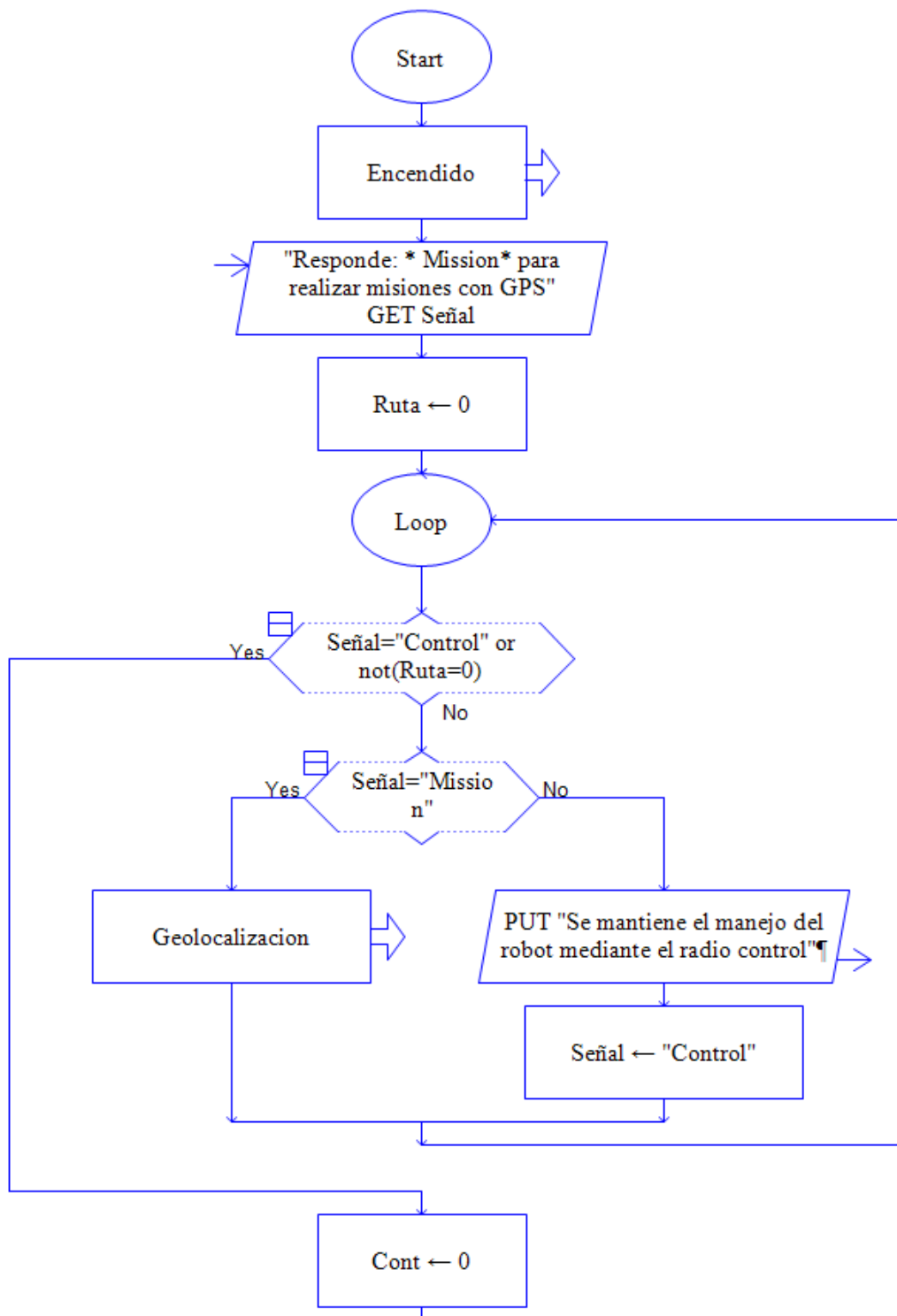


Fig. 14 Esquema del proyecto

Desde su creación de la estructura del ROV junto con la aplicación mecánica de la actuación tanto de potencia (motor) como de dirección (pixhawk y Arduino). Para el control del ROV, se utilizó un autopiloto y también se demostró cómo se llevó a cabo la

gestión de datos a través de un microcontrolador. En la **Fig. 15** se realiza el algoritmo a actuar del robot.



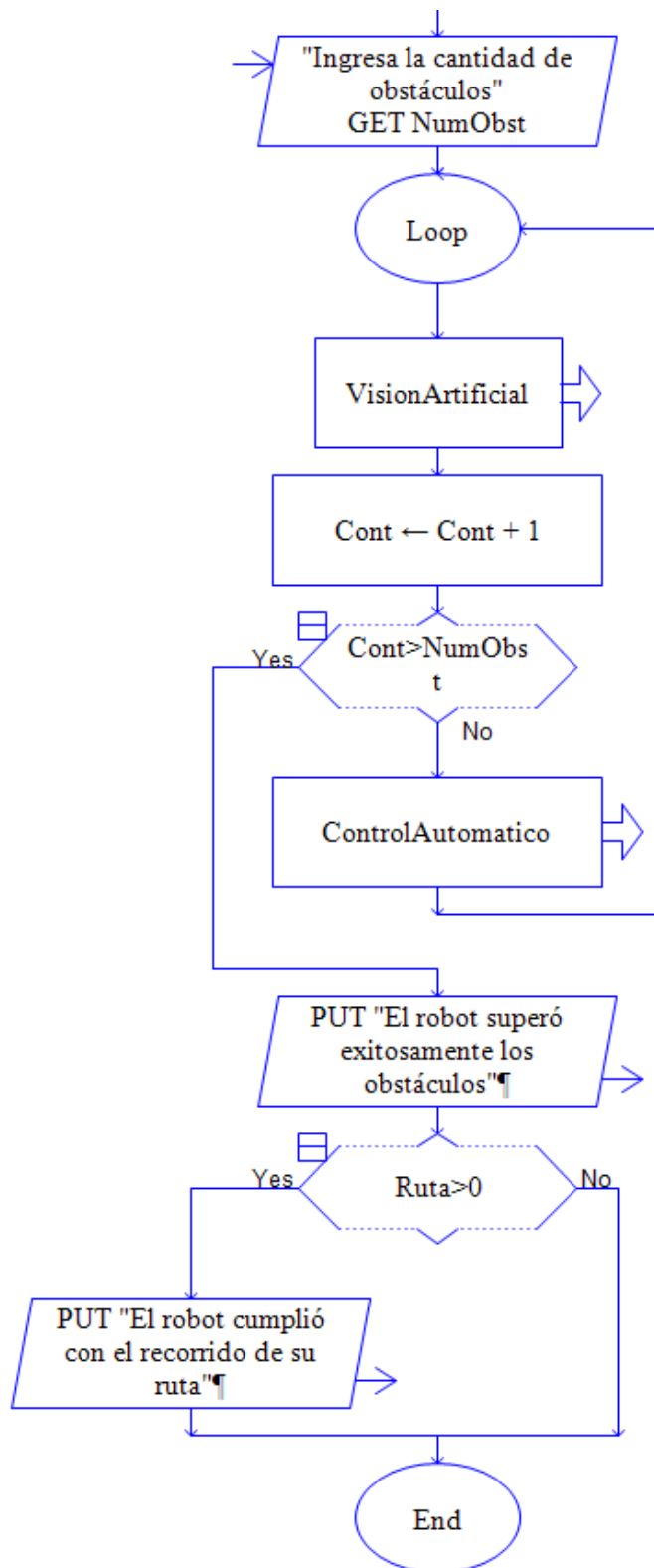


Fig. 15 Diagrama de flujo principal del funcionamiento del robot

Respecto a la navegación, se realizó de manera manual (por radio-control) y autónoma (por sensores ultrasónicos y Mission Planner). La transmisión de datos se realizará por telemetría para los datos de la navegación, esta segunda es parte del autopiloto y la lleva intrínseca en su programación.

### **3.1. FASE INICIAL**

#### **3.1.1. Estructura y enlaces**

En primer lugar, se encuentra el diseño como se muestra en el *ANEXO C: Fig. 76* y *Fig. 77*, creación y montaje, como se muestra en el *ANEXO C: Fig. 73, Fig. 74* y *Fig. 75* de la estructura base compuesta por metal o lata que aporta rigidez con cubierta realizada en impresión 3D permitiendo una buena funcionalidad a los mismos. Como se muestra en *Fig. 16*.

Características:

- Agujeros de motores: 2mm
- Altura: 18.5 cm
- Largo: 26.5cm
- Ancho: 29.5 cm
- Radio de giro: 40cm – 45 cm
- Diámetro de giro: 80 cm – 85 cm
- Detección de obstáculos: 60cm – 55cm
- Giro máximo: 30 cm – 35 cm

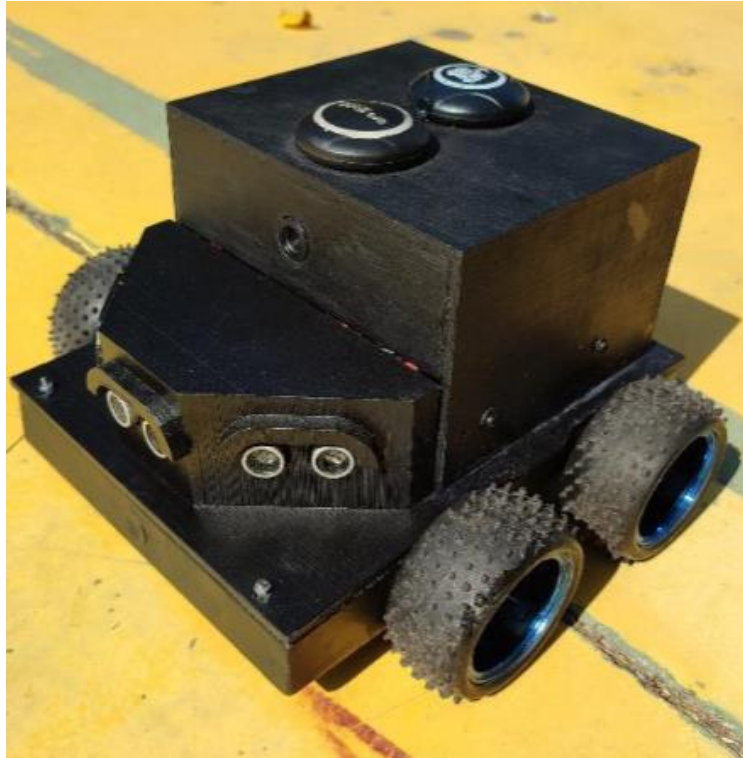


Fig. 16 Imagen de la estructura externa del robot

Los elementos que conforman este sistema son los que se pueden visualizar en la figura. En primer lugar, se requiere de un equipo informático o central (PC) que será su referencia como estación base y se encargará de monitorizar al ROV, transmitirle órdenes para la navegación autónoma, medio de visualización de los objetivos detectados por la visión artificial y de igual manera, será el equipo en el que se desea finalmente obtener los datos de los sensores.

Luego se encuentra el ROV, que se encargará de dirigirse a los diferentes puntos que desee el operador o a su vez los que establezca el mismo para poder recoger los datos captados por su cámara y transmitir esos datos recogidos a la central para poder procesarlos y guardarlos en un archivo Excel.



Finalmente, el último elemento es la emisora de radio-control que nos brinda pilotar de manera manual al ROV en el caso de que no se desee el pilotaje automático de la central como se muestra en *Fig. 17*, o en caso de avería que no responda a la telemetría o sufra algún inconveniente que le impida seguir de forma autónoma.



Fig. 17 Elementos del proyecto

### 3.1.2. Enlaces para la navegación

La navegación del ROV se puede efectuar de manera autónoma con la central a través de un enlace de telemetría en la banda de las radiofrecuencias de 433 MHz, ver en **ANEXO C: Fig. 81**. Este enlace también tiene la función de transmitir datos de navegación a tiempo real desde el ROV hacia la central. De lo contrario, si se desea una navegación manual, se dispone del radiocontrol el cual trabaja a una frecuencia de 2,4 GHz creando un segundo enlace, siendo posible el uso simultáneo de las dos opciones como se muestra en la *Fig. 18*.



Fig. 18 Enlaces para la navegación

## 3.2. CONFIGURACIÓN

### 3.2.1. Controlador PIXHAWK

En el diseño del piloto automático del ROV se utiliza el controlador Pixhawk porque es una placa diseñada para el control remoto de vehículos como se muestra en *Fig. 19*, equipado con la instrumentación necesaria para la navegación aérea o terrestre, ver *ANEXO C: Fig. 79 y Fig. 80*.



Fig. 19 Placa controladora Pixhawk

Para el control del ROV, se ha optado por programar este controlador con el firmware de un ROVER como lo muestra en la **Fig. 20**.

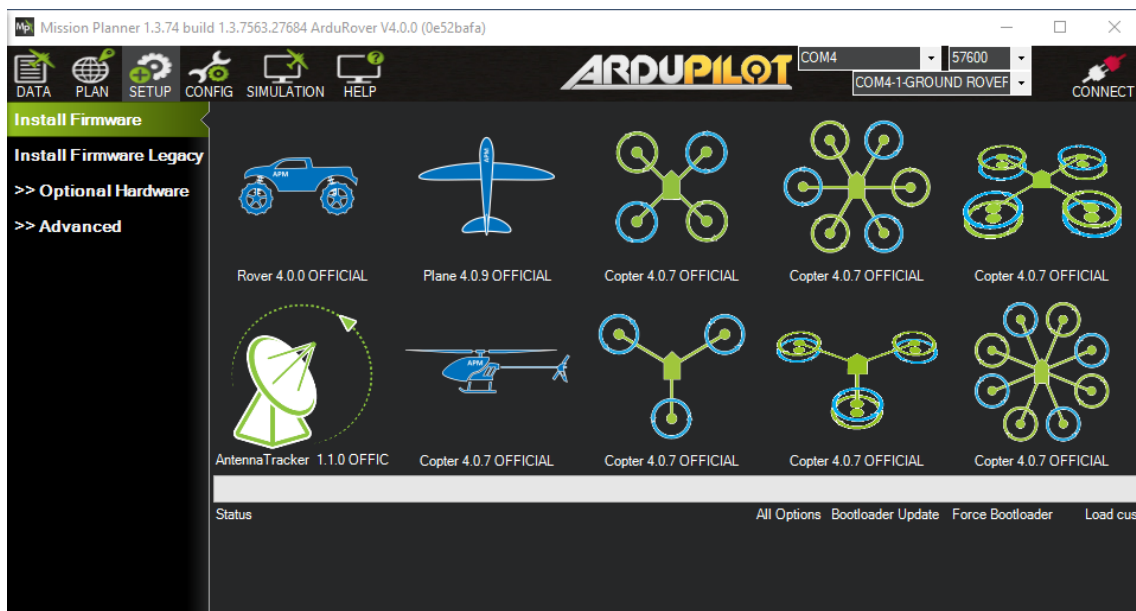


Fig. 20 Pantalla del ArduPilot

Luego hay que seleccionar en la pantalla el menú de “Initial Setup” y dentro de esta sección se debe realizar cuatro configuraciones obligatorias ya que el controlador Pixhawk también cuenta con una serie de sensores integrados para poder conocer con mayor precisión la posición en la que se encuentra. Los sensores equipados serían los siguientes mostrados en la **TABLA IV**.

TABLA IV  
TABLA DE SENSORES INTEGRADOS - PIXHAWK

SENSOR	FINALIDAD
<b>Giroscopio</b>	Conocer los ángulos en los que rota la placa (trabaja junto con el acelerómetro).

<b>Acelerómetro</b>	Conoce movimientos verticales u horizontales.
<b>Magnetómetro</b>	Conocer la dirección respecto al norte magnético terrestre.
<b>Barómetro</b>	Conocer la altitud a la que se encuentra.

Se debe tener en cuenta que este controlador no permite ponerse en marcha sin antes haber eliminado posibles problemas de calibrado o de inconsistencia en los valores de los sensores.

Como configuración obligatoria en primer lugar se encuentra la calibración del acelerómetro del robot como se muestra en la **Fig. 21**, que se debe dar clic en el primer botón Calibrate Accel, y notar que el primero es obligatorio para vehículos de 3 ejes, el segundo es opcional y más usado para vehículos aéreos ya que controla el eje de elevación.



Fig. 21 Pantalla de calibración del acelerómetro

Luego, se deberá colocar el vehículo sobre las seis caras diferentes de forma correcta con la mayor exactitud posible como se muestra en **Fig. 22** hasta **Fig. 27**.

1. Nivel:



Fig. 22 Prototipo colocado al nivel del piso

2. Lado izquierdo:



Fig. 23 Prototipo colocado al lado izquierdo

### 3. Lado derecho



Fig. 24 Prototipo colocado al lado derecho

### 4. Nariz arriba



Fig. 25 Prototipo colocado nariz arriba

## 5. Nariz abajo



Fig. 26 Prototipo colocado nariz abajo

## 6. Lado de abajo



Fig. 27 Prototipo colocado cabeza abajo

De igual manera, es importante indicar que la versión de este programa dice Modo de Vuelo 5, esto es error del programa, al culminar la calibración debe salir el botón bloqueado diciendo “Done” como lo muestra la **Fig. 28**.

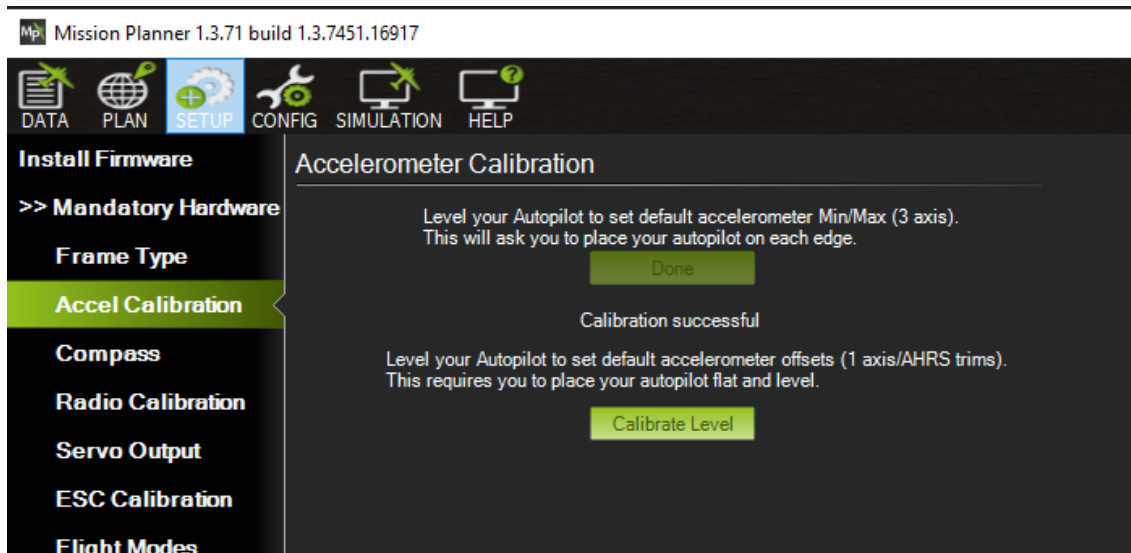


Fig. 28 Pantalla de calibración del acelerómetro

Seguidamente, para el barómetro, simplemente se deberá nivelar el vehículo en el plano que se desea comenzar navegar y este nivel será la referencia.

En tercer lugar, se deberá calibrar la brújula (compas) pulsando el botón Start para lo cual se empezarán a guardar las posiciones XYZ de la calibración de la brújula; la placa Pixhawk, tiene su propio compás y el GPS 3D también para ello se debe tener señal del GPS con buena saturación, ver también en el *ANEXO C: Fig. 82 y Fig. 83* para recibir la señal de la brújula que nos da el Satélite.

Una vez guardado los datos se quedarán en colores las compensaciones en verde quiere decir que esta correcto y amarillo advertencia que puede deberse esto a una lectura fallida o mal dato recibido del satélite para brújula #2. Para que el vehículo no tenga problemas al menos un compass debe estar bien calibrado y seleccionado como la brújula primaria tal como muestra en la *Fig. 29*.





Fig. 29 Pantalla de configuración de la brújula

Luego, el vehículo debe calibrarse para todos los ejes, es decir moviendo el vehículo de forma circular de un lado hacia el otro.

En cuarto lugar y como final de configuraciones obligatorias, se debe calibrar el control RC al máximo y mínimo de valores de las perillas, esto servirá para enviar la señal a los actuadores (Motores) sobre la velocidad y aceleración del robot como lo muestra en la **Fig. 30**. La sección del Servo Output, ESC Calibration Flight modes, no serán considerados ya que el robot no usa controladores de velocidad ESC, los modos de vuelo (Modos de operación) serán configurados por pantalla de Mision Planner y Servo Output serán configurados mediante Parámetros.

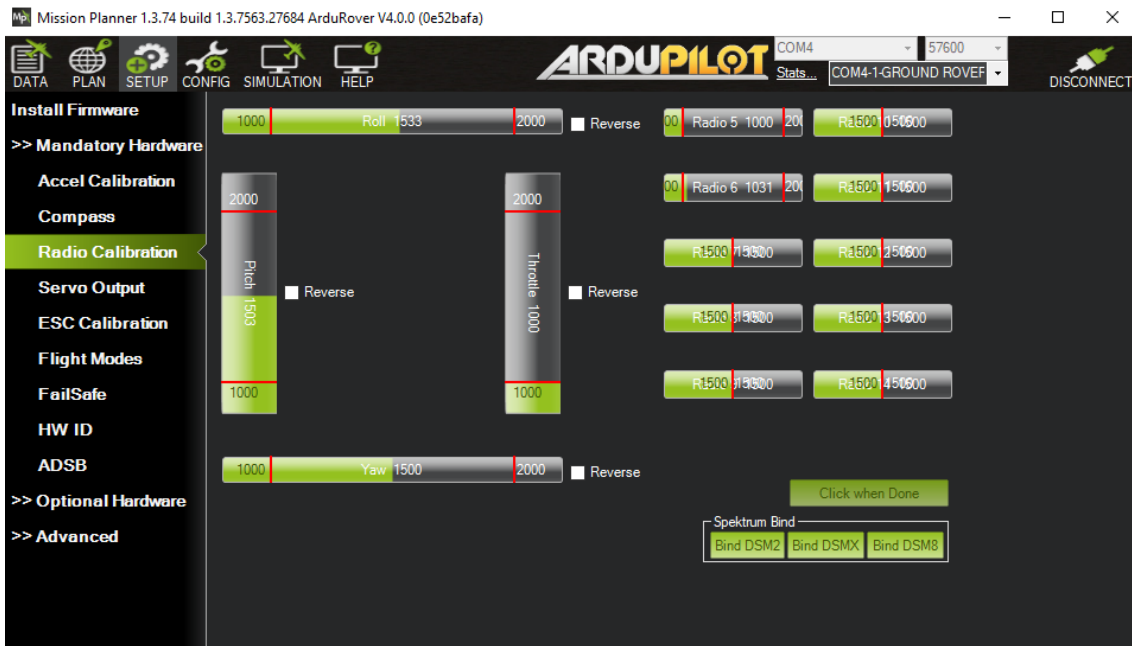


Fig. 30 Pantalla de calibración de la brújula

Para la realización de este segmento se procede de la siguiente manera:

- Abrir dentro de Mission Planner INITIAL SETUP | Mandatory Hardware | Pantalla de Radio Calibration.
- Haga clic en el botón verde "Calibrate Radio" en la parte inferior derecha.
- Presione "OK" cuando se le solicite para verificar que el equipo de control de radio esté encendido, la batería no esté conectada.
- Mueva las palancas de control, las perillas y los interruptores del transmisor hasta sus límites. Aparecerán líneas rojas a lo largo de las barras de calibración para mostrar los valores mínimos y máximos vistos hasta ahora como se muestra en la **Fig. 31**.



Fig. 31 Imagen de los rangos que tienen los controles

- Seleccione Click when Done
- Aparecerá una ventana con el mensaje, “Asegúrese de que todos sus sticks estén centrados y el acelerador esté abajo y haga clic en Aceptar para continuar”. Mueva el acelerador al centro y presione “OK” **Fig. 32**.
- Mission Planner mostrará un resumen de los datos de calibración. Los valores normales están alrededor de 1100 para mínimos y 1900 para máximos.

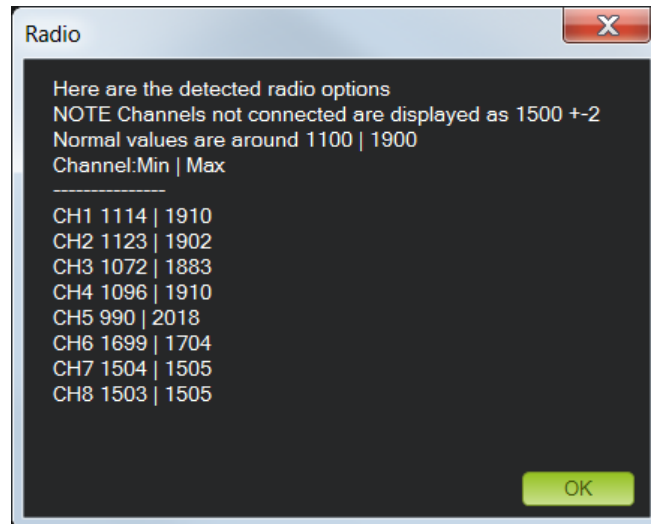


Fig. 32 Numeración de los Canales según su rango mínimo y máximo

En cuarto lugar, tenemos la configuración del Battery Monitor debido a que para este proyecto se tiene un monitor de voltaje conectado con la batería de LiPo Turnigy de 3000 mAH con 3 celdas de voltaje de 3,7V y se debe configurar como se muestra en la siguiente imagen **Fig. 33**:

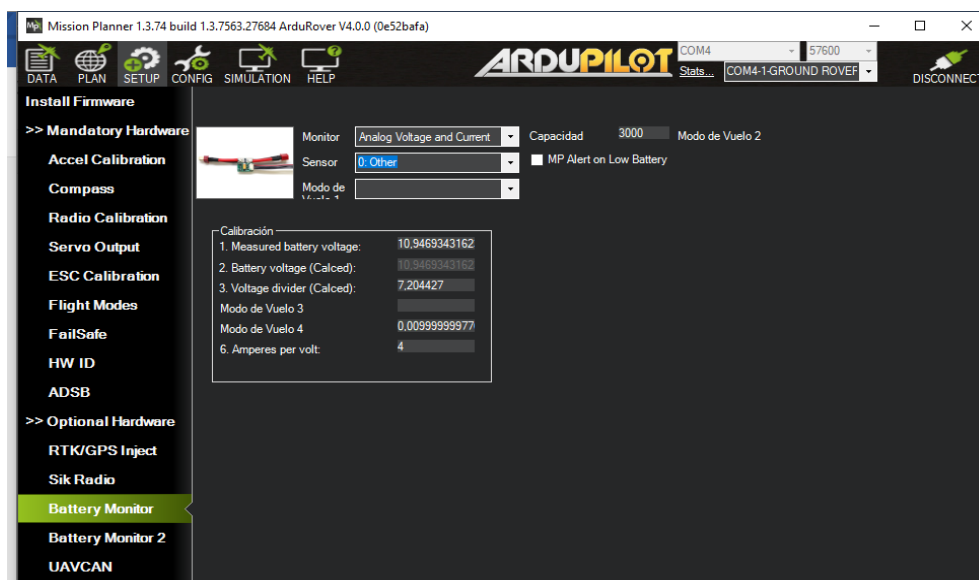


Fig. 33 Pantalla de configuración del Monitor de batería

En quinto lugar, se configurará el Basic Turning como se muestra en **Fig. 34** que se dejarán por defecto las configuraciones de PID como se muestran, aunque es recomendable ir cambiando dependiendo de la velocidad de desaceleración o el radio de giro en cada Way Point que se programe en las rutas automáticas.

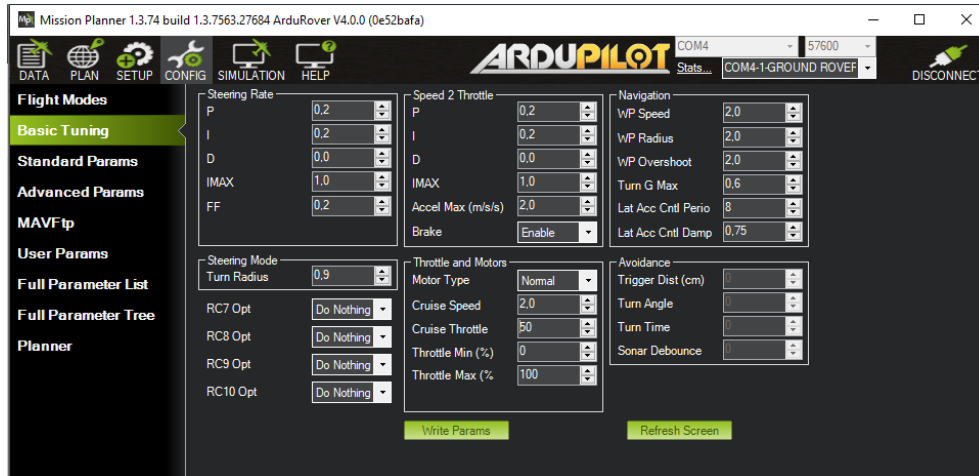


Fig. 34 Pantalla de configuración de las afinaciones básicas.

En sexto lugar, se tiene el Standard Params como se muestra en **Fig. 35** que se dejarán por defecto los parámetros y solo se editará “Arm Checks to Perform (ARMING CHECK)” y se dejará solo la opción de RC Channels para que, si existen parámetros que se descalibran, o que el GPS no reciba señal, o existan interferencias en la brújula, estos no interfieran al momento de hacer pruebas en el modo manual.

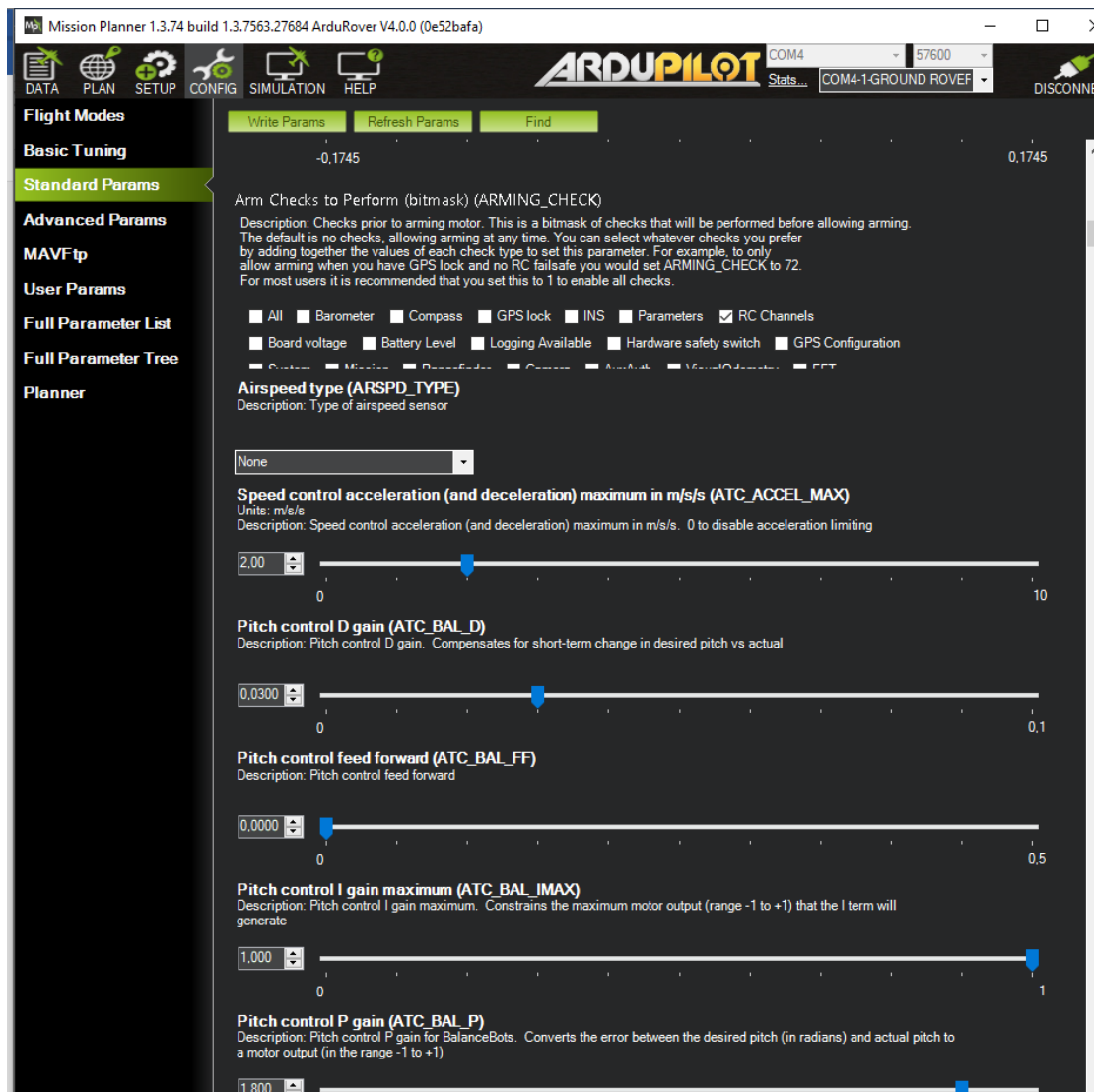


Fig. 35 Pantalla de configuración de parámetros estándar

En séptimo lugar, se debe configurar el Full Parameter List como se muestra en **Fig. 36** ya que estos parámetros serán los más importantes en la configuración del vehículo debido a que se configura como por ejemplos los pines de salidas y entradas tanto para los motores y periféricos.

Configuración de los servos (Salidas de los motores.)

SERVO\_FUNCTION = 73

## ThrottleLeft (Acelerador Izquierdo)

El SERVO1\_FUNCTION es la función de salida del servo, si lo configura en Desactivado (0), esta salida se configurará para el control mediante misiones automáticas o comandos de configuración de servo MAVLink. cualquier otro valor habilitará la función correspondiente.

EL SERVO1\_MAX es el ancho máximo de pulso PWM en microsegundos. Por lo general, 1000 es el límite inferior, 1500 es neutral y 2000 es el límite superior.

El SERVO1\_MIN es el ancho mínimo de pulso PWM en microsegundos. Por lo general, 1000 es el límite inferior, 1500 es neutral y 2000 es el límite superior.

Esta misma configuración se realiza para cada uno de los servos

SERVO1_FUNCTION	73	0:Disabled 1:RCPassThru 2:Rap 3:Rap_auto 4:Aleron 5:mount_pan 7:mount_roll 8:mount_roll 9:mount_open 10:camera_trigger 12:mount2_pan 13:mount2_roll 14:mount2_roll 15:mount2_open 16:DifferentialSpoilerLeft1 17:DifferentialSpoilerRight1 19:Elevator 21:Rudder 22:SprayerPump 23:SprayerSpinner 24:RaperonLeft 25:RaperonRight 26:GroundSteering 27:Parachute 28:Grppler 29:LandingGear 30:EngineRunEnable 31:HelRSC 32:HelTailRSC 33:Motor1 34:Motor2 35:Motor3 36:Motor4 37:Motor5 38:Motor6 39:Motor7 40:Motor8 41:Motor9 51:RCIN1 52:RCIN2 53:RCIN3 54:RCIN4 55:RCIN5 56:RCIN6 57:RCIN7 58:RCIN8 59:RCIN9 60:RCIN10 61:RCIN11 62:RCIN12 63:RCIN13 64:RCIN14 65:RCIN15 66:RCIN16 67:Ignition 69:Starter 70:Throttle 71:TrackerYaw 72:TrackerPitch 73:ThrottleLeft 74:ThrottleRight 75:MotorLeft 76:MotorRight 77:ElevonLeft 78:ElevonRight 79:VTailLeft 80:VTailRight 81:BoostThrottle 82:Motor9 83:Motor10 84:Motor11 85:Motor12 86:DifferentialSpoilerLeft2 87:DifferentialSpoilerRight2 88:Winch 89:Man Sail 90:CameraISO 91:CameraAperture 92:CameraFocus 93:CameraShutterSpeed 94:Script 1 95:Script 2 96:Script 3 97:Script 4 98:Script 5 99:Script 6 100:Script 7 101:Script 8 102:Script 9 103:Script 10 104:Script 11 105:Script 12 106:Script 13 107:Script 14 108:Script 15 109:Script 16 120:NeoPixel1 121:NeoPixel2 122:NeoPixel3 123:NeoPixel4 124:RateRoll 125:RatePitch 126:RateThrust 127:RateYaw 128:Wing Sail Elevator 129:ProfilLED 1 130:ProfilLED 2 131:ProfilLED 3 132:ProfilLED Clock 133:Winch Clutch	Function assigned to this servo. Seeing this to Disabled(0) will set up this output for control by auto missions or MAVLink servo set commands. any other value will enable the corresponding function	<input checked="" type="checkbox"/>	
SERVO1_MAX	30000	PWM	800 2200	maximum PWM pulse width in microseconds. Typically 1000 is lower limit, 1500 is neutral and 2000 is upper limit.	<input checked="" type="checkbox"/>
SERVO1_MIN	0	PWM	500 2200	minimum PWM pulse width in microseconds. Typically 1000 is lower limit, 1500 is neutral and 2000 is upper limit.	<input checked="" type="checkbox"/>
SERVO2_FUNCTION	74	0:Disabled 1:RCPassThru 2:Rap 3:Rap_auto 4:Aleron 5:mount_pan 7:mount_roll 8:mount_roll 9:mount_open 10:camera_trigger 12:mount2_pan 13:mount2_roll 14:mount2_roll 15:mount2_open 16:DifferentialSpoilerLeft1 17:DifferentialSpoilerRight1 19:Elevator 21:Rudder 22:SprayerPump 23:SprayerSpinner 24:RaperonLeft 25:RaperonRight 26:GroundSteering 27:Parachute 28:Grppler 29:LandingGear 30:EngineRunEnable 31:HelRSC 32:HelTailRSC 33:Motor1 34:Motor2 35:Motor3 36:Motor4 37:Motor5 38:Motor6 39:Motor7 40:Motor8 41:Motor9 51:RCIN1 52:RCIN2 53:RCIN3 54:RCIN4 55:RCIN5 56:RCIN6 57:RCIN7 58:RCIN8 59:RCIN9 60:RCIN10 61:RCIN11 62:RCIN12 63:RCIN13 64:RCIN14 65:RCIN15 66:RCIN16 67:Ignition 69:Starter 70:Throttle 71:TrackerYaw 72:TrackerPitch 73:ThrottleLeft 74:ThrottleRight 75:MotorLeft 76:MotorRight 77:ElevonLeft 78:ElevonRight 79:VTailLeft 80:VTailRight 81:BoostThrottle 82:Motor9 83:Motor10 84:Motor11 85:Motor12 86:DifferentialSpoilerLeft2 87:DifferentialSpoilerRight2 88:Winch 89:Man Sail 90:CameraISO 91:CameraAperture 92:CameraFocus 93:CameraShutterSpeed 94:Script 1 95:Script 2 96:Script 3 97:Script 4 98:Script 5 99:Script 6 100:Script 7 101:Script 8 102:Script 9 103:Script 10 104:Script 11 105:Script 12 106:Script 13 107:Script 14 108:Script 15 109:Script 16 120:NeoPixel1 121:NeoPixel2 122:NeoPixel3 123:NeoPixel4 124:RateRoll 125:RatePitch 126:RateThrust 127:RateYaw 128:Wing Sail Elevator 129:ProfilLED 1 130:ProfilLED 2 131:ProfilLED 3 132:ProfilLED Clock 133:Winch Clutch	Function assigned to this servo. Seeing this to Disabled(0) will set up this output for control by auto missions or MAVLink servo set commands. any other value will enable the corresponding function	<input checked="" type="checkbox"/>	
SERVO2_MAX	30000	PWM	800 2200	maximum PWM pulse width in microseconds. Typically 1000 is lower limit, 1500 is neutral and 2000 is upper limit.	<input checked="" type="checkbox"/>
SERVO2_MIN	0	PWM	500 2200	minimum PWM pulse width in microseconds. Typically 1000 is lower limit, 1500 is neutral and 2000 is upper limit.	<input checked="" type="checkbox"/>
SERVO3_FUNCTION	73	0:Disabled 1:RCPassThru 2:Rap 3:Rap_auto 4:Aleron 5:mount_pan 7:mount_roll 8:mount_roll 9:mount_open 10:camera_trigger 12:mount2_pan 13:mount2_roll 14:mount2_roll 15:mount2_open 16:DifferentialSpoilerLeft1 17:DifferentialSpoilerRight1 19:Elevator 21:Rudder 22:SprayerPump 23:SprayerSpinner 24:RaperonLeft 25:RaperonRight 26:GroundSteering 27:Parachute 28:Grppler 29:LandingGear 30:EngineRunEnable 31:HelRSC 32:HelTailRSC 33:Motor1 34:Motor2 35:Motor3 36:Motor4 37:Motor5 38:Motor6 39:Motor7 40:Motor8 41:Motor9 51:RCIN1 52:RCIN2 53:RCIN3 54:RCIN4 55:RCIN5 56:RCIN6 57:RCIN7 58:RCIN8 59:RCIN9 60:RCIN10 61:RCIN11 62:RCIN12 63:RCIN13 64:RCIN14 65:RCIN15 66:RCIN16 67:Ignition 69:Starter 70:Throttle 71:TrackerYaw 72:TrackerPitch 73:ThrottleLeft 74:ThrottleRight 75:MotorLeft 76:MotorRight 77:ElevonLeft 78:ElevonRight 79:VTailLeft 80:VTailRight 81:BoostThrottle 82:Motor9 83:Motor10 84:Motor11 85:Motor12 86:DifferentialSpoilerLeft2 87:DifferentialSpoilerRight2 88:Winch 89:Man Sail 90:CameraISO 91:CameraAperture 92:CameraFocus 93:CameraShutterSpeed 94:Script 1 95:Script 2 96:Script 3 97:Script 4 98:Script 5 99:Script 6 100:Script 7 101:Script 8 102:Script 9 103:Script 10 104:Script 11 105:Script 12 106:Script 13 107:Script 14 108:Script 15 109:Script 16 120:NeoPixel1 121:NeoPixel2 122:NeoPixel3 123:NeoPixel4 124:RateRoll 125:RatePitch 126:RateThrust 127:RateYaw 128:Wing Sail Elevator 129:ProfilLED 1 130:ProfilLED 2 131:ProfilLED 3 132:ProfilLED Clock 133:Winch Clutch	Function assigned to this servo. Seeing this to Disabled(0) will set up this output for control by auto missions or MAVLink servo set commands. any other value will enable the corresponding function	<input checked="" type="checkbox"/>	
SERVO3_MAX	30000	PWM	800 2200	maximum PWM pulse width in microseconds. Typically 1000 is lower limit, 1500 is neutral and 2000 is upper limit.	<input checked="" type="checkbox"/>
SERVO3_MIN	0	PWM	500 2200	minimum PWM pulse width in microseconds. Typically 1000 is lower limit, 1500 is neutral and 2000 is upper limit.	<input checked="" type="checkbox"/>
SERVO4_FUNCTION	74	0:Disabled 1:RCPassThru 2:Rap 3:Rap_auto 4:Aleron 5:mount_pan 7:mount_roll 8:mount_roll 9:mount_open 10:camera_trigger 12:mount2_pan 13:mount2_roll 14:mount2_roll 15:mount2_open 16:DifferentialSpoilerLeft1 17:DifferentialSpoilerRight1 19:Elevator 21:Rudder 22:SprayerPump 23:SprayerSpinner 24:RaperonLeft 25:RaperonRight 26:GroundSteering 27:Parachute 28:Grppler 29:LandingGear 30:EngineRunEnable 31:HelRSC 32:HelTailRSC 33:Motor1 34:Motor2 35:Motor3 36:Motor4 37:Motor5 38:Motor6 39:Motor7 40:Motor8 41:Motor9 51:RCIN1 52:RCIN2 53:RCIN3 54:RCIN4 55:RCIN5 56:RCIN6 57:RCIN7 58:RCIN8 59:RCIN9 60:RCIN10 61:RCIN11 62:RCIN12 63:RCIN13 64:RCIN14 65:RCIN15 66:RCIN16 67:Ignition 69:Starter 70:Throttle 71:TrackerYaw 72:TrackerPitch 73:ThrottleLeft 74:ThrottleRight 75:MotorLeft 76:MotorRight 77:ElevonLeft 78:ElevonRight 79:VTailLeft 80:VTailRight 81:BoostThrottle 82:Motor9 83:Motor10 84:Motor11 85:Motor12 86:DifferentialSpoilerLeft2 87:DifferentialSpoilerRight2 88:Winch 89:Man Sail 90:CameraISO 91:CameraAperture 92:CameraFocus 93:CameraShutterSpeed 94:Script 1 95:Script 2 96:Script 3 97:Script 4 98:Script 5 99:Script 6 100:Script 7 101:Script 8 102:Script 9 103:Script 10 104:Script 11 105:Script 12 106:Script 13 107:Script 14 108:Script 15 109:Script 16 120:NeoPixel1 121:NeoPixel2 122:NeoPixel3 123:NeoPixel4 124:RateRoll 125:RatePitch 126:RateThrust 127:RateYaw 128:Wing Sail Elevator 129:ProfilLED 1 130:ProfilLED 2 131:ProfilLED 3 132:ProfilLED Clock 133:Winch Clutch	Function assigned to this servo. Seeing this to Disabled(0) will set up this output for control by auto missions or MAVLink servo set commands. any other value will enable the corresponding function	<input checked="" type="checkbox"/>	
SERVO4_MAX	30000	PWM	800 2200	maximum PWM pulse width in microseconds. Typically 1000 is lower limit, 1500 is neutral and 2000 is upper limit.	<input checked="" type="checkbox"/>
SERVO4_MIN	0	PWM	500 2200	minimum PWM pulse width in microseconds. Typically 1000 is lower limit, 1500 is neutral and 2000 is upper limit.	<input checked="" type="checkbox"/>

Fig. 36 Pantalla de configuración de todos los parámetros.

En el RC5\_OPTION y RC6\_OTION es la configuración de entrada RC, como se muestra en **Fig. 37**.

RC5_OPTION	34	0: Do Nothing, 4: RTL, 5: Save Trim, 7: Save WP, 9: Camera Trigger, 11: Fence, 16: Auto, 19: Gripper, 24: Auto Mission Reset, 28: Relay On/Off, 30: Lost Rover Sound, 31: Motor Emergency Stop, 34: Relay2 On/Off, 35: Relay3 On/Off, 36: Relay4 On/Off, 40: Proximity Avoidance, 41: ArmDisarm, 42: SmartRTL, 46: RC Override Enable, 50: LearnCruise, 51: Manual, 52: Acro, 53: Steering, 54: Hold, 55: Guided, 56: Loiter, 57: Follow, 58: Clear Waypoints, 59: Simple Mode, 62: Compass Learn, 63: Sailboat Tack, 65: GPS Disable, 66: Relay5, 67: Relay6, 74: Sailboat motoring, 8pos, 78: RunCam Control, 79: RunCam OSD Control, 80: Viso Align, 81: Disarm, 100: KILLMU1, 101: KILLMU2, 102: Camera Mode Toggle, 105: GPS Disable Yaw, 207: Man/Sat, 300: Scripting1, 301: Scripting2, 302: Scripting3, 303: Scripting4, 304: Scripting5, 305: Scripting6, 306: Scripting7, 307: Scripting8	Function assigned to this RC channel
RC6_OPTION	41	0: Do Nothing, 4: RTL, 5: Save Trim, 7: Save WP, 9: Camera Trigger, 11: Fence, 16: Auto, 19: Gripper, 24: Auto Mission Reset, 28: Relay On/Off, 30: Lost Rover Sound, 31: Motor Emergency Stop, 34: Relay2 On/Off, 35: Relay3 On/Off, 36: Relay4 On/Off, 40: Proximity Avoidance, 41: ArmDisarm, 42: SmartRTL, 46: RC Override Enable, 50: LearnCruise, 51: Manual, 52: Acro, 53: Steering, 54: Hold, 55: Guided, 56: Loiter, 57: Follow, 58: Clear Waypoints, 59: Simple Mode, 62: Compass Learn, 63: Sailboat Tack, 65: GPS Disable, 66: Relay5, 67: Relay6, 74: Sailboat motoring, 8pos, 78: RunCam Control, 79: RunCam OSD Control, 80: Viso Align, 81: Disarm, 100: KILLMU1, 101: KILLMU2, 102: Camera Mode Toggle, 105: GPS Disable Yaw, 207: Man/Sat, 300: Scripting1, 301: Scripting2, 302: Scripting3, 303: Scripting4, 304: Scripting5, 305: Scripting6, 306: Scripting7, 307: Scripting8	Function assigned to this RC channel

Fig. 37 Sección de la configuración RC

El WP\_RADIUS es el radio del waypoint que Define la distancia máxima desde un waypoint que cuando se cruza indica que el waypoint puede estar completo. Para evitar que el prototipo dé vueltas alrededor del waypoint en caso de que falle por más de WP\_RADIUS, se realiza una verificación adicional para ver si el prototipo ha cruzado una "línea de meta" pasando por el waypoint y perpendicular a la trayectoria desde el waypoint anterior. Si se cruza esa línea de meta, el waypoint se considera completo. Tenga en cuenta que el controlador de navegación puede decidir girar más tarde que WP\_RADIUS antes de un waypoint, en función de la brusquedad del giro y la velocidad del prototipo. Es seguro configurar WP\_RADIUS mucho más grande que el radio de giro habitual de su prototipo y el controlador de navegación determinará cuándo girar. Si configura WP\_RADIUS demasiado pequeño, tenderá a sobrepasar los giros.

Como se ve en la

WP_OVERSHOOT	2	m	0.10	Waypoint overshoot maximum in meters. The vehicle will attempt to stay within this many meters of the track as it completes one waypoint and moves to the next.
WP_PIVOT_ANGLE	60	deg	0.360	Pivot when the difference between the vehicle's heading and its target heading is more than this many degrees. Set to zero to disable pivot turns
WP_PIVOT_RATE	90	deg/s	0.360	Turn rate during pivot turns
WP_RADIUS	2	m	0.100	The distance in meters from a waypoint when we consider the waypoint has been reached. This determines when the vehicle will turn toward the next waypoint.
WP_SPEED	2	m/s	0.100	Waypoint speed default
WP_SPEED_MIN	0	m/s	0.100	Vehicle will not slow below this speed for corners. Should be set to boat's plane speed. Does not apply to pivot turns.
WRC_ENABLE	0		0:Disabled 1:Enabled	Enable or disable wheel rate control



**Fig. 38.**

WP_OVERSHOOT	2	m	0 10	Waypoint overshoot maximum in meters. The vehicle will attempt to stay within this many meters of the track as it completes one waypoint and moves to the next.
WP_PIVOT_ANGLE	60	deg	0 360	Pivot when the difference between the vehicle's heading and its target heading is more than this many degrees. Set to zero to disable pivot turns
WP_PIVOT_RATE	90	deg/s	0 360	Turn rate during pivot turns
WP_RADIUS	2	m	0 100	The distance in meters from a waypoint when we consider the waypoint has been reached. This determines when the vehicle will turn toward the next waypoint.
WP_SPEED	2	m/s	0 100	Waypoint speed default
WP_SPEED_MIN	0	m/s	0 100	Vehicle will not slow below this speed for corners. Should be set to boat's plane speed. Does not apply to pivot turns.
WRC_ENABLE	0		0:Disabled 1:Enabled	Enable or disable wheel rate control

Fig. 38 Sección de la configuración Waitpoint

El RNGFND1\_TYPE es el tipo de telémetro; es decir, la selección de que tipo de telémetro está conectado. Como se muestra en **Fig. 39**.

El RNGFND1\_FUNCTION es el control sobre qué función se utiliza para calcular la distancia. Para una función lineal, la distancia es (compensación de voltaje) \* escala. Para una función invertida, la distancia es (voltaje de compensación) \* escala. Para una función hiperbólica, la distancia es escalada / (compensación de voltaje). Las funciones devuelven la distancia en metros.

El RNGFND1\_PIN es el Pin de entrada analógica o PWM al que está conectado el telémetro. Los puertos Airspeed se pueden usar para entrada analógica, AUXOUT se puede usar para entrada PWM.

El RNGFND1\_PWRRNG es el rango de ahorro de energía, el cual establece la distancia estimada del terreno en metros por encima de la cual el sensor se pondrá en modo de ahorro de energía (si está disponible). Un valor de cero significa que el ahorro de energía no está habilitado.

El RNGFND1\_STOP\_PIN es un pin de parada del telémetro, este es un Pin digital que habilita / deshabilita la medición del telémetro para el telémetro pwm. Un valor de -1

significa que no hay pin. Si esto está configurado, entonces el pin se establece en 1 para habilitar el telémetro y en 0 para deshabilitarlo. Esto se utiliza para permitir el ahorro de energía cuando está fuera de rango.

El RNFND1\_SCALING es un escalado del telémetro, el cual es un factor de escala entre la lectura del telémetro y la distancia. Para las funciones lineal e invertida, esto está en metros por voltio. Para la función hiperbólica, las unidades son metros Voltios. [22]

RNFND1_FUNCTION	0		0:Linear 1:Inverted 2:Hyperbolic	Control over what function is used to calculate distance. For a linear function, the distance is (voltage-offset)/scaling. For an inverted function the distance is (offset-voltage)/scaling. For a hyperbolic function the distance is scaling/(voltage-offset). The functions return the distance in meters.
RNFND1_PIN	54		-1:Not Used 11:PX4-airspeed port 15:Pixhawk-airspeed port 50:Pixhawk AUXOUT1 51:Pixhawk AUXOUT2 52:Pixhawk AUXOUT3 53:Pixhawk AUXOUT4 54:Pixhawk AUXOUT5 55:Pixhawk AUXOUT6	Analog or PWM input pin that rangefinder is connected to. Airspeed ports can be used for Analog input. AUXOUT can be used for PWM input.
RNFND1_PWRWRNG	0	m	0 32767	This parameter sets the estimated terrain distance in meters above which the sensor will be put into a power saving mode (if available). A value of zero means power saving is not enabled.
RNFND1_SCALING	3	m/V		Scaling factor between rangefinder reading and distance. For the linear and inverted functions this is in meters per volt. For the hyperbolic function the units are meterVolts.
RNFND1_STOP_PIN	55		-1:Not Used 50:Pixhawk AUXOUT1 51:Pixhawk AUXOUT2 52:Pixhawk AUXOUT3 53:Pixhawk AUXOUT4 54:Pixhawk AUXOUT5 55:Pixhawk AUXOUT6 111:PX4 FMU Relay 1 112:PX4 FMU Relay 2 113:PX4IO Relay 1 114:PX4IO Relay 2 115:PX4IO ACC1 116:PX4IO ACC2	Digital pin that enables/disables rangefinder measurement for the pwm rangefinder. A value of -1 means no pin. If this is set, then the pin is set to 1 to enable the rangefinder and set to 0 to disable it. This is used to enable powersaving when out of range.
RNFND1_TYPE	30		0:None 1:Analog 2:MaobotixI2C 3:LidarLite-I2C 5:PWM 6:BBB-PRU 7:LightWareI2C 8:LightWareSerial 9:BeBop 10:MAVLink 11:Landing 12:LeddarOne 13:MaibotuSerial 14:TeraRangerI2C 15:LidarLiteV3-I2C 16:VL53L0X or VL53L1X 17:NMEA 18:WASP-LRF 19:BenewakeTF02 20:Benewake-Serial 21:LidarLightV3HP 22:PWM 23:BlueRoboticsPing 24:UAVCAN 25:BenewakeTFminiPlus-I2C 26:LarbasPSK-CM3U165-CC5 27:BenewakeTF03 28:VL53L1X-ShortRange 29:LeddarVu8-Serial 30:HC-SR04 31:GY-US42v2	What type of rangefinder device that is connected

Fig. 39 Sección de la configuración del telémetro

### 3.3. ACTUACIÓN

Las señales PWM de control del Pixhawk son normalmente leídas por un controlador ESC (Electronic Speed Controller), también llamado variador de velocidad y para la elaboración del código y calibración de los sensores ultrasónicos es necesario que este microcontrolador trabaje con niveles de tensión de 0 a 5 V para que sea compatible con el conjunto del sistema y que la alimentación de la batería instalada de 11,1 V sea suficiente para su correcto funcionamiento tal como se puede encontrar en la referencia.

Por esta razón se ha elegido implementar el microcontrolador Arduino UNO como se muestra en **Fig. 40**, ya que cumple estos requisitos y es ampliamente utilizado para el control.



Fig. 40 Placa Arduino UNO

El controlador Pixhawk utiliza señales PWM cuya duración significa lo siguiente:

- 2000  $\mu\text{s}$  Máxima velocidad hacia delante.
- 1500  $\mu\text{s}$  Estado en reposo.
- 1000  $\mu\text{s}$  Máxima velocidad hacia atrás.

La programación del Arduino tiene como objetivo realizar recoger, guardar y transmitir los datos obtenidos de los sensores a través del sensor HC-SR04 que es un sensor de distancia que utiliza ultrasonido para determinar la distancia de un objeto en un rango de 2 a 450 cm, estos sensores ultrasónicos miden la distancia mediante el uso de ondas ultrasónicas y se destaca por su pequeño tamaño, bajo consumo energético y buena precisión el cual permitirá obtener un mejor manejo y dirección cuando se esté desarrollando las rutas. En este caso, para el prototipo se utilizan 3 sensores ultrasónicos cuyos grados y distancia de detección de obstáculos son los que se muestran en la **Fig. 41**. Visualizar código de la programación de los sensores en **ANEXO A: REFERENCIA 1**. El datasheet de los sensores se visualiza en el **ANEXOS B: DATASHEET DE EQUIPOS - DATASHEET DEL SENSOR ULTRASÓNICO HC-SR04**.

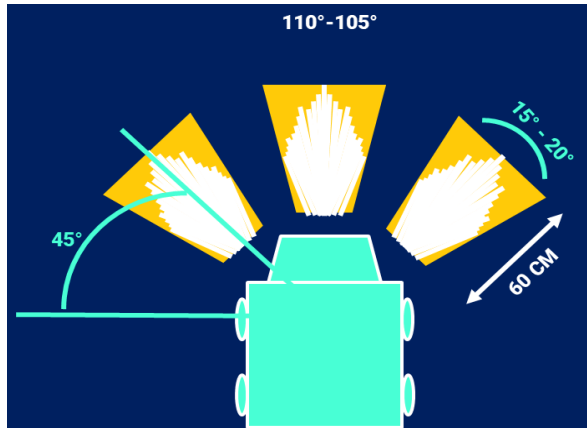
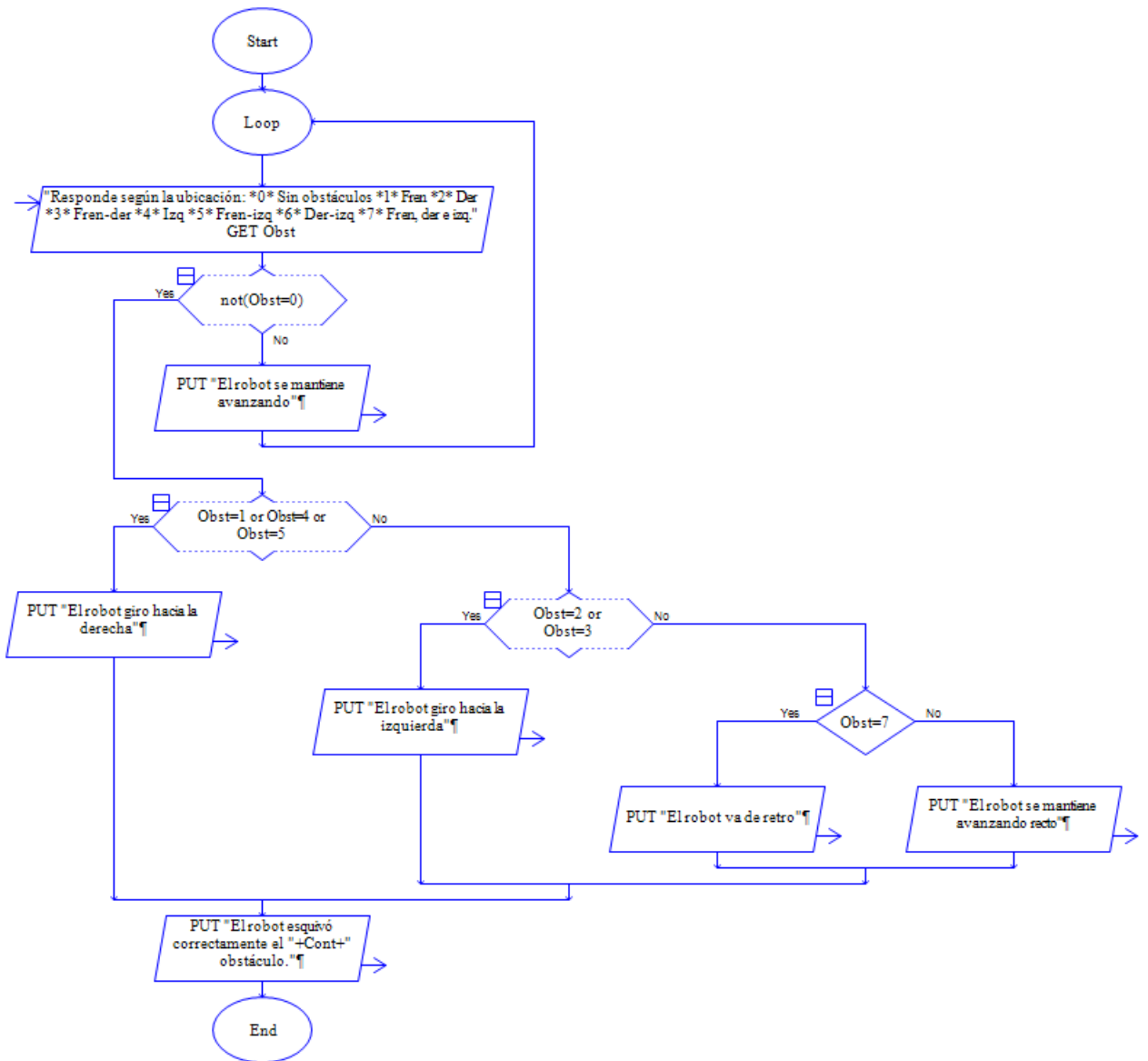


Fig. 41 Grados de distancia de detección del robot



### 3.4. NAVEGACIÓN

El controlador Pixhawk permite dos tipos de navegación: Autónoma guiada a través de la plataforma Mission Planner o bien Manual mediante un control remoto (a radiocontrol).

#### 3.4.1. Navegación Manual

En la **Fig. 42** se realiza el algoritmo a actuar del robot.

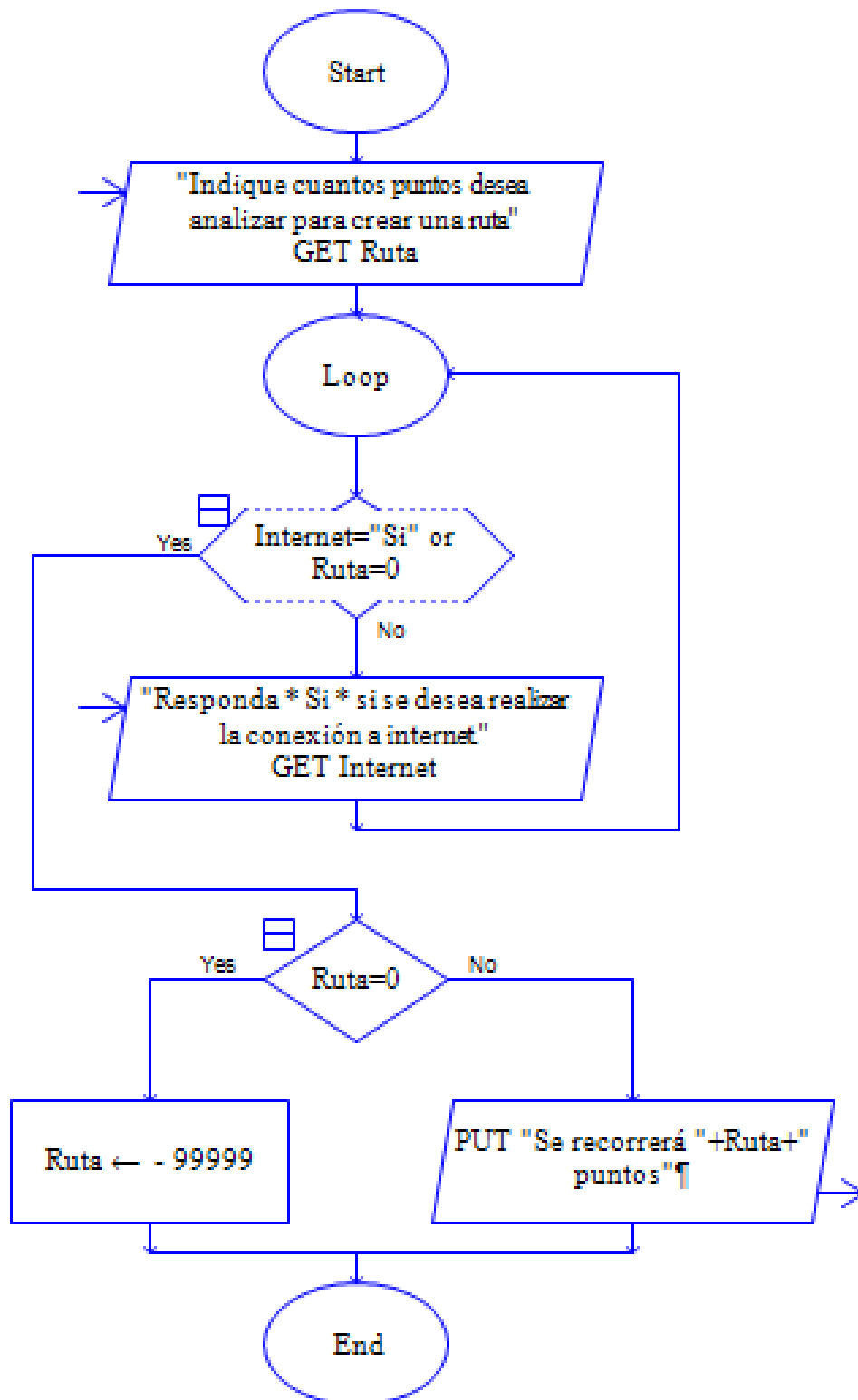


Fig. 42 Diagrama de flujo del funcionamiento del geoposicionamiento para la realizar la misión

Para poder realizar pruebas de los motores en modo manual y modificar su manejo se debe adquirir un transmisor de radio de control, se puede usar distintos tipos de transmisor / receptor.

Los transmisores RC permiten a la persona configurar el modo de operar, controlar el movimiento y la orientación del vehículo como lo muestra en el *ANEXO C: Fig. 78*.

La calibración RC implica capturar los valores mínimos, máximos y de "recorte" de cada canal de entrada RC para que ArduPilot (que es un software de código abierto) pueda interpretar correctamente la entrada.

El número de canales determina cuantas acciones individuales pueden ser controladas en el aparato. Por ejemplo, para avanzar, retroceder, moverse hacia la derecha o izquierda cada uno de ellos exige un canal. El ArduPilot requiere al menos 5 canales para la mayoría de los vehículos, sin embargo, de 8 a 16 canales están comúnmente disponibles en la mayoría de los sistemas y son muy convenientes para controlar otras funciones del vehículo.

El radio control presentado es flysky fs-i6x 2,4 de la *Fig. 43* gigahercios con receptor fs-ia6b es uno de los más indicados para este tipo de prototipo por su estructura y características que aportan al prototipo, en este caso como nuestro prototipo al iniciar o estar en ruta la radio telemetría nos permite comunicar nuestra estación base con el vehículo y a su vez hace también que podamos incluir a nuestros sensores y placas controladoras permitiendo detectar objetos en el camino y que estos los esquive, pero sin olvidar a donde quiere llegar.



Fig. 43 Imagen del Radio Control RC

Para poder realizar un pilotaje manual, primero de todo se deberá instalar el receptor del radiocontrol en el ROV. Este receptor consta de tres canales, pero sólo dos sirven para el control, los cuales son el de potencia del motor y el de dirección del timón (el tercero es para emparejar con otra emisora si fuera necesario). Estos canales del receptor devuelven señales PWM tal como se ha mencionado en el apartado de la actuación. El controlador Pixhawk cuenta con una entrada RC (Radio-Control) en la que se debe introducir una señal PPM de ocho canales, siendo cada uno una actuación específica. Es aquí donde se aplica el codificador PPM, el cual cuenta con los ocho canales necesarios, pero de los cuales solamente se utilizarán dos. Según las especificaciones del codificador PPM, el canal que se encarga de la dirección es el canal 1 (CH1) y el que se encarga de la potencia del motor es el canal 3 (CH3). Mientras que en el receptor del radiocontrol el canal 1 es también para la dirección y el canal 2 es para la potencia. Por tanto, habrá que conectar el canal 1 del receptor al canal 1 del codificador PPM y el canal 2 del receptor al canal 3 del codificador. Conectando así la salida PPM al correspondiente pin de RC del Pixhawk que se muestra en la **Fig. 44**.



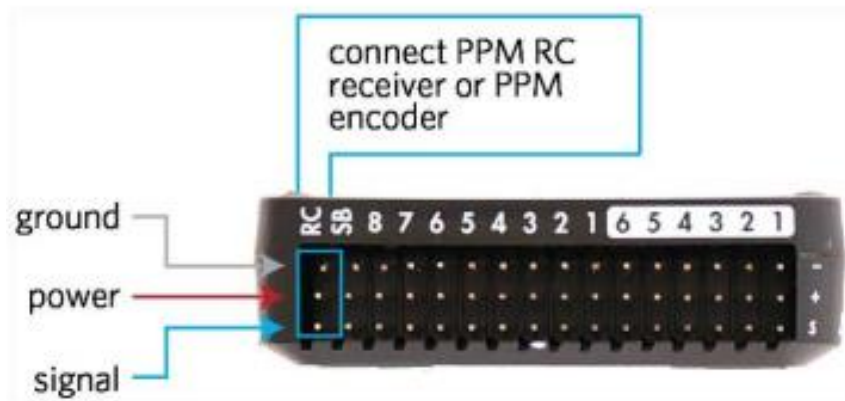


Fig. 44 Pines posteriores de la placa pixhawk

Una vez realizada la conexión se puede comenzar a utilizar el radiocontrol para la navegación. Las condiciones para poder armar el motor son las mismas que con la navegación autónoma: debe tener buena visibilidad GPS, ningún error en los sensores y se deberá armar el motor manteniendo pulsado el pulsador de seguridad. [23]

La diferencia es que, para armar el motor en este caso se deberá mantener el timón abajo y los switches que están en la parte superior del radio control posicionarlos hacia arriba y girar la perilla para proceder armar el motor. También se puede armar a través de la plataforma Mission Planner y utilizar el radiocontrol para pilotarlo y viceversa, ambos modos son compatibles simultáneamente.

### 3.4.2. Navegación autónoma.

En la *Fig. 45* se realiza el algoritmo a actuar del robot.

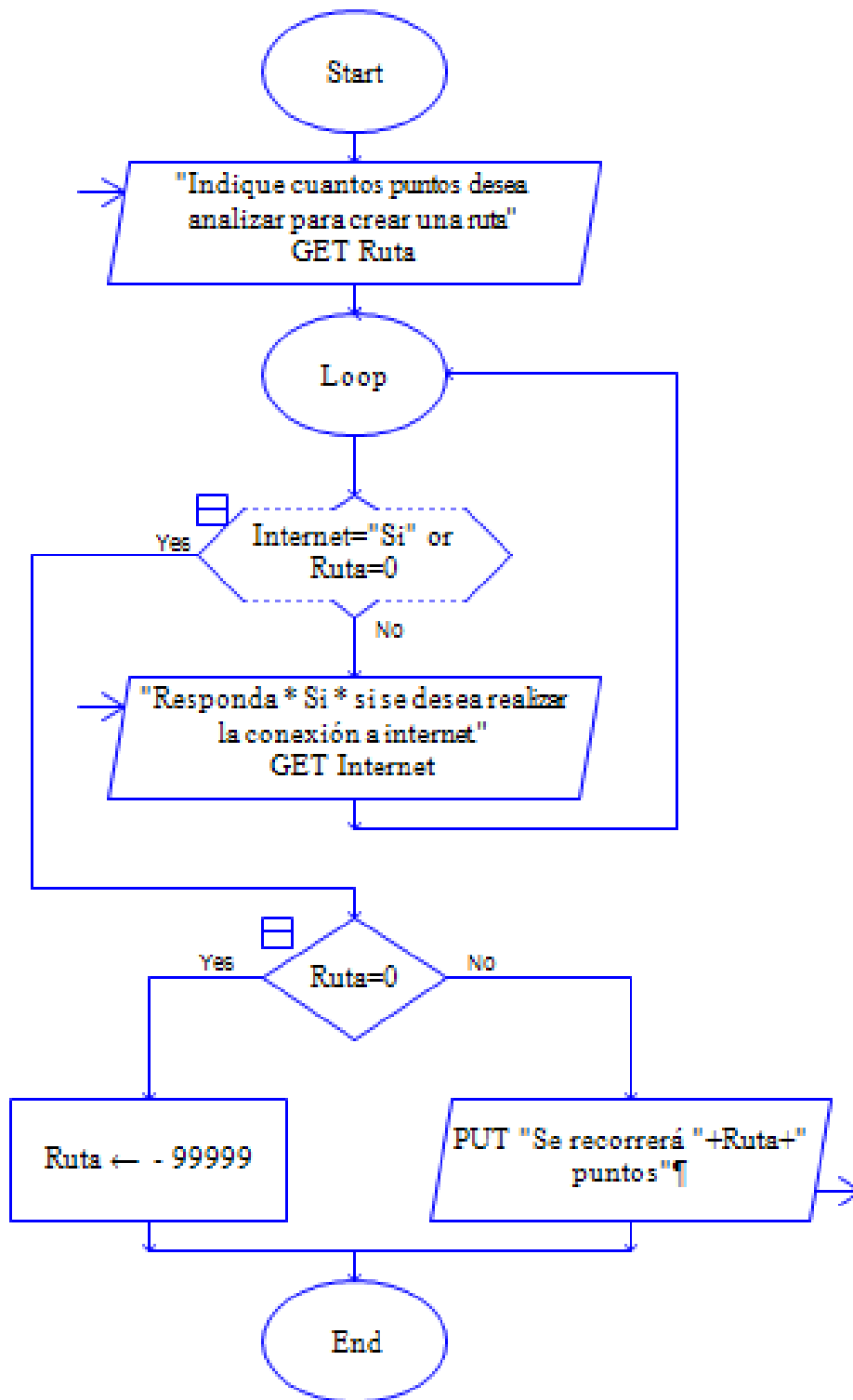


Fig. 45 Diagrama de flujo del funcionamiento del control automático

Al igual que en modo manual, se deberá tener buena visibilidad GPS y estar bien calibrado. La navegación autónoma se realiza a través de la plataforma de Mission Planner y se realiza de la siguiente manera:

Se verifica que el dispositivo este en modo COM y con numeración o tarifa como se denomina aquí 57600 que es una indicación que se va a trabajar en el simulador del Mission Planner.

Una vez conectado, mostrará un mapa y punto de la geolocalización que se encuentra en ese momento, a ese punto que se grafica se denomina home.

Luego se ingresa a la sección Flight Plan y deberás comenzar a diseñar la ruta haciendo clic con el mouse del ordenador logrando graficar puntos donde se desea que avance el robot, los mismo que se mostrarán en la parte inferior como se muestra en la **Fig. 46**.



Fig. 46 Pantalla del Mission Planner

Una vez trazada la ruta, se compila haciendo clic en Write y se ingresa en la sección Data que se encuentra en la parte superior derecha; en esta sección, se selecciona en

Action y se configura cambiando sus parámetros a Mission Start, su estado a forma manual, se arma el robot y se ejecuta con Do Action, como se muestra en **Fig. 47**.

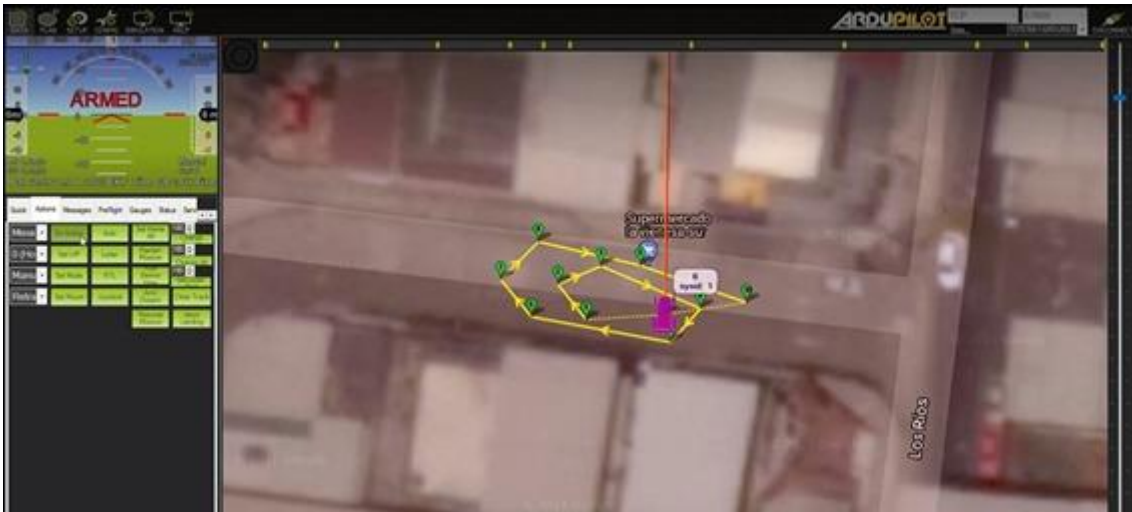


Fig. 47 Pantalla del Mission Planner

De la misma forma para realizar las simulaciones se verifica que el dispositivo este en modo TCP y con numeración o tarifa como se denomina aquí 57600.

Luego, se ingresa a la sección de simulación para seleccionar el tipo de prototipo que se está utilizando, en este caso un Rover. Una vez que se selecciona el tipo de prototipo y se carga la orden que se manda, se procede a verificar en la esquina superior del ordenador que se esté conectado, y a su vez se abre una pantalla que mostrará la telemetría enviada por ArduPilot en el cual muestra el mapa y como punto de la geolocalización que se encuentra en ese momento, a ese punto que se grafica se denomina home.

Como siguiente punto, se ingresa a la sección de PLAN para comenzar a diseñar la ruta, con el mouse del ordenador se da clic en los puntos donde se desea que avance el robot y se compila haciendo clic en Write. Luego se ingresa en Data y en la parte derecha

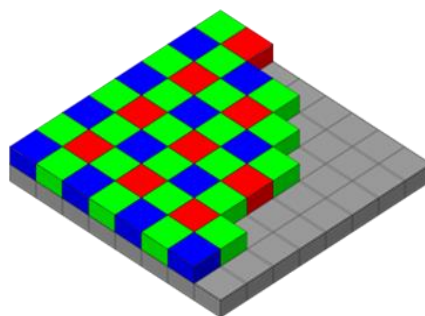
se visualizan unas opciones el cual se señala Action y Mission Start que es para iniciar la misión, se arma el robot y se ejecuta con Do Action.

### 3.5. VISIÓN ARTIFICIAL

Por medio de la utilización de distintas técnicas como: captación de imágenes, memorización de la información, procesado e interpretación de los resultados. Se permite la obtención, procesamiento y análisis de cualquier tipo de información especial obtenida mediante imágenes digitales, a esto se lo define “Visión Artificial”.

El punto de vista artificial la componen un conjunto de procesos destinados a hacer el análisis de imágenes. A esto se lo ofrece dentro del campo de la “Inteligencia Artificial”.

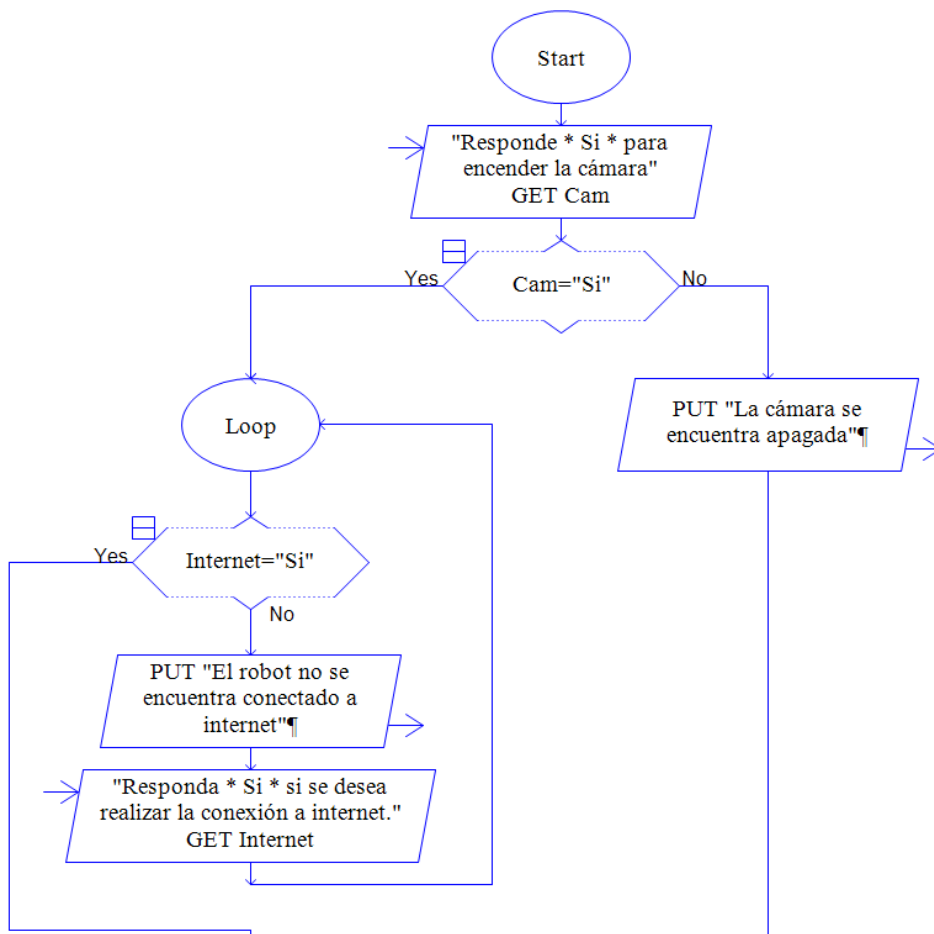
Mediante el cromo o clave de color se puede identificar objetivos de color verde, ya que se utiliza más que cualquier otro color, porque los sensores de imagen en cámaras de vídeo digitales son más sensibles al verde, debido al patrón de Bayer que asignan más píxeles en el canal verde, imitando aumento de la sensibilidad del ojo humano a la luz verde. Como se puede observar en **Fig. 48** está formado por un 50 % de filtros verdes, un 25 % de rojos y un 25 % de azules. Interpolando dos muestras verdes, una roja y una azul, se obtiene un pixel de color.



*Fig. 48 Mosaico de Bayer*

Por lo tanto, el canal de la cámara verde contiene un ruido menor y puede producir una máscara más limpia. Además, se necesita menos luz para iluminar el color verde, de nuevo debido a la mayor sensibilidad del verde en los sensores de imagen.

En la **Fig. 49** se realiza el algoritmo a actuar del robot con respecto a la detección del color verde.



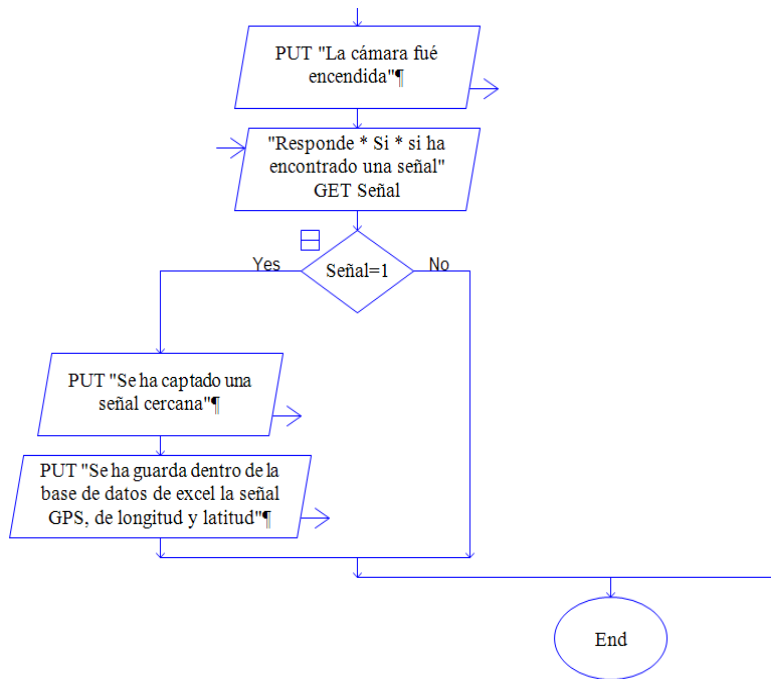


Fig. 49 Diagrama de flujo del funcionamiento de la visión artificial

La identificación de colores y determinación de la posición mediante el GPS, se lo realizó mediante código Python, como se puede visualizar en los siguientes códigos:

### 3.5.1. Vision.py

Es el programa principal a ejecutar, como se muestra en el *ANEXO A: REFERENCIA 2*, que se encargó de ver los objetivos marcados, en este caso esta calibrado para objetivos de color verde, ideal para señaléticas como: punto de encuentro, señales de salida y rescate.

### 3.5.2. gpsfinal.py

En este script como se muestra en el *ANEXO A: REFERENCIA 3*, vamos a leer los datos que llegan del GPS y vamos a generar un archivo Excel en formato .xlsx a su vez vamos a leer un pin GPIO que se activa cada vez que la cámara detecta un objetivo para saber su ubicación en el mapa.

### 3.5.3. GMAP.py

Este script de Python como se muestra en el *ANEXO A: REFERENCIA 4*, será el que lea el archivo Excel .xlsx para tener los puntos registrados cuando la visión artificial detecto los objetivos y envió a gps.py a escribir al archivo Excel las ubicaciones, luego el usuario podrá generar en pantalla con un botón el mapa con los puntos graficados como marcadores.

### 3.6.LIBRERÍAS:

Para realizarlo se necesitó contar con las siguientes librerías y sus comandos para la instalación desde terminal de Raspberry Pi:

#### CV2 – OPENCV

Librería usada para la visión artificial. Previo a la instalación de OpenCV en Python 3, necesitamos instalar algunos paquetes, esto lo haremos a través de la siguiente línea:

```
sudo apt-get install libhdf5-dev libhdf5-serial-dev  
libatlas-base-dev libjasper-dev libqtgui4 libqt4-test
```

Instalar con

```
pip3 OpenCV
```

cuando usamos pip3 los scripts que usen esta librería de OpenCV (CV2) deben ejecutarse bajo python3:

```
pip3 install OpenCV -contrib-python==4.1.0.25
```

#### NUMPY

Librería usada para los cálculos matemáticos como arreglos en CV2



Esta librería ya viene por defecto instalada en versiones de Python, no sería necesaria la instalación, pero el comando pip3 es:

```
pip3 install numpy
```

## **QUEUE**

Librería usada en multiprocesos para pasar argumentos de un hilo a otro.

```
pip install queuelib
```

## **THREADING**

Librería usada para crear multiprocesos en Python. Por lo general esta librería ya viene incluida en versiones de Python

## **XLSXWRITER**

Librería para escribir en archivos Excel en formato .xlsx

```
pip install XlsxWriter
```

y también

```
pip3 install XlsxWriter
```

## **PYNMEA2**

Librería para uso de GPS TINY

```
pip install pynmea2==1.8.0
```

y también

```
pip3 install pynmea2==1.8.0
```

## **GMPLLOT**

Librería para graficar mapas en Python

```
pip3 install gmpplot
```

## **OPENPYXL**

Librería para leer archivos Excel en todos sus formatos

```
Pip3 install openpyxl
```

## **WEBBROWSER**

Librería para abrir vínculos desde Python, por lo general ya viene instalada.

## **TKINTER**

Librería para abrir pantallas de ejecución en Python, por lo general ya vienen.

## **PIL**

Librería para ingresar imágenes a pantallas de ejecución de Python, se usa para agregar el logo de la universidad, por lo general ya vienen instaladas en las versiones de Python.

### **3.7. CREACIÓN DE CLAVE API KEY**

Para uso de Google maps con Python. En la página [www.console.cloud.google.com](http://www.console.cloud.google.com) ingresar con una cuenta Google y crear un nuevo proyecto, como se muestra en *Fig. 50*.

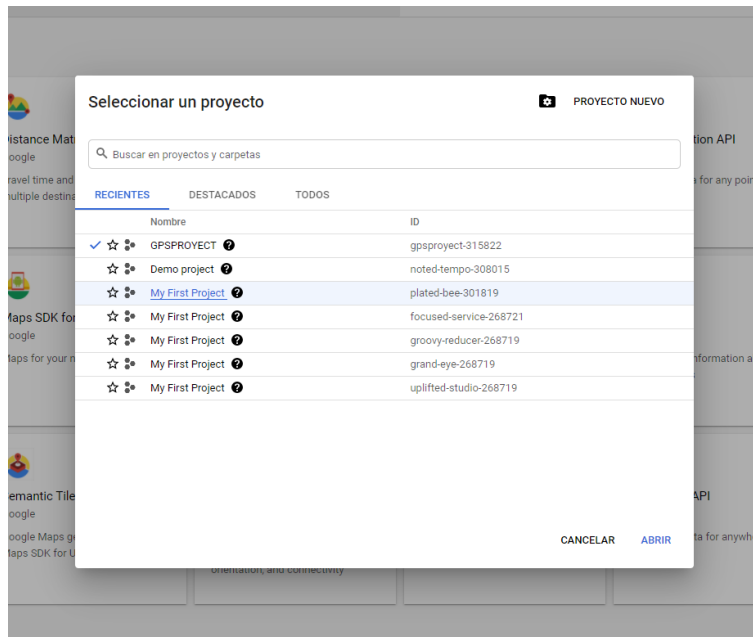


Fig. 50 Selección de proyecto dentro de la página de Google

Se nos mostrara el panel, donde vamos a ingresar a biblioteca en la parte izquierda de la pantalla, como se muestra en la **Fig. 51**.

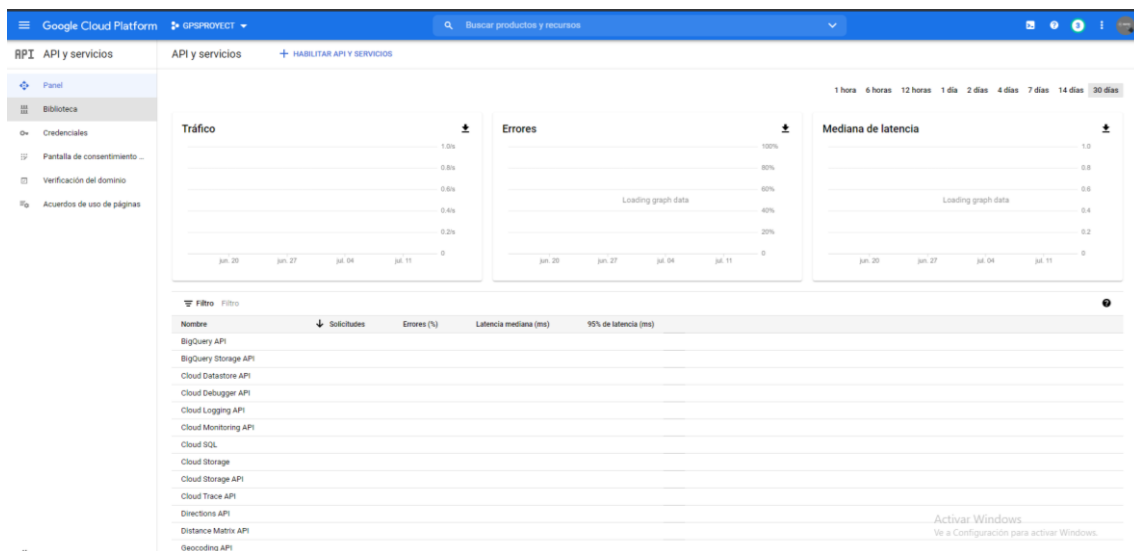


Fig. 51 Pantalla del Google Cloud Platform

Una vez ingresado a la biblioteca que es donde estarán todos los servicios de Google que un usuario desarrollador puede usar, vamos a instalar todas las bibliotecas de Mapas, esto se hace por que existen bibliotecas que funcionan cuando otras también están instaladas, para este caso es mejor tener todas instaladas para que no nos de error de conexión al momento de usar una API KEY desde Python. Como se muestra en las **Fig. 52** al **Fig. 54**.

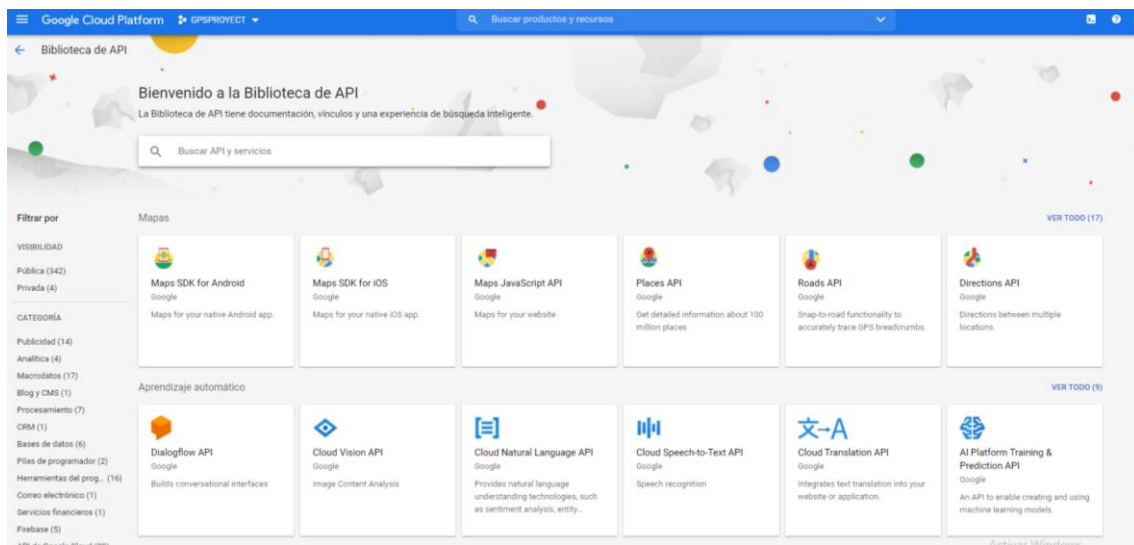


Fig. 52 Bibliotecas de API de Google

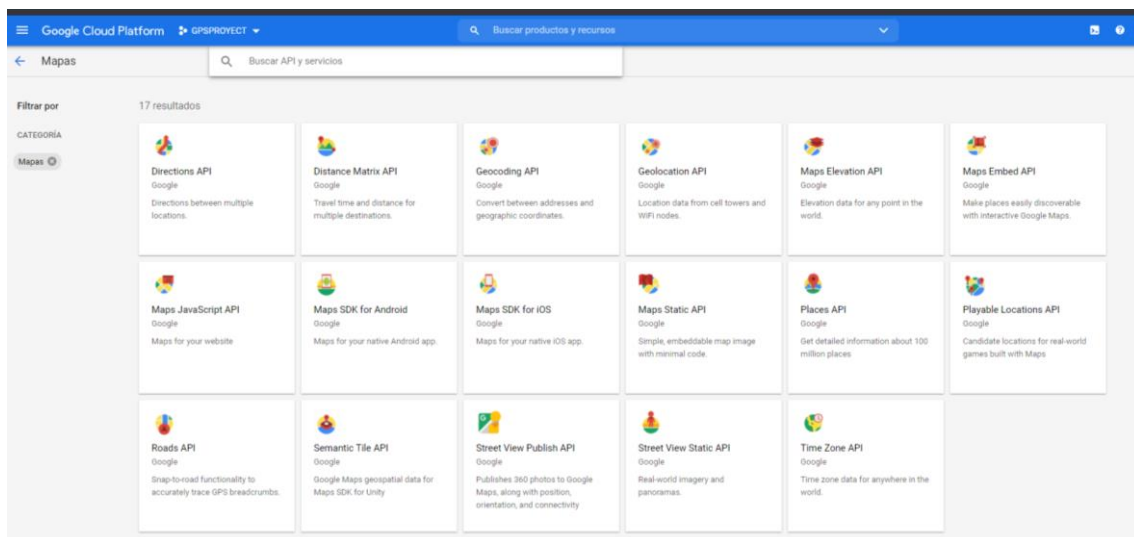


Fig. 53 Bibliotecas de API de Google

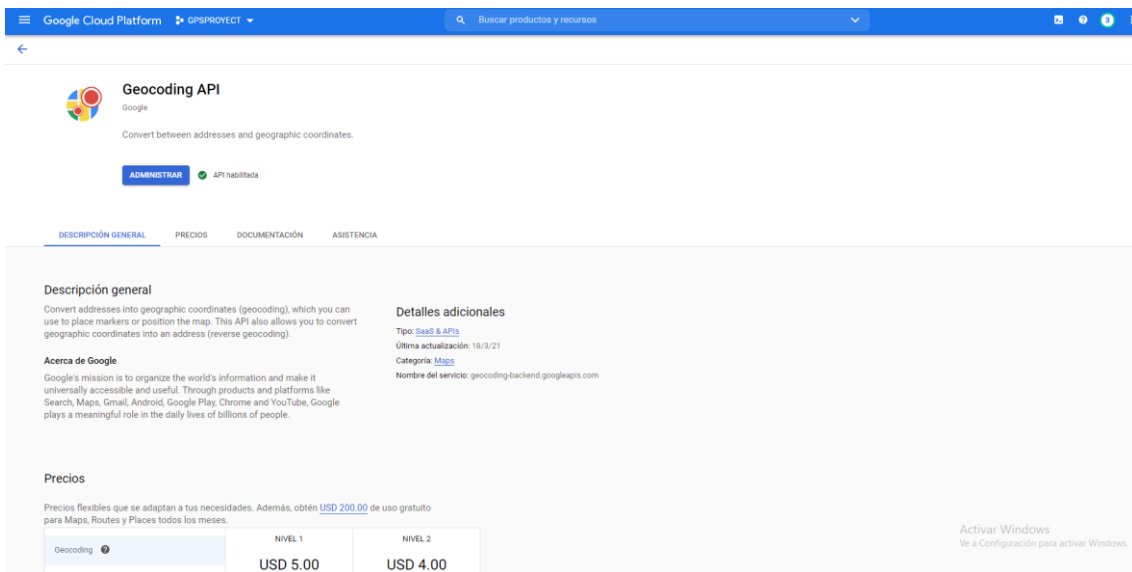


Fig. 54 Instalado producto de la biblioteca de Google

Una vez instaladas todas las bibliotecas de mapas, vamos a la ruta: menú de navegación → API y servicios → Credenciales. Como se muestra en **Fig. 55**.

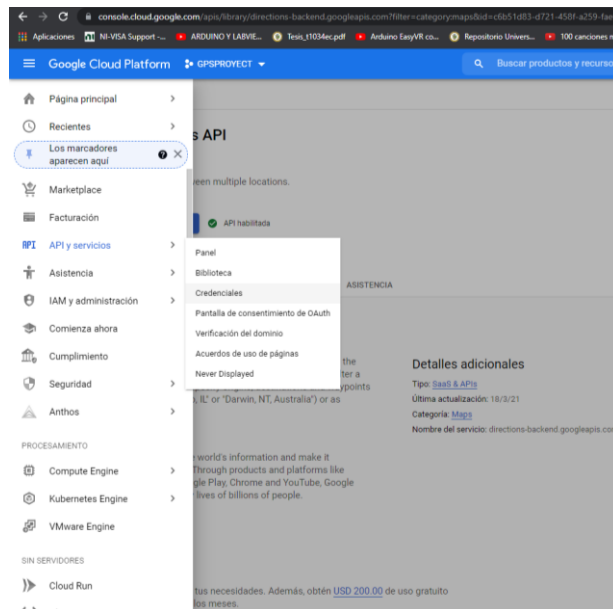


Fig. 55 Sección de credenciales de API

Vamos a crear una credencial y copiar el código. Para finalizar se debe ingresar los datos de facturación ya que usar estas bibliotecas no son gratis, pero Google da USD \$300 de crédito que por lo general se consumen en muy poco tiempo cuando son aplicaciones comerciales, para este caso solo se debe registrar la tarjeta de crédito o débito y prácticamente no se gastara nada. la ruta: menú de navegación - Facturación

Luego de eso se debe copiar la API KEY Generada y usarla en el script de Python. Como se muestra en **Fig. 56** hasta **Fig. 58**.

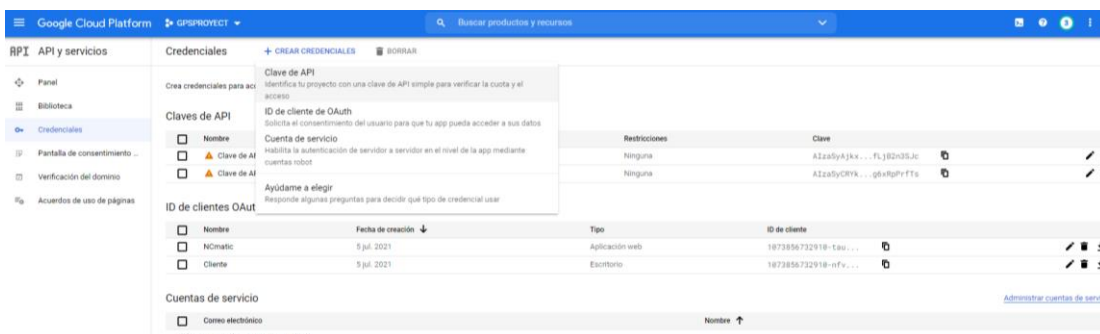


Fig. 56 Código de la credencial

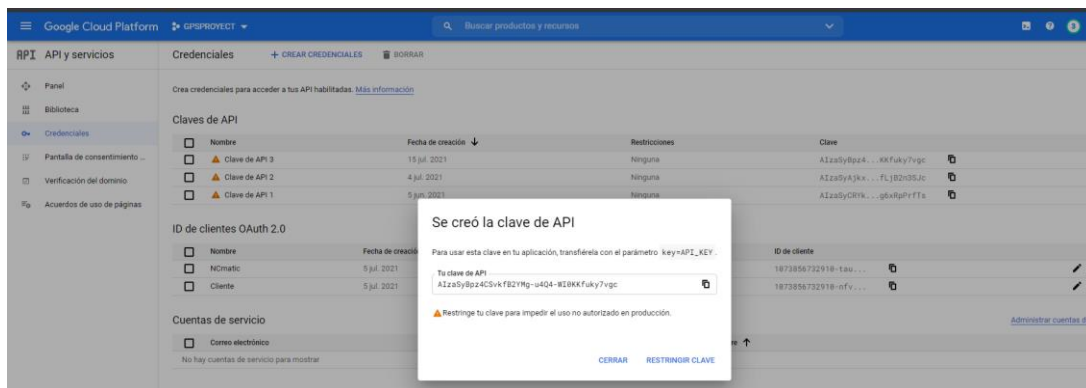


Fig. 57 Clave API

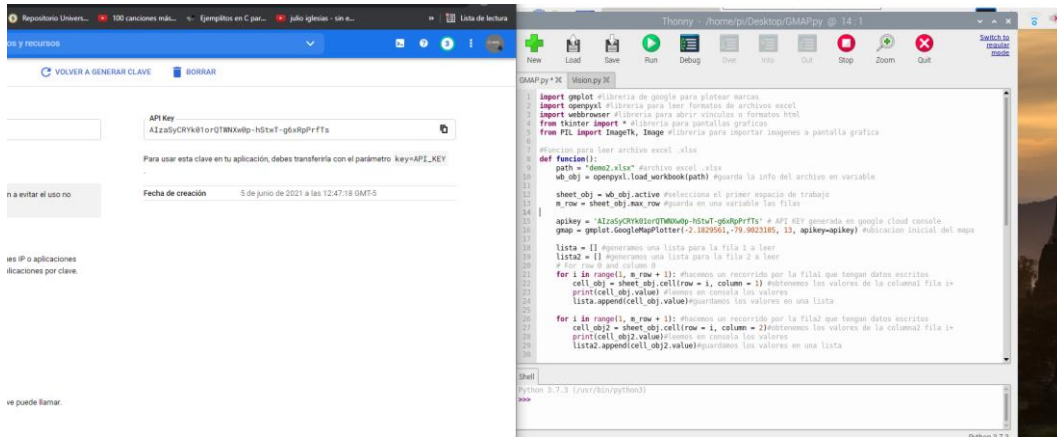


Fig. 58 Puesta de la clave en el código Python

### 3.8.COMANDOS E INSTRUCCIONES

Para ejecutar scripts del proyecto, primero arrancar desde Thony Python solo el script de GMAP.py desde botón run, aparecerá como se muestra en la **Fig. 59**.

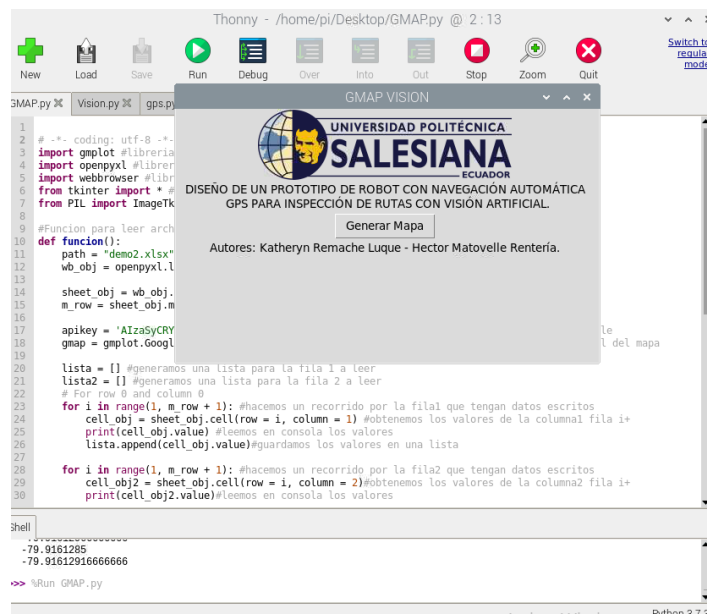


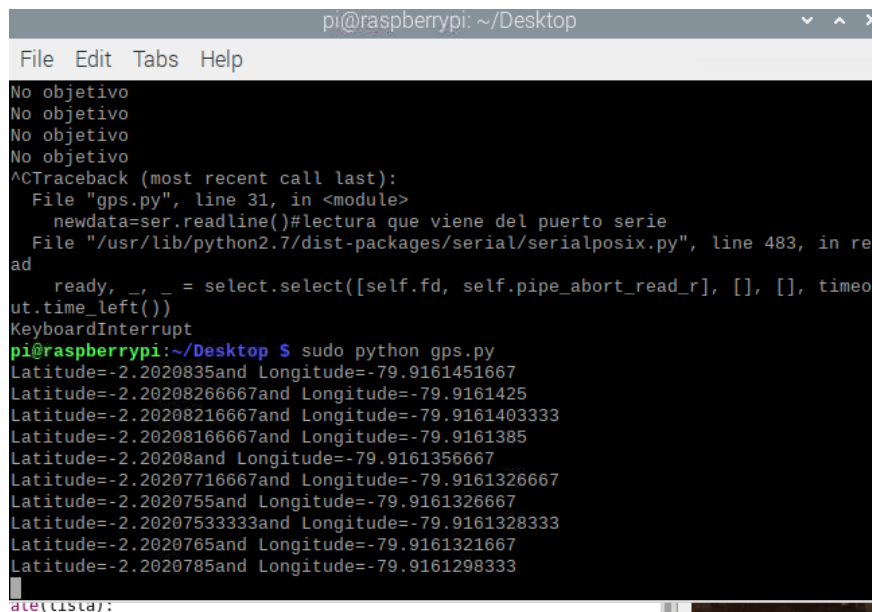
Fig. 59 Ventana de GUI para generar mapa

Abrir un terminal y ejecutar `> cd /home/pi/desktop` y clic en enter

Ubicados ya en la carpeta desktop, se deben ejecutar los scripts. En caso de cambiar la ubicación de los scripts se debe ejecutar la ubicación de estos con el comando cd para abrir la carpeta raíz del proyecto

Ejecutar > sudo Python gps.py

Aparecera como se muestra en **Fig. 60**.



```
pi@raspberrypi: ~/Desktop
File Edit Tabs Help
No objetivo
No objetivo
No objetivo
No objetivo
^CTraceback (most recent call last):
  File "gps.py", line 31, in <module>
    newdata=ser.readline()#lectura que viene del puerto serie
  File "/usr/lib/python2.7/dist-packages/serial/serialposix.py", line 483, in re
ad
    ready, __, __ = select.select([self.fd, self.pipe_abort_read_r], [], [], timeo
ut.time_left())
KeyboardInterrupt
pi@raspberrypi:~/Desktop $ sudo python gps.py
Latitude=-2.2020835and Longitude=-79.9161451667
Latitude=-2.20208266667and Longitude=-79.9161425
Latitude=-2.20208216667and Longitude=-79.9161403333
Latitude=-2.20208166667and Longitude=-79.9161385
Latitude=-2.20208and Longitude=-79.9161356667
Latitude=-2.20207716667and Longitude=-79.9161326667
Latitude=-2.2020755and Longitude=-79.9161326667
Latitude=-2.20207533333and Longitude=-79.9161328333
Latitude=-2.2020765and Longitude=-79.9161321667
Latitude=-2.2020785and Longitude=-79.9161298333
ale(tista):
```

Fig. 60 Ventana de terminal de comandos

Abrir otro terminal y ejecutar > sudo python3 Vision.py

Esto se lo hace de esta manera ya que hay librerías que son compatibles solo con python2 o con Python3, es por eso que se decidió trabajar con scripts diferentes y ejecutados desde entorno de programación Thony Python y desde terminales diferentes, como se observa en **Fig. 61**.



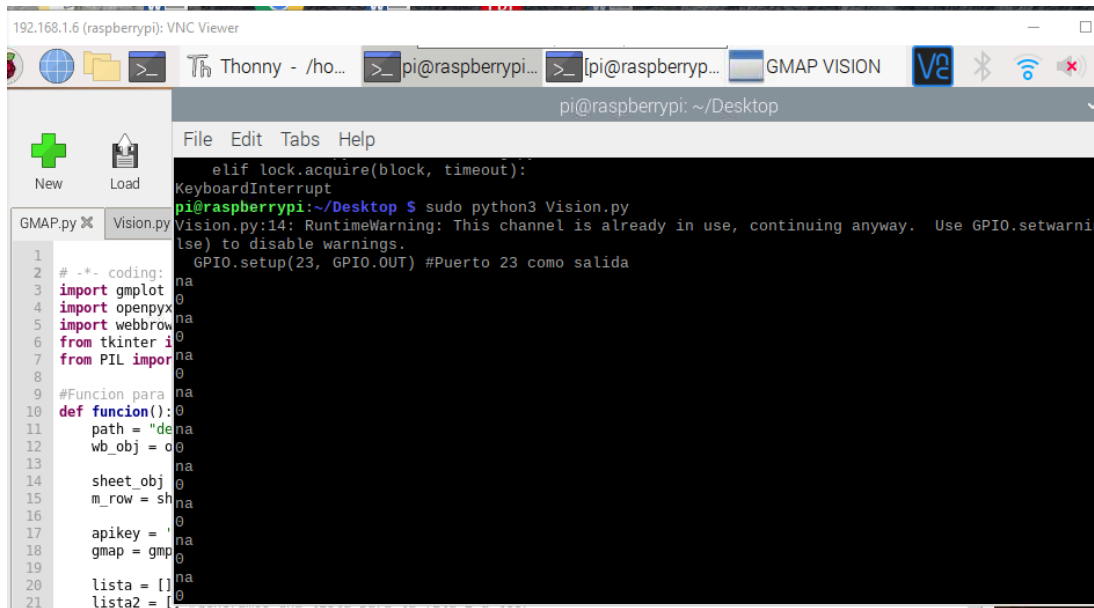


Fig. 61 Muestra de datos del terminal

## VERIFICACIÓN Y FUNCIONAMIENTO

### 4.1. RESULTADOS

En las pruebas realizadas en la U.E.S.” Domingo Comín” a las 17:02 p.m. se desarrolló una ruta de inspección autónoma vía GPS y usando el software Mission Planner se concluye que en un espacio abierto y en horario donde se tenga luz, el GPS tiene mayor precisión en recorrer la ruta trazada, como se observa en *Fig. 62*.



Fig. 62 Ruta recorrida en el bloque B1 de la Universidad Politécnica Salesiana

De igual manera, se verificó que es sensible en lugares donde tuvo perturbaciones como lugares cerrados, cercanías a lagos o ríos ya que se observa que tiene pequeñas variaciones o fuera de línea al momento que va ejecutando la ruta como se muestra en la *Fig. 63* cuya prueba se realizó en el bloque B de la Universidad Politécnica Salesiana.



Fig. 63 Trayectoria realizada dentro del bloque B de la Universidad Politécnica Salesiana

En la interacción con el medio físico y uso de los sensores ultrasónicos muestra un giro que se llevó a cabo con el ROV en donde se puede obtener el radio de giro interpolando las distancias reales y del papel. Por lo que se obtiene un radio de giro de aproximadamente 40 a 45 centímetros, como se muestra en *Fig. 64*.

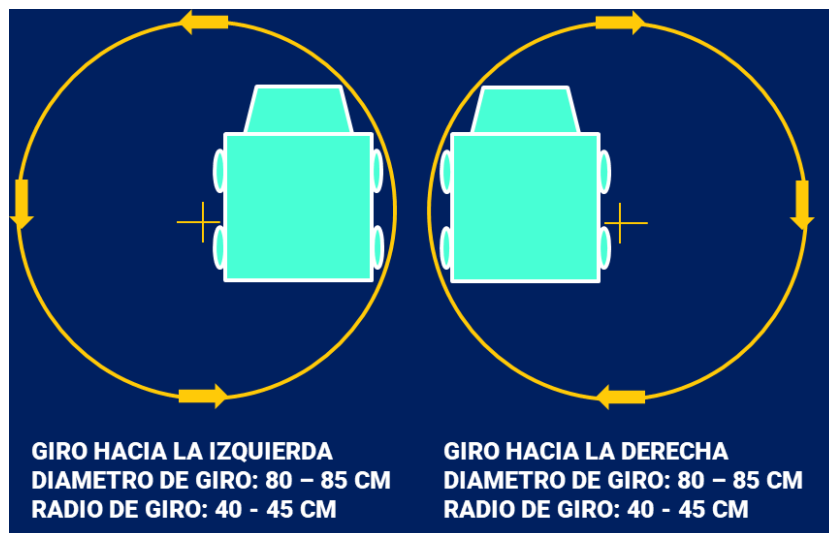


Fig. 64 Radio de giro del prototipo

Con respecto a la medición en la distancia entre el prototipo y el obstáculo para tener una respuesta en esquivar el objeto es de 55 a 60 centímetros, y teniendo un radio cuando detecta el objetivo es de 30 a 35 centímetros, como se muestra en *Fig. 65* y *Fig. 66*.

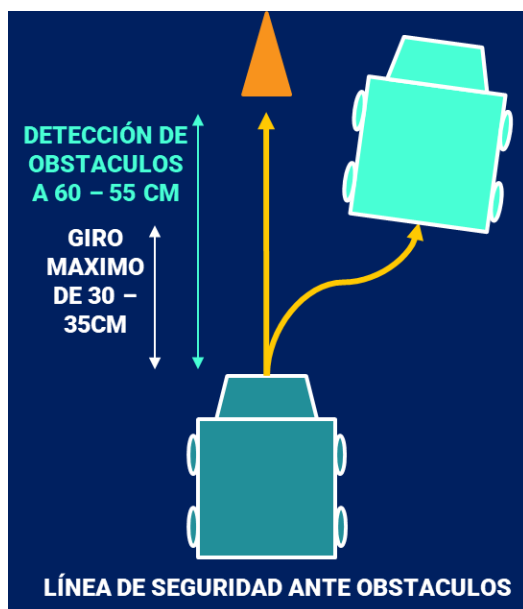


Fig. 65 Medidas de seguridad ante obstáculos



Fig. 66 Prueba de los sensores ultrasónicos

## 4.2. ANÁLISIS DEL RESULTADO

Este prototipo es un inicio y una apertura a proyectos a futuros para continuar evolucionándolo y desarrollando investigaciones aplicando el software Mission Planer en proyectos y competencias de robots por su variedad y funcionalidad en varios prototipos ya sea aéreos, terrestres o acuáticos por el gran impacto que causa y que fue comprobado en exposiciones *ANEXO C: Fig. 86 y Fig. 85* y según las encuestas realizadas *ANEXO C: Fig. 84*.

En la puesta en marcha de la placa programadora pixhawk se realizaron las pruebas de las mismas mediante la ejecución del prototipo con el radio control, comprobando la correcta comunicación entre el RC con el prototipo y la configuración de la pixhawk con los Monster shield que permite el funcionamiento de los motores, como se muestra en *Fig. 67*.



Fig. 67 Puesta en marcha del robot

En la construcción del robot todo terreno para la inspección de lugares complicados se utilizaron materiales como lata para su base logrando que cubra de mejor manera los

elementos y componentes que tiene en su interior. De igual manera, tiene una impresión 3D para su cubierta externa el cual permite que partículas o gotas de agua que se enfrente en su exterior no entre y afecte a los equipos internos. Otra característica son sus llantas con pequeños picos que permiten tener mayor tracción y facilidad para recorrer terrenos irregulares, como se muestra en *Fig. 68*.



Fig. 68 Robot recorriendo camino de piedra con desniveles

En la programación de visión artificial se utilizó la raspberry pi y se programó en Python para la detección de objetos de color verde usando librerías como cv2, numpy, entre otras (ver librerías en anexo). Estas imágenes, son captadas mediante una cámara y reflejas en el ordenador del usuario, mediante el código que se muestra en *Fig. 69*.

```
Thonny - /home/pi/Desktop/Vision.py @ 135:1
New Load Save Run Debug Over Into Out Stop Zoom
Vision.py * x
1 import cv2 #Libreria de open CV
2 import numpy as np #Libreria para calculos numericos como arreglos
3 import RPi.GPIO as GPIO #Libreria para puertos de IN OUT
4 import time #Libreria de delay y timers
5 from queue import Queue #Lib de colas para intercambiar info entre subprocessos
6 from threading import Thread #Lib de ejecucion de hilos
7
8 x = 0 #Variable x en la pantalla
9 w = 0 #Variable y en la pantalla
10 dist = 0 #Distancia calculada
11 b = 0
12
13 GPIO.setmode(GPIO.BCM) #Configuracion de GPIO
14 GPIO.setup(23, GPIO.OUT) #Puerto 23 como salida
15 captura = cv2.VideoCapture(0) #encerramos en variable los valores
16 #capturados de la camara,arreglos en 2 dimensiones.
17
```

Fig. 69 Código de visión en lenguaje Python

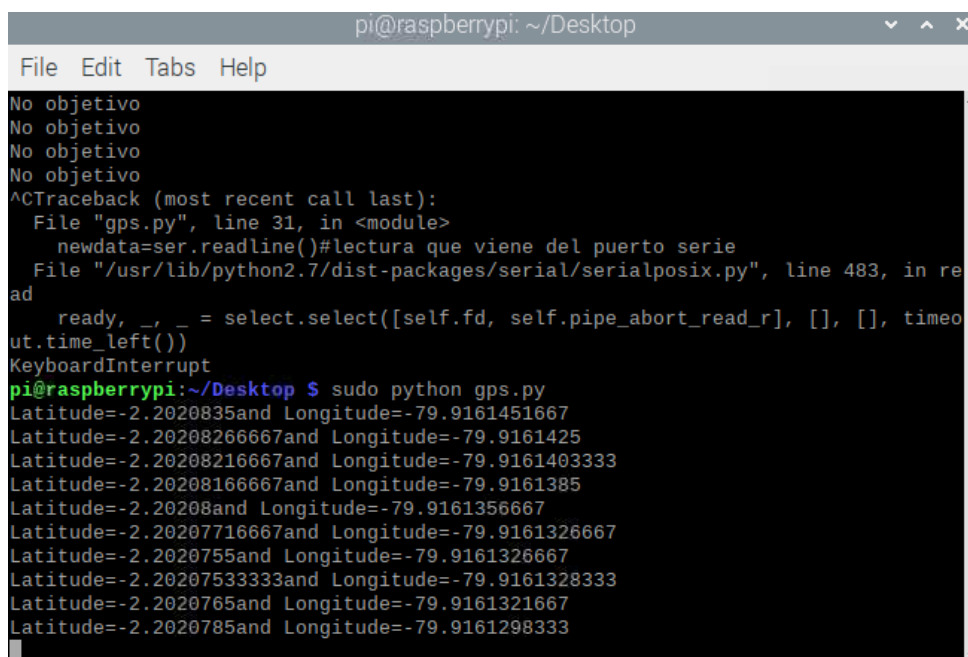
Otra característica que se obtiene y se debe tener en cuenta es cuando existen dos puntos verdes cercanos debido a que este automáticamente traza un área y se enfoca o posiciona en el centro del mismo. De igual manera, cuando se tiene en diferentes distancias una señalética de color verde, el prototipo capta y se posiciona en el objetivo más cercano, como se muestra en *Fig. 70*.



Fig. 70 Detección de señalética de Salida de Emergencia

Otra programación que se desarrolló en Python fue el script gps y gmap el cual permite leer los datos que llegan del GPS y mostrarlos en el terminal de la raspberry como latitud y longitud; de igual manera, como se muestra en **Fig. 71**, se procedió a generar un archivo Excel en formato .xlsx que permita llevar un registro y que se guarden dichas coordenadas cuando capte un objetivo verde.

De la misma forma, se leerá un pin GPIO que se activa cada vez que la cámara detecta un objetivo para saber su ubicación en el mapa.



```
pi@raspberrypi: ~/Desktop
File Edit Tabs Help
No objetivo
No objetivo
No objetivo
No objetivo
^CTraceback (most recent call last):
  File "gps.py", line 31, in <module>
    newdata=ser.readline()#lectura que viene del puerto serie
  File "/usr/lib/python2.7/dist-packages/serial/serialposix.py", line 483, in read
    ready, _, _ = select.select([self.fd, self.pipe_abort_read_r], [], [], timeo
ut.time_left())
KeyboardInterrupt
pi@raspberrypi:~/Desktop $ sudo python gps.py
Latitude=-2.2020835and Longitude=-79.9161451667
Latitude=-2.2020826667and Longitude=-79.9161425
Latitude=-2.20208216667and Longitude=-79.9161403333
Latitude=-2.20208166667and Longitude=-79.9161385
Latitude=-2.20208and Longitude=-79.9161356667
Latitude=-2.20207716667and Longitude=-79.9161326667
Latitude=-2.2020755and Longitude=-79.9161326667
Latitude=-2.20207533333and Longitude=-79.9161328333
Latitude=-2.2020765and Longitude=-79.9161321667
Latitude=-2.2020785and Longitude=-79.9161298333
```

Fig. 71 Terminal de comando de la raspberry pi

La verificación al interactuar con el medio físico a través de los sensores ultrasónicos se obtiene al momento de ejecutar una ruta ya sea por RC o de forma automática debido a que los mismo se mantienen censando cuando el prototipo se encuentra armado y ejecutando una ruta; esto se logra a la programación que se realiza en el Arduino cuyas



características establecidas por el usuario en este caso es de una línea de vida o distancia de respuesta al encontrarse con un obstáculo es de 30cm a 35 cm.

En el desarrollo de rutas de inspección autónoma GPS se realizan las misiones utilizando el prototipo y el software Mission Planner que permitirá ejecutar la ruta que el usuario desarrolle.

Para poner en marcha una misión, se debe de encender el ordenador donde se tenga instalado el software y realizar la conexión o enlace de la telemetría para que el prototipo y el ordenador o home estén comunicados, otro punto a considerar es tener un campo abierto, buena luz y en lo posible no tener perturbaciones para poner en marcha el prototipo, esto permitirá que tenga mayor precisión al seguir la ruta trazada. Como lo puede visualizar en la *Fig. 72*.



Fig. 72 Puesta en marcha del robot en modo automático

### **4.3.CONCLUSIONES**

Tras la realización de este proyecto se puede concluir que se ha demostrado la viabilidad técnica y funcional del diseño y montaje de un vehículo operado remotamente a partir del piloto automático Pixhawk cumpliendo con las especificaciones necesarias para la navegación tanto autónoma utilizando el Mission Planner, como manual utilizando el RC. Así mismo, tras muchas investigaciones se consiguió desarrollar las rutas y puesta en marcha de la placa programadora Pixhawk partiendo de su diagrama de conectores para la interacción con los sensores mediante el enlace funcional del Arduino para poder realizar las rutas y evitando obstáculos que se presenten en el camino.

De igual manera, el uso del software Mission Planer permitió una mayor flexibilidad, y mejor interactividad a la hora de trazar rutas y realizar un recorrido de forma automática. De la misma manera que la tecnología en radioenlaces ha crecido de manera significativa permite que la comunicación sea más sencilla entre módulos tanto para control como para gestión de datos, en este caso la conexión y comunicación que existe por la telemetría es la que aporta y logra hacer esa interactividad ente el home y el prototipo. Se ha llegado a poner en funcionamiento a todo un sistema funcional de recogida de datos de sensores a partir de un ROV en diferentes terrenos, la aplicación de la visión artificial al momento de detectar objetivos de color verde y mostrar su posición gracias a la configuración del GPS que muestra la geolocalización actual en el terminal tanto en latitud como en longitud brindando una mayor precisión al usuario de donde captó el objetivo y una central para poder monitorizar o pilotar al ROV.

Este proyecto permitió ampliar los conocimientos adquiridos en la carrera de ingeniería electrónica, de estudios y concursos de robótica, de aplicación y manejo en desarrollo de sensores y programación, el cómo actúa un vehículo a radiocontrol y cómo se puede diseñar y hacer funcionar con un piloto automático, y sobre todo la realización al momento de detectar, leer, capturar y guardar datos de un objetivo al usar la visión artificial que se logró llevar todo a la práctica, lo cual ha aportado a la demostración y verificación de esos conocimientos plasmado en la desarrollo y cumplimiento de cada punto de los objetivos en la puesta en marcha de nuestro prototipo.

#### **4.4.RECOMENDACIONES**

Como futuras aplicaciones de este proyecto se sugiere disponer de un GPS militar, el cual permite tener una trilateración; es decir, que la información que se envían sea por tres satélites aledaños y no por uno como es el GPS civil que se está usando, esto permitirá minimizar el error informativo que se produce en el receptor considerando la diferencia entre los tiempos reales de GPS y el reloj interno del mismo.

De igual manera, en la elección y manipulación de las baterías que dan la alimentación al prototipo se debe tener en cuenta y realizar un análisis previo para adquirir en lo posible una sola batería que alimente a todo el prototipo ya que esto permitirá la reducción de espacio de manera interna y que el prototipo pueda llegar a escalas más pequeñas.

La manipulación de las baterías es muy importante ya que se debe tener cuidado en no descargarlas por completo sino hasta un máximo de 50% y a su vez que al cargarlas se mantenga balanceadas las celdas haciendo que el cuidado de las baterías mantenga la vida útil de las mismas.

En el desarrollo o puesta en marcha de las misiones es importante estar en un espacio abierto, sin muchos edificios, construcciones o ríos cercanos ya que esto provoca perturbaciones al momento de la ejecución correcta del prototipo y GPS.

## REFERENCIAS BIBLIOGRÁFICAS

- [1] A. Ocles Padilla, «Gestionderiesgos.gob.ec,» 19 Junio 2020. [En línea]. Available: <https://www.gestionderiesgos.gob.ec/plan-de-respuesta/>. [Último acceso: 24 Febrero 2021].
- [2] L. . M. Rodríguez Flores y C. G. Marin Molina, «Repositorio.ug.edu.ec,» 20 Diciembre 2017. [En línea]. Available: <http://repositorio.ug.edu.ec/bitstream/redug/24306/1/B-CINT-PTG-N.233.Rodr%C3%ADguez%20Flores%20Leslie%20Michelle.Marin%20Molina%20Christian%20Guillermo.pdf>. [Último acceso: 24 Febrero 2021].
- [3] . J. Woodfill , «Er.jsc.nasa.gov,» 25 Marzo 2011. [En línea]. Available: [https://er.jsc.nasa.gov/seh/robots\\_in\\_space.htm](https://er.jsc.nasa.gov/seh/robots_in_space.htm). [Último acceso: 24 Febrero 2021].
- [4] G. Pino, «Instituto Geofísico - Escuela Politécnica Nacional,» 16 Abril 2020. [En línea]. Available: <https://www.igepn.edu.ec/interactuamos-con-usted/1810-cuatro-anos-despues-del-terremoto-de-pedernales-un-testimonio-sobre-el-peligro-sismico-en-el-ecuador>. [Último acceso: 16 Febrero 2020].
- [5] «Rambal,» [En línea]. Available: <https://rambal.com/motor-reductorgearbox/294-motor-dc-engranaje-alto-torque-3-24vdc-108rpm.html>. [Último acceso: 10 10 2021].
- [6] «Dhgate,» [En línea]. Available: <https://es.dhgate.com/product/small-smart-car-model-robot-rubber-tire-wheel/584610166.html>. [Último acceso: 10 10 2021].
- [7] «Elec3d,» [En línea]. Available: <https://elec3d.com/motores/433-monster-shield-30amp-vnh3asp30.html>. [Último acceso: 10 10 2021].
- [8] «Ja-bots,» [En línea]. Available: <https://ja-bots.com/producto/shield-de-expansion-arduino-nano/>. [Último acceso: 10 10 2021].
- [9] «Store usa arduino,» [En línea]. Available: <https://store-usa.arduino.cc/products/arduino-nano?selectedStore=us>. [Último acceso: 10 10 2021].
- [10] «Naylampmechatronics,» [En línea]. Available: <https://naylampmechatronics.com/sensores-proximidad/10-sensor-ultrasonido-hc-sr04.html>. [Último acceso: 10 10 2021].

- [11] L. Rodríguez León, «Implementación de un Sistema de Control de Vuelo Automático para un Vehículo,» Cartago, 2015.
- [12] «ArduPilot,» [En línea]. Available: <https://ArduPilot.org/copter/docs/common-buzzer.html>. [Último acceso: 10 10 2021].
- [13] «Grupo Electro Store,» [En línea]. Available: <https://grupoelectrostore.com/shop/modulos-y-shields/reguladores-de-voltaje/modulo-de-poder-2-8-2-6-bec-5v-3a-para-pixhawk/>. [Último acceso: 10 10 2021].
- [14] «Robu,» [En línea]. Available: <https://robu.in/product/pixhawk-i2c-port-expand-board-cable/>. [Último acceso: 10 10 2021].
- [15] E. J. Rabuffetti, «Diseño y construcción de un ROV para aplicaciones de sensorización medioambiental,» 2016.
- [16] «Idcmayoristas,» [En línea]. Available: <https://www.idcmayoristas.com/productos/camara-de-video-conferencia-genius-1000x-hd-720p-usb/#:~:text=Conexi%C3%B3n%20en%20720p%20HD%20en,con%20v%C3%ADdeo%20de%20alta%20definici%C3%B3n>. [Último acceso: 10 10 2021].
- [17] «Mvelectronica,» [En línea]. Available: <https://mvelectronica.com/products/7637>. [Último acceso: 10 10 2021].
- [18] «Shopee,» [En línea]. Available: <https://shopee.com.mx/-white-conector-xt60-con-carcasa-de-funda-hembra-conector-xt60-macho-i.414821508.11006109735>. [Último acceso: 10 10 2021].
- [19] «Mercado libre,» [En línea]. Available: [https://articulo.mercadolibre.com.ec/MEC-501095887-switch-3pdt-de-9-pines-true-bypass-para-efectos-de-guitarra-\\_JM#redirectedFromParent](https://articulo.mercadolibre.com.ec/MEC-501095887-switch-3pdt-de-9-pines-true-bypass-para-efectos-de-guitarra-_JM#redirectedFromParent). [Último acceso: 10 10 2021].
- [20] «ArduPilot,» [En línea]. Available: <https://ArduPilot.org/dev/docs/rover-sitlmavproxy-tutorial.html>. [Último acceso: 10 10 2021].
- [21] «ArduPilot,» [En línea]. Available: <https://ArduPilot.org/planner/docs/mission-planner-overview.html>. [Último acceso: 10 10 2021].
- [22] «AArduPilot,» [En línea]. Available: <https://ArduPilot.org/planner/>. [Último acceso: 10 10 2021].
- [23] E. J. Rabuffetti, «UNIVERSIDAD POLITECNICA DE CATALUÑA,» 27 JUNIO 2016. [En línea]. Available:

<https://upcommons.upc.edu/bitstream/handle/2117/88782/memoria.pdf?sequence=1&isAllowed=y>. [Último acceso: 10 10 2021].

- [24] Electronicoscaldas, «Electronicoscaldas,» SEPTEMBER 2012. [En línea]. Available: <https://www.electronicoscaldas.com/datasheet/HC-SR04.pdf>. [Último acceso: 10 10 2021].
- [25] Alldatasheet, «Alldatasheet,» SEPTEMBER 2016. [En línea]. Available: <https://www.alldatasheet.es/datasheet-pdf/pdf/100179/STMICROELECTRONICS/VNH2SP30.htm>. [Último acceso: 10 10 2021].
- [26] Grupo Electrostore, «Grupoelectrostore.com,» 4 Marzo 2020. [En línea]. Available: [https://grupoelectrostore.com/shop/radio-controles/flysky/rc-radio-control-flysky-fs-i6x-hasta-10-canales-24ghz-con-receptor-fs-ia6b/](https://grupoelectrostore.com/shop/radio-contrroles/flysky/rc-radio-control-flysky-fs-i6x-hasta-10-canales-24ghz-con-receptor-fs-ia6b/). [Último acceso: 24 Febrero 2021].

## ANEXOS A: CÓDIGOS DE PROGRAMACIÓN

### REFERENCIA 1: CÓDIGO DE LOS SENSORES ULTRASÓNICOS-

#### ARDUINO

```
#include <Ultrasonic.h>

Ultrasonic ultrasonic1(2, 3);
Ultrasonic ultrasonic2(4, 5);
Ultrasonic ultrasonic3(7, 6);
// variables de posicion
int izq;
int der;
int cen;
int retro;
void setup() {
  Serial.begin(9600);
  pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(13, OUTPUT);
  pinMode(12, OUTPUT);
}

void loop() {
  retro = analogRead(A0);

  izq = ultrasonic1.read(CM); //Leemos la distancia del sonar
  cen = ultrasonic2.read(CM);
  der = ultrasonic3.read(CM);
  //Serial.println(retro);
  Serial.println(cen);
  Serial.println(izq);
  Serial.println(der);

  if(cen >60 && izq >60 && der >60 ){ //Sin obstaculos
    digitalWrite(9, HIGH);
    digitalWrite(8, HIGH);
    digitalWrite(10, HIGH);
    digitalWrite(11, HIGH);
  }

  if(cen >60 && izq >60 && der <60 ){ //Con obtaculos derechos
    digitalWrite(8, HIGH);
    digitalWrite(9, LOW);
    digitalWrite(10, LOW);
    digitalWrite(11, HIGH);
  }

  if(cen >60 && izq <60 && der >60 ){ //Con obstaculos izquierda
    digitalWrite(8, LOW);
    digitalWrite(9, HIGH);
  }
}
```



```

    digitalWrite(10, HIGH);
    digitalWrite(11, LOW);
}

if(cen <60 && izq <60 && der >60 ){ //Con obstaculos centro-
izquierda
    digitalWrite(9, HIGH);
    digitalWrite(10, HIGH);
    digitalWrite(8, LOW);
    digitalWrite(11, LOW);
}

if(cen <60 && izq >60 && der <60 ){ //Con obstaculos centro-derecha
    digitalWrite(8, HIGH);
    digitalWrite(11, HIGH);
    digitalWrite(9, LOW);
    digitalWrite(10, LOW);
}

if(cen <60 && izq >60 && der >60 ){ //Con obstaculos centro
    digitalWrite(8, HIGH);
    digitalWrite(11, HIGH);
    digitalWrite(9, LOW);
    digitalWrite(10, LOW);
}

if(cen >60 && izq <60 && der <60 ){ //Con obstaculos centro
    digitalWrite(8, HIGH);
    digitalWrite(11, HIGH);
    digitalWrite(9, HIGH);
    digitalWrite(10, HIGH);
}

if(retro > 50){ //NO TOCAR
    digitalWrite(13, HIGH);
    digitalWrite(12, HIGH);
    digitalWrite(10, LOW);
    digitalWrite(11, LOW);
}
else{
    digitalWrite(13, LOW);
    digitalWrite(12, LOW);
}
}
}

```

## REFERENCIA 2: CÓDIGO DEL SCRIPT VISION – RASPBERRY

```
import cv2 #Librería de OpenCV
import numpy as np #Librería para calculos numéricos como arreglos
import RPi.GPIO as GPIO #librería para puertos de IN OUT
import time #Librería de delay y timers
from queue import Queue #Librería de colas para intercambiar
información entre subprocesos
from threading import Thread #Librería de ejecución de hilos

x = 0 #Variable x en pantalla
w = 0 #Variable y en pantalla
dist = 0 #Distancia calculada
b = 0

GPIO.setmode(GPIO.BCM) #Configuración de GPIO
GPIO.setup(23, GPIO.OUT) #Puerto 23 como salida
captura = cv2.VideoCapture(0) #Encerramos en variable los valores
#capturados de la cámara, arreglos en 2 dimensiones

def loop_1(out_q): #Función de hilo 1 de la cámara
    #out_q como variable a compartir entre hilos

    while(captura.isOpened()): #Mientras la cámara este encendida

        #Capturamos una imagen y la convertimos de RGB -> HSV
        ret, imagen = captura.read()
        #ret es un valor booleano que retorna del marco de la cámara
        #imagen obtiene el siguiente cuadro en la cámara

        if (ret):

            hsv = cv2.cvtColor(imagen, cv2.COLOR_BGR2HSV) #convierte
la imagen a B/N
            #para un mejor procesamiento
            h,w,m=imagen.shape #dimensionamiento de la imagen que
permite obtener coordenadas

            #En este escanea color verde en formato HSV
            bajos = np.array([35,50,50], dtype=np.uint8)
            altos = np.array([90,255,255], dtype=np.uint8)

            #Crear una máscara con solo los pixeles dentro del rango
            mask = cv2.inRange(hsv,bajos,altos)
            #encontrar el área de los objetos que detecta la cámara
            moments = cv2.moments(mask)
            area =int( moments['m00'])

            x=0
            y=0
            dist=0

            #Dato que se comparte al siguiente hilo out q.put(a)
            if(area < 20000): #Se envia valor a = 0
                print("na")#100000
```

```

a = 0
out_q.put(a)

#print ("no detecta objetos")
if(area > 20000):#100000
#Buscamos el centro x, y del objeto
x = int(moments['m10']/moments['m00'])
y = int(moments['m01']/moments['m00'])

#Se realiza cálculos para definir la distancia
respecto al área
if area>=845695.7:
    if area>=2301333.5:
        dist=- (area/307781.175)+37.4771743
    if area<2301333.5:
        if area>=859737.4:
            dist=- (area/72079.83)+61.9278725
        if area<859737.4:
            dist=- (area/19652.8425)+93.74622211
    if int(dist)<=15:
        dist=15
    if int(dist)>=70:
        dist=70

#Coordenada en x
if (x-w/2)>50 and dist>40:
    pass
#Coordanada en y
if (x-w/2)< -50 and dist>40:
    pass

print((x-w/2))

a=0
#Si el objetivo esta entre x and y
if (x-w/2)<300 and (x-w/2)> -300 and int(dist)>40:
    #fprint("Objeto encontrado")
    a = 1
out_q.put(a)

#Dibuja objetivo en la pantalla donde esta el objetivo
cv2.rectangle(imagen, (x-5, y-5), (x+5, y+5),
(255,255,255), 18)
cv2.circle(imagen, (x,y), 3, (0,255,0),2)
cv2.line(imagen, (0,y), (w,y), (255,0,0),2)
cv2.line(imagen, (x,0), (x,h), (255,0,0),2)

#Muestra imagen en una pantalla
cv2.imshow('Camara',imagen)
tecla = cv2.waitKey(5) & 0xFF
if tecla == ord('q'): #para cerrar pantalla
    break
cv2.destroyAllWindows()

#Segundo hilo donde se comparte los valores booleanos

```

```

#que son de deteccion e objetivos
def loop_2(in_q):

    while True: #Como es un proceso dentro de un while
        #Se debe seguir usando un while en este hilo
        a = in_q.get() #variable donde entra el valor compartido entre
        #procesos
        print(a)
        if a == 1: #escribe el valor a un pin del GPIO
            #en el siguiente script se leera ese pin digital
            GPIO.output(23, GPIO.HIGH)
        if a == 0:
            GPIO.output(23, GPIO.LOW)

q = Queue() # declara variable a compartir
t1 = Thread(target = loop_1, args =(q, )) #con de thread con el nombre
de la
#funcion con argumento q
t2 = Thread(target = loop_2, args =(q, ))
t1.start() #indica el proceso 1 y 2
t2.start()

```

### REFERENCIA 3: CÓDIGO DEL SCRIPT GPSFNAL – RASPBERRY

```
#!/usr/bin/python
import xlswriter
import serial
import time
import string
import RPi.GPIO as GPIO
import os, time
import pynmea2

GPIO.setmode(GPIO.BCM) #Configuracion de GPIO y lectura de pin 24
GPIO.setup(24, GPIO.IN)
#from Vision import b

#os.system("sudo chmod 777 /dev/serial0")
lista = []
lista2 = []
row = 0

while True:
    #print(b)
    workbook = xlswriter.Workbook('demo2.xlsx') #Archivo para
    escritura
    worksheet = workbook.add_worksheet()
    try:
        port="/dev/serial0"
        ser=serial.Serial(port,baudrate=9600,timeout=0.5)
        dataout =pynmea2.NMEAStreamReader()
        newdata=ser.readline()

        if not GPIO.input(24):
            if newdata[0:6]=="$GPRMC":
                newmsg=pynmea2.parse(newdata)
                lat=newmsg.latitude
                lng=newmsg.longitude
                gps="Latitude=" +str(lat) + "and Longitude=" +str(lng)
                print(gps)
                lista.append(lat) #Crea una lista en base a los datos
                de latitud y longitud
                lista2.append(lng)
                for row_num, data in enumerate(lista):
                    worksheet.write(row_num, 0, data) #fila 0

                for row_num2, data2 in enumerate(lista2):
                    worksheet.write(row_num2, 1, data2)

                workbook.close()
            else:
                print("no objetivo")
    except Exception:
        print("pass")
    pass
```

## REFERENCIA 4: CÓDIGO DEL SCRIPT GMAP – RASPBERRY

```
import gmpplot #Libreria de google para plotear marcas
import openpyxl #Libreria para leer formatos de archivos excel
import webbrowser #Libreria para abrir vnculos o formatos html
from tkinter import * #Libreria para pantallas grficas
from PIL import ImageTk, Image #Libreria para importar imagenes a
pantalla grfica

#Funcin para leer archivo excel .xlsx
def funcion():
    path = "demo2.xlsx" #Archivo excel .xlsx
    wb_obj = openpyxl.load_workbook(path) #Guarda la info del archivo
    en variable

    sheet_obj = wb_obj.active #selecciona el primer espacio de trabajo
    m_row = sheet_obj.max_row #guarda en una variable las filas

    apikey = 'AIzaSyBpz4CSvkfB2YMg-u4Q4-WI0KKfuky7vgc' #API KEY
    generada en google cloud console
    gmap = gmpplot.GoogleMapPlotter(-2.1829561,-79.9023105, 13,
    apikey=apikey) #Ubicacin inicial del maps

    lista = [] #Generamos una lista para la fila 1 a leer
    lista2 = [] #Generamos una lista para la fila 2 a leer
    #For row 0 and 0 column 0
    for i in range(1, m_row + 1): #Hacemos un recorrido por la fila1
    que tengan datos escritos
        cell_obj = sheet_obj.cell(row = i, column = 1) #Obtenemos los
    valores de la columna1 fila i+
        print(cell_obj.value) #Leemos en consola los valores
        lista.append(cell_obj.value) #Guardamos los valores en una
    lista

    for i in range(1, m_row + 1): #Hacemos un recorrido por la fila2
    que tengan datos escritos
        cell_obj2 = sheet_obj.cell(row = i, column = 2) #Obtenemos los
    valores de la columna2 fila i+
        print(cell_obj2.value) #Leemos en consola los valores
        lista2.append(cell_obj2.value) #Guardamos los valores en una
    lista

    #El formtato para graficar puntos en google maps debe ser una
    tupla
    mytuple = tuple(lista) #Convertimos lista en tupla
    mytuple2 = tuple(lista2) #Convertimos lista2 en tupla

    #Graficamos con funcion scatter los puntos, lat y long obtenidos
    de las tuplas
    #poniendo marcador de color rojo y un tamao a visualizar
    gmap.scatter(mytuple , mytuple2, marker=True,color='red', size=12)
    #Dibuja el mapa en formato html y lo abre en un webbrowser
    gmap.draw('map.html')
    webbrowser.open_new_tab('map.html')
```

```

#Realizamos la pantalla de usuario
root = Tk()
root.title("GMAP VISION")
root.geometry('500x300+230+130')

#Agregamos la funcin de leer y generar el mapa con command
Button(root, text="Generar Mapa", command=funcion).grid(row =10,
column=2)

autores=""Autores: Katherine Remache Luque - Hector Matovelle
Renteria""
label = Label(root, justify=LEFT,padx = 10, text=autores).grid(row
=15, column= 2)

#img = ImageTK.PhotoImage(Image.open("image.png"))
#panel = Label(root, justify=CENTER,image =img).grid(row = 5, column=
5)

tema= ""DISEÑO DE UN PROTOTIPO DE ROBOT CON NAVEGACION AUTOMATICA
GPS PARA INSPECCION DE RUTAS CON VISION ARTIFICIAL.""
label = Label(root, justify=LEFT,padx = 10, text=tema).grid(row =7,
column= 2)

root.mainloop()

```

## ANEXOS B: DATASHEET DE EQUIPOS

### DATASHEET DEL SENSOR ULTRASÓNICO HC-SR04 [24]

Product User's Manual – HC-SR04 Ultrasonic Sensor

---



## User's Manual

V1.0

September 2012

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Cytron Technologies Incorporated with respect to the accuracy or use of such information or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Cytron Technologies's products as critical components in life support systems is not authorized except with express written approval by Cytron Technologies. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.



## Index

1. <a href="#">Introduction</a>	3
2. <a href="#">Product Layout</a>	4
3. <a href="#">Product Specification and Limitation</a>	5
4. <a href="#">Operation</a>	6

## 1.0 INTRODUCTION

The HC-SR04 ultrasonic sensor uses sonar to determine distance to an object like bats or dolphins do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package. From 2cm to 450cm or 1" to 16ft. It operation is not affected by sunlight or black material like Sharp rangefinders are (although acoustically soft materials like cloth can be difficult to detect). It comes complete with ultrasonic transmitter and receiver module.

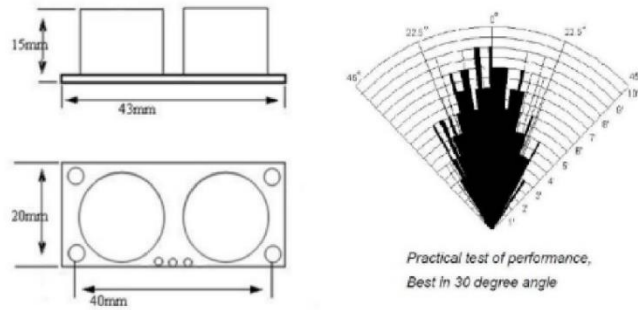
### Features:

- Power Supply :+5V DC
- Quiescent Current : <2mA
- Working Current: 15mA
- Effectual Angle: <15°
- Ranging Distance : 2cm – 400 cm / 1" - 13ft
- Resolution : 0.3 cm
- Measuring Angle: 30 degree
- Trigger Input Pulse width: 10uS
- Dimension: 45mm x 20mm x 15mm

## 2.0 PRODUCT LAYOUT



VCC = +5VDC  
Trig = Trigger input of Sensor  
Echo = Echo output of Sensor  
GND = GND



---

### 3.0 PRODUCT SPECIFICATION AND LIMITATIONS

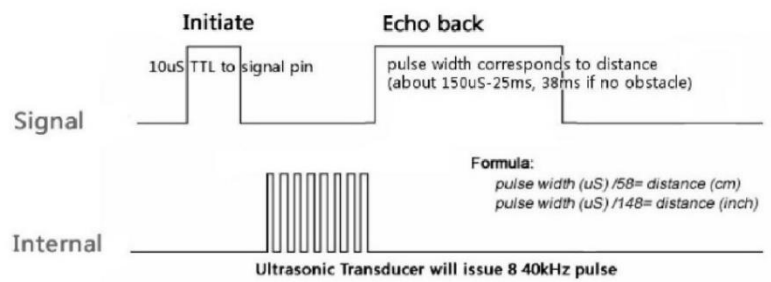
<b>Parameter</b>	<b>Min</b>	<b>Typ.</b>	<b>Max</b>	<b>Unit</b>
Operating Voltage	4.50	5.0	5.5	V
Quiescent Current	1.5	2	2.5	mA
Working Current	10	15	20	mA
Ultrasonic Frequency	-	40	-	kHz

#### 4.0 OPERATION

The timing diagram of HC-SR04 is shown. To start measurement, Trig of SR04 must receive a pulse of high (5V) for at least 10us, this will initiate the sensor will transmit out 8 cycle of ultrasonic burst at 40kHz and wait for the reflected ultrasonic burst. When the sensor detected ultrasonic from receiver, it will set the Echo pin to high (5V) and delay for a period (width) which proportion to distance. To obtain the distance, measure the width (Ton) of Echo pin.

Time = Width of Echo pulse, in uS (micro second)

- Distance in centimeters = Time / 58
- Distance in inches = Time / 148
- Or you can utilize the speed of sound, which is 340m/s



Note:

- Please connect the GND pin first before supplying power to VCC, otherwise it will affect the normal work of the module.
- Please make sure the surface of object to be detect should have at least 0.5 meter<sup>2</sup> for better performance.

# DATASHEET DEL MONSTER SHIEL 30 AMP VNH3ASP30 [25]



## VNH2SP30-E AUTOMOTIVE FULLY INTEGRATED H-BRIDGE MOTOR DRIVER

**Table 1. General Features**

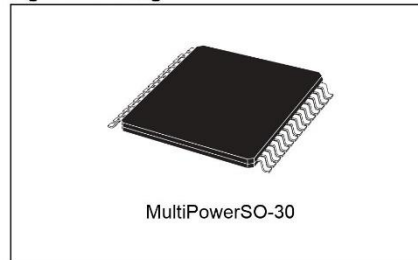
Type	$R_{DS(on)}$	$I_{out}$	$V_{CCmax}$
VNH2SP30-E	19 mΩ max (per leg)	30 A	41 V

- OUTPUT CURRENT: 30A
- 5V LOGIC LEVEL COMPATIBLE INPUTS
- UNDERVOLTAGE AND OVERVOLTAGE SHUT-DOWN
- OVERVOLTAGE CLAMP
- THERMAL SHUT DOWN
- CROSS-CONDUCTION PROTECTION
- LINEAR CURRENT LIMITER
- VERY LOW STAND-BY POWER CONSUMPTION
- PWM OPERATION UP TO 20 KHz
- PROTECTION AGAINST:
  - LOSS OF GROUND AND LOSS OF  $V_{CC}$
- CURRENT SENSE OUTPUT PROPORTIONAL TO MOTOR CURRENT
- IN COMPLIANCE WITH THE 2002/95/EC EUROPEAN DIRECTIVE

### DESCRIPTION

The VNH2SP30-E is a full bridge motor driver intended for a wide range of automotive applications. The device incorporates a dual monolithic High-Side drivers and two Low-Side switches. The High-Side driver switch is designed using STMicroelectronics well known and proven proprietary VIPower™ M0 technology that allows to efficiently integrate on the same die a true Power MOSFET with an intelligent signal/protection circuitry.

**Figure 1. Package**



The Low-Side switches are vertical MOSFETs manufactured using STMicroelectronics proprietary EHD ('STripFET™') process. The three dice are assembled in MultiPowerSO-30 package on electrically isolated leadframes. This package, specifically designed for the harsh automotive environment offers improved thermal performance thanks to exposed die pads. Moreover, its fully symmetrical mechanical design allows superior manufacturability at board level. The input signals  $IN_A$  and  $IN_B$  can directly interface to the microcontroller to select the motor direction and the brake condition. The  $DIAG_A/EN_A$  or  $DIAG_B/EN_B$ , when connected to an external pull-up resistor, enable one leg of the bridge. They also provide a feedback digital diagnostic signal. The normal condition operation is explained in the truth table on page 14. The CS pin allows to monitor the motor current by delivering a current proportional to its value. The PWM, up to 20KHz, lets us to control the speed of the motor in all possible conditions. In all cases, a low level state on the PWM pin will turn off both the  $LS_A$  and  $LS_B$  switches. When PWM rises to a high level,  $LS_A$  or  $LS_B$  turn on again depending on the input pin state.

**Table 2. Order Codes**

Package	Tube	Tape and Reel
MultiPowerSO-30	VNH2SP30-E	VNH2SP30TR-E

VNH2SP30-E

Figure 2. Block Diagram

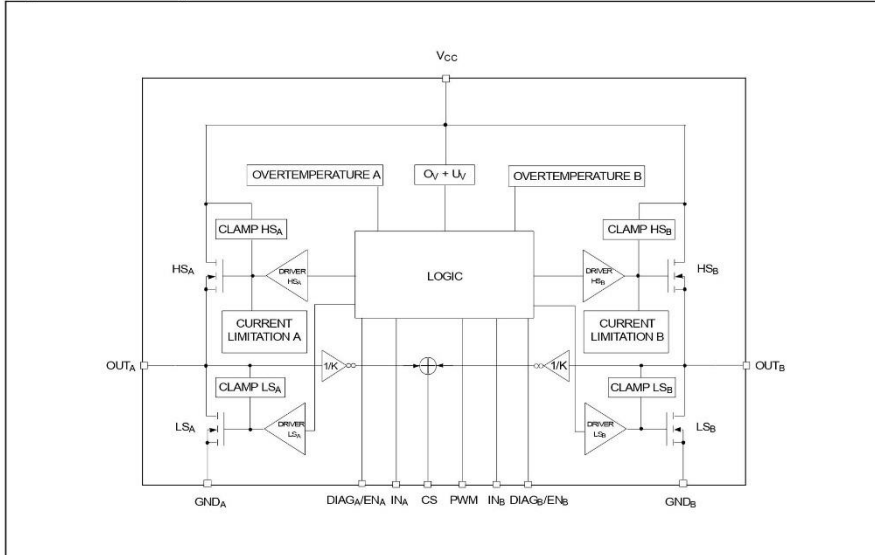
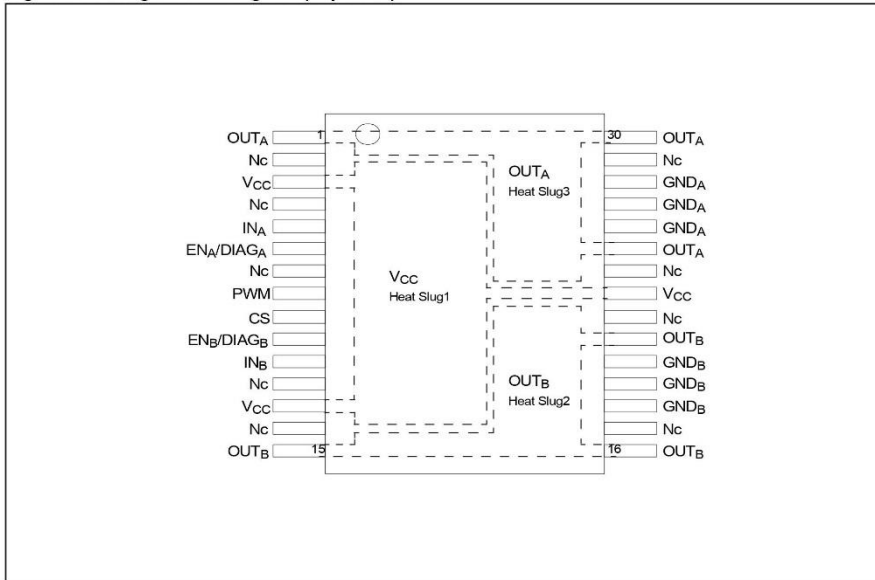


Figure 3. Configuration Diagram (Top View)



**Table 3. Pin Definitions And Functions**

Pin No	Symbol	Function
1, 25, 30	OUT <sub>A</sub> , Heat Slug2	Source of High-Side Switch A / Drain of Low-Side Switch A
2,4,7,12,14,17, 22, 24,29	NC	Not connected
3, 13, 23	VCC, Heat Slug1	Drain of High-Side Switches and Power Supply Voltage
6	EN <sub>A</sub> /DIAG <sub>A</sub>	Status of High-Side and Low-Side Switches A; Open Drain Output
5	IN <sub>A</sub>	Clockwise Input
8	PWM	PWM Input
9	CS	Output of Current sense
11	IN <sub>B</sub>	Counter Clockwise Input
10	EN <sub>B</sub> /DIAG <sub>B</sub>	Status of High-Side and Low-Side Switches B; Open Drain Output
15, 16, 21	OUT <sub>B</sub> , Heat Slug3	Source of High-Side Switch B / Drain of Low-Side Switch B
26, 27, 28	GND <sub>A</sub>	Source of Low-Side Switch A (*)
18, 19, 20	GND <sub>B</sub>	Source of Low-Side Switch B (*)

Note: (\*) GND<sub>A</sub> and GND<sub>B</sub> must be externally connected together

**Table 4. Pin Functions Description**

Name	Description
V <sub>CC</sub>	Battery connection.
GND <sub>A</sub> GND <sub>B</sub>	Power grounds, must always be externally connected together.
OUT <sub>A</sub> OUT <sub>B</sub>	Power connections to the motor.
IN <sub>A</sub> IN <sub>B</sub>	Voltage controlled input pins with hysteresis, CMOS compatible. These two pins control the state of the bridge in normal operation according to the truth table (brake to V <sub>CC</sub> , Brake to GND, clockwise and counterclockwise).
PWM	Voltage controlled input pin with hysteresis, CMOS compatible. Gates of Low-Side FETS get modulated by the PWM signal during their ON phase allowing speed control of the motor
EN <sub>A</sub> /DIAG <sub>A</sub> EN <sub>B</sub> /DIAG <sub>B</sub>	Open drain bidirectional logic pins. These pins must be connected to an external pull up resistor. When externally pulled low, they disable half-bridge A or B. In case of fault detection (thermal shutdown of a High-Side FET or excessive ON state voltage drop across a Low-Side FET), these pins are pulled low by the device (see truth table in fault condition).
CS	Analog current sense output. This output sources a current proportional to the motor current. The information can be read back as an analog voltage across an external resistor.



## VNH2SP30-E

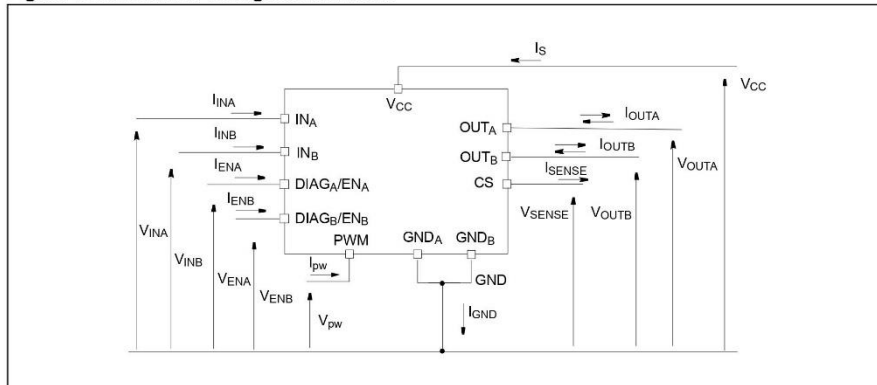
**Table 5. Block Descriptions** (see Block Diagram)

Name	Description
LOGIC CONTROL	Allows the turn-on and the turn-off of the High Side and the Low Side switches according to the truth table.
OVERVOLTAGE + UNDERVOLTAGE	Shut-down the device outside the range [5.5V..16V] for the battery voltage.
HIGH SIDE AND LOW SIDE CLAMP VOLTAGE	Protect the High Side and the Low Side switches from the high voltage on the battery line in all configuration for the motor.
HIGH SIDE AND LOW SIDE DRIVER	Drive the gate of the concerned switch to allow a proper $R_{DS(on)}$ for the leg of the bridge.
LINEAR CURRENT LIMITER	Limits the motor current, by reducing the High Side Switch gate-source voltage when short-circuit to ground occurs.
OVERTEMPERATURE PROTECTION	In case of short-circuit with the increase of the junction's temperature, shuts-down the concerned High Side to prevent its degradation and to protect the die.
FAULT DETECTION	Signalize an abnormal behavior of the switches in the half-bridge A or B by pulling low the concerned ENx/DIAGx pin.

**Table 6. Absolute Maximum Rating**

Symbol	Parameter	Value	Unit
$V_{CC}$	Supply Voltage	+ 41	V
$I_{max}$	Maximum Output Current (continuous)	30	A
$I_R$	Reverse Output Current (continuous)	-30	A
$I_{IN}$	Input Current ( $IN_A$ and $IN_B$ pins)	+/- 10	mA
$I_{EN}$	Enable Input Current ( $DIAG_A/EN_A$ and $DIAG_B/EN_B$ pins)	+/- 10	mA
$I_{pw}$	PWM Input Current	+/- 10	mA
$V_{CS}$	Current Sense Maximum Voltage	-3/+15	V
$V_{ESD}$	Electrostatic Discharge ( $R=1.5k\Omega$ , $C=100pF$ )		
	- CS pin	2	kV
	- logic pins	4	kV
	- output pins: $OUT_A$ , $OUT_B$ , $V_{CC}$	5	kV
$T_j$	Junction Operating Temperature	Internally Limited	$^{\circ}C$
$T_c$	Case Operating Temperature	-40 to 150	$^{\circ}C$
$T_{STG}$	Storage Temperature	-55 to 150	$^{\circ}C$

**Figure 4. Current and Voltage Conventions**



**Table 7. Thermal Data**

See MultiPowerSO-30 Thermal Data section (page )

**ELECTRICAL CHARACTERISTICS**

(V<sub>CC</sub>=9V up to 16V; -40°C<T<sub>J</sub><150°C; unless otherwise specified)

**Table 8. Power**

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V <sub>CC</sub>	Operating supply voltage		5.5		16	V
I <sub>S</sub>	Supply Current	Off state: I <sub>NA</sub> =I <sub>NB</sub> =PWM=0; T <sub>J</sub> =25°C; V <sub>CC</sub> =13V I <sub>NA</sub> =I <sub>NB</sub> =PWM=0		12	30 60	μA μA
		On state: I <sub>NA</sub> or I <sub>NB</sub> =5V, no PWM			10	mA
R <sub>ONHS</sub>	Static High-Side resistance	I <sub>OUT</sub> =15A; T <sub>J</sub> =25°C I <sub>OUT</sub> =15A; T <sub>J</sub> = - 40 to 150°C			14 28	mΩ mΩ
R <sub>ONLS</sub>	Static Low-Side resistance	I <sub>OUT</sub> =15A; T <sub>J</sub> =25°C I <sub>OUT</sub> =15A; T <sub>J</sub> = - 40 to 150°C			5 10	mΩ mΩ
V <sub>f</sub>	High Side Free-wheeling Diode Forward Voltage	I <sub>f</sub> =15A		0.8	1.1	V
I <sub>L(off)</sub>	High Side Off State Output Current (per channel)	T <sub>J</sub> =25°C; V <sub>OUTX</sub> =EN <sub>X</sub> =0V; V <sub>CC</sub> =13V T <sub>J</sub> =125°C; V <sub>OUTX</sub> =EN <sub>X</sub> =0V; V <sub>CC</sub> =13V			3 5	μA μA
I <sub>RM</sub>	Dynamic Cross-conduction Current	I <sub>OUT</sub> =15A (see fig. 8)		0.7		A

**Table 9. Logic Inputs (I<sub>NA</sub>, I<sub>NB</sub>, EN<sub>A</sub>, EN<sub>B</sub>)**

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V <sub>IL</sub>	Input Low Level Voltage	Normal operation (DIAG <sub>X</sub> /EN <sub>X</sub> pin acts as an input pin)			1.25	V
V <sub>IH</sub>	Input High Level Voltage	Normal operation (DIAG <sub>X</sub> /EN <sub>X</sub> pin acts as an input pin)	3.25			V
V <sub>IHYST</sub>	Input Hysteresis Voltage	Normal operation (DIAG <sub>X</sub> /EN <sub>X</sub> pin acts as an input pin)	0.5			V
V <sub>ICL</sub>	Input Clamp Voltage	I <sub>IN</sub> =1mA	5.5	6.3	7.5	V
		I <sub>IN</sub> =-1mA	-1.0	-0.7	-0.3	V
I <sub>INL</sub>	Input Current	V <sub>IN</sub> =1.25 V	1			μA
I <sub>INH</sub>	Input Current	V <sub>IN</sub> =3.25 V			10	μA
V <sub>DIAG</sub>	Enable Output Low Level Voltage	Fault operation (DIAG <sub>X</sub> /EN <sub>X</sub> pin acts as an output pin); I <sub>EN</sub> =1mA			0.4	V



## VNH2SP30-E

### ELECTRICAL CHARACTERISTICS (continued)

**Table 10. PWM**

Symbol	Parameter	Test Conditions	Min	Typ	Max	Unit
$V_{pwl}$	PWM Low Level Voltage				1.25	V
$I_{pwl}$	PWM Pin Current	$V_{pw}=1.25V$	1			mA
$V_{pwh}$	PWM High Level Voltage		3.25			V
$I_{pwh}$	PWM Pin Current	$V_{pw}=3.25V$			10	mA
$V_{pwhyst}$	PWM Hysteresis Voltage		0.5			V
$V_{pwcl}$	PWM Clamp Voltage	$I_{pw} = 1 \text{ mA}$	$V_{CC}+0.3$	$V_{CC}+0.7$	$V_{CC}+1.0$	V
		$I_{pw} = -1 \text{ mA}$	-6.0	-4.5	-3.0	V
$C_{INPWM}$	PWM Pin Input Capacitance	$V_{IN} = 2.5V$			25	pF

**Table 11. Switching** ( $V_{CC}=13V$ ,  $R_{LOAD}=0.87\Omega$ )

Symbol	Parameter	Test Conditions	Min	Typ	Max	Unit
$f$	PWM Frequency		0		20	kHz
$t_{d(on)}$	Turn-on Delay Time	Input rise time < 1ns (see fig. 8)			250	ns
$t_{d(off)}$	Turn-off Delay Time	Input rise time < 1ns (see fig. 8)			250	ns
$t_r$	Rise Time	(see fig. 7)		1	1.6	ns
$t_f$	Fall Time	(see fig. 7)		1.2	2.4	ns
$t_{DEL}$	Delay Time During Change of Operating Mode	(see fig. 6)	300	600	1800	ns
$t_{rr}$	High Side Free Wheeling Diode Reverse Recovery Time	(see fig. 9)		110		ns
$t_{off(min)}$	PWM Minimum off time	$9V < V_{CC} < 16V$ ; $-40^\circ C < T_j < 150^\circ C$ ; $I_{OUT}=15A$			6	ns

**Table 12. Protection And Diagnostic**

Symbol	Parameter	Test Conditions	Min	Typ	Max	Unit
$V_{USD}$	Undervoltage Shut-down				5.5	V
	Undervoltage Reset			4.7		V
$V_{OV}$	Overvoltage Shut-down		16	19	22	V
$I_{LIM}$	High-Side Current Limitation		30	50	70	A
$V_{CLP}$	Total Clamp Voltage ( $V_{CC}$ to GND)	$I_{OUT}=15A$	43	48	54	V
$T_{TSD}$	Thermal Shut-down Temperature	$V_{IN} = 3.25 V$	150	175	200	°C
$T_{TR}$	Thermal Reset Temperature		135			°C
$T_{HYST}$	Thermal Hysteresis		7	15		°C

**ELECTRICAL CHARACTERISTICS** (continued)

**Table 13. Current Sense** ( $9V < V_{CC} < 16V$ )

Symbol	Parameter	Test Conditions	Min	Typ	Max	Unit
$K_1$	$I_{OUT}/I_{SENSE}$	$I_{OUT}=30A$ ; $R_{SENSE}=1.5k\Omega$ $T_j = -40$ to $150^\circ C$	9665	11370	13075	
$K_2$	$I_{OUT}/I_{SENSE}$	$I_{OUT}=8A$ ; $R_{SENSE}=1.5k\Omega$ $T_j = -40$ to $150^\circ C$	9096	11370	13644	
$dK_1 / K_1$ (*)	Analog sense current drift	$I_{OUT}=30A$ ; $R_{SENSE}=1.5k\Omega$ $T_j = -40$ to $150^\circ C$	-8		+8	%
$dK_2 / K_2$ (*)	Analog sense current drift	$I_{OUT} > 8A$ ; $R_{SENSE}=1.5k\Omega$ $T_j = -40$ to $150^\circ C$	-10		+10	%
$I_{SENSE0}$	Analog Sense Leakage Current	$I_{OUT}=0A$ ; $V_{SENSE}=0V$ ; $T_j = -40$ to $150^\circ C$	0		65	$\mu A$

Note: (\*) Analog sense current drift is deviation of factor K for a given device over (-40°C to 150°C and  $9V < V_{CC} < 16V$ ) with respect to its value measured at  $T_j=25^\circ C$ ,  $V_{CC}=13V$ .

**WAVEFORMS AND TRUTH TABLE**

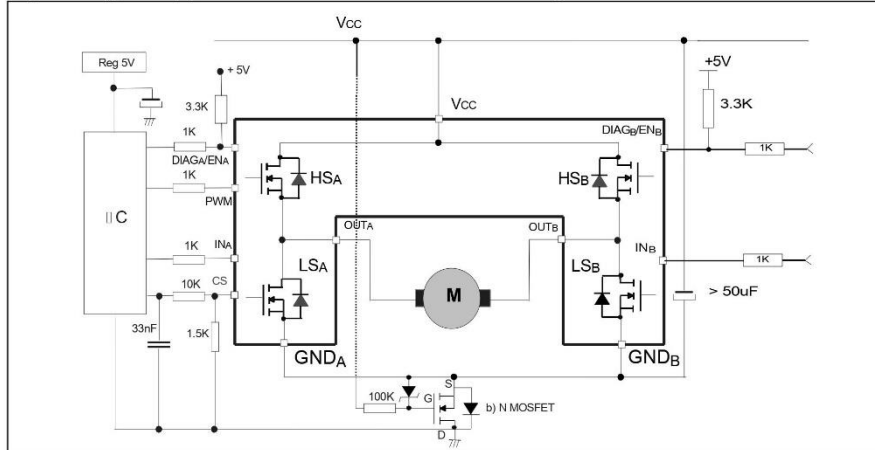
**Table 14. Truth Table In Normal Operating Conditions**

In normal operating conditions the  $DIAG_X/EN_X$  pin is considered as an input pin by the device. This pin must be externally pulled high.

PWM pin usage: in all cases, a "0" on the PWM pin will turn-off both  $LS_A$  and  $LS_B$  switches. When PWM rises back to "1",  $LS_A$  or  $LS_B$  turn on again depending on the input pin state.

$IN_A$	$IN_B$	$DIAG_A/EN_A$	$DIAG_B/EN_B$	$OUT_A$	$OUT_B$	CS	Operating mode
1	1	1	1	H	H	High Imp.	Brake to $V_{CC}$
1	0	1	1	H	L	$I_{SENSE}=I_{OUT}/K$	Clockwise (CW)
0	1	1	1	L	H	$I_{SENSE}=I_{OUT}/K$	Counterclockwise (CCW)
0	0	1	1	L	L	High Imp.	Brake to GND

Figure 5. Typical Application Circuit For Dc To 20khz PWM Operation Short Circuit Protection



In case of a fault condition the DIAG<sub>X</sub>/EN<sub>X</sub> pin is considered as an output pin by the device.

The fault conditions are:

- overtemperature on one or both high sides (for example if a short to ground occurs as it could be the case described in line 1 and 2 in the table below);
- short to battery condition on the output (saturation detection on the Low-Side Power MOSFET).

which power element is in fault by monitoring the IN<sub>A</sub>, IN<sub>B</sub>, DIAG<sub>A</sub>/EN<sub>A</sub> and DIAG<sub>B</sub>/EN<sub>B</sub> pins.

In any case, when a fault is detected, the faulty leg of the bridge is latched off. To turn-on the respective output (OUT<sub>X</sub>) again, the input signal must rise from low to high level.

Possible origins of fault conditions may be:

OUT<sub>A</sub> is shorted to ground ---> overtemperature detection on high side A.

OUT<sub>A</sub> is shorted to V<sub>CC</sub> ---> Low-Side Power MOSFET saturation detection.

When a fault condition is detected, the user can know

Table 15. Truth Table In Fault Conditions (Detected On OUT<sub>A</sub>)

IN <sub>A</sub>	IN <sub>B</sub>	DIAG <sub>A</sub> /EN <sub>A</sub>	DIAG <sub>B</sub> /EN <sub>B</sub>	OUT <sub>A</sub>	OUT <sub>B</sub>	CS
1	1	0	1	OPEN	H	High Imp.
1	0	0	1	OPEN	L	High Imp.
0	1	0	1	OPEN	H	I <sub>OUTB</sub> /K
0	0	0	1	OPEN	L	High Imp.
X	X	0	0	OPEN	OPEN	High Imp.
X	1	0	1	OPEN	H	I <sub>OUTB</sub> /K
X	0	0	1	OPEN	L	High Imp.

Fault Information

Protection Action

Table 16. Electrical Transient Requirements

ISO T/R 7637/1 Test Pulse	Test Level I	Test Level II	Test Level III	Test Level IV	Test Levels Delays and Impedance
1	-25V	-50V	-75V	-100V	2ms, 10Ω
2	+25V	+50V	+75V	+100V	0.2ms, 10Ω
3a	-25V	-50V	-100V	-150V	0.1 s, 50Ω
3b	+25V	+50V	+75V	+100V	0.1 s, 50Ω
4	-4V	-5V	-6V	-7V	100ms, 0.01Ω
5	+26.5V	+46.5V	+66.5V	+86.5V	400ms, 2Ω

ISO T/R 7637/1 Test Pulse	Test Levels Result I	Test Levels Result II	Test Levels Result III	Test Levels Result IV
1	C	C	C	C
2	C	C	C	C
3a	C	C	C	C
3b	C	C	C	C
4	C	C	C	C
5	C	E	E	E

Class	Contents
C	All functions of the device are performed as designed after exposure to disturbance.
E	One or more functions of the device are not performed as designed after exposure to disturbance and cannot be returned to proper operation without replacing the device.

**Reverse Battery Protection**

Three possible solutions can be thought of:

- a) a Schottky diode D connected to V<sub>CC</sub> pin
- b) a N-channel MOSFET connected to the GND pin (see Typical Application Circuit on page 8)
- c) a P-channel MOSFET connected to the V<sub>CC</sub> pin

The device sustains no more than -30A in reverse battery conditions because of the two Body diodes of the Power MOSFETs. Additionally, in reverse battery condition the I/Os of VNH2SP30-E will be pulled down to the V<sub>CC</sub> line (approximately -1.5V). Series resistor must be inserted to limit the current sunk from the microcontroller I/Os. If I<sub>Rmax</sub> is the maximum target reverse current through I/Os, series resistor is:

$$R = \frac{V_{IOs} - V_{CC}}{I_{Rmax}}$$

Figure 6. Definition Of The Delay Times Measurement

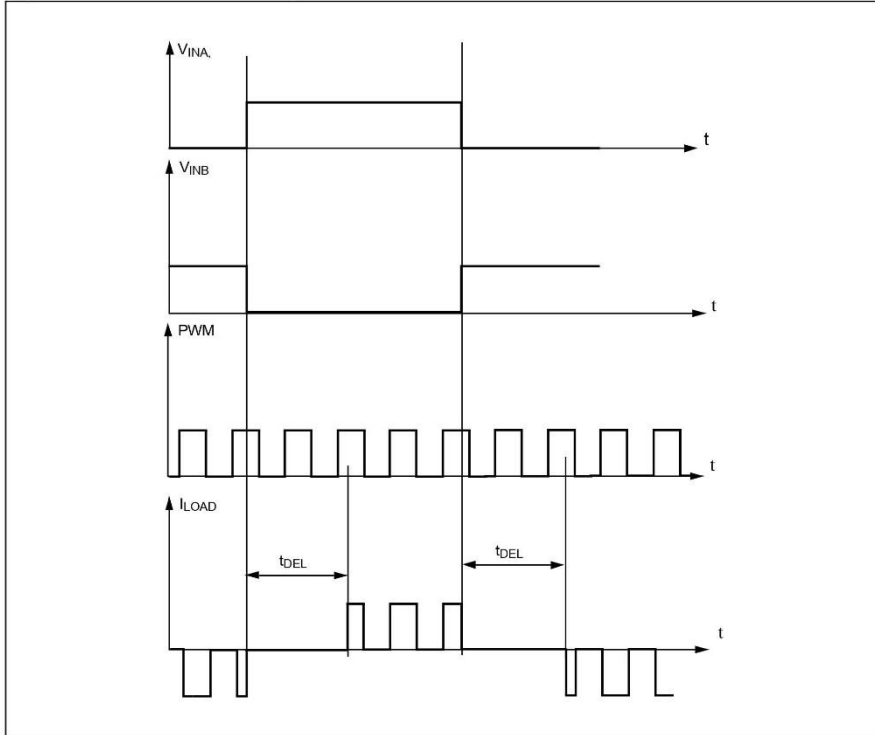


Figure 7. Definition Of The Low Side Switching Times

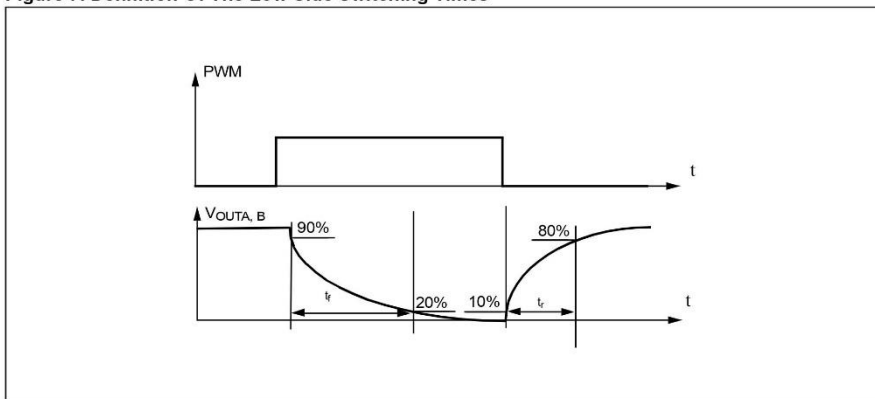


Figure 8. Definition Of The High Side Switching Times

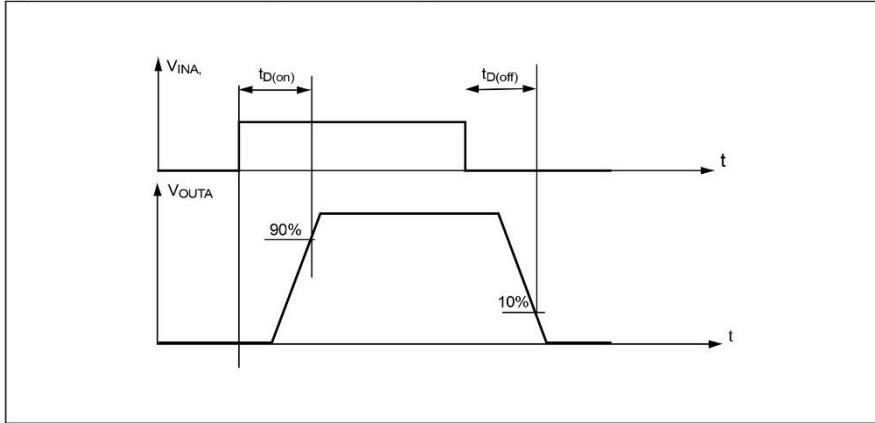


Figure 9. Definition Of Dynamic Cross Conduction Current During A Pwm Operation

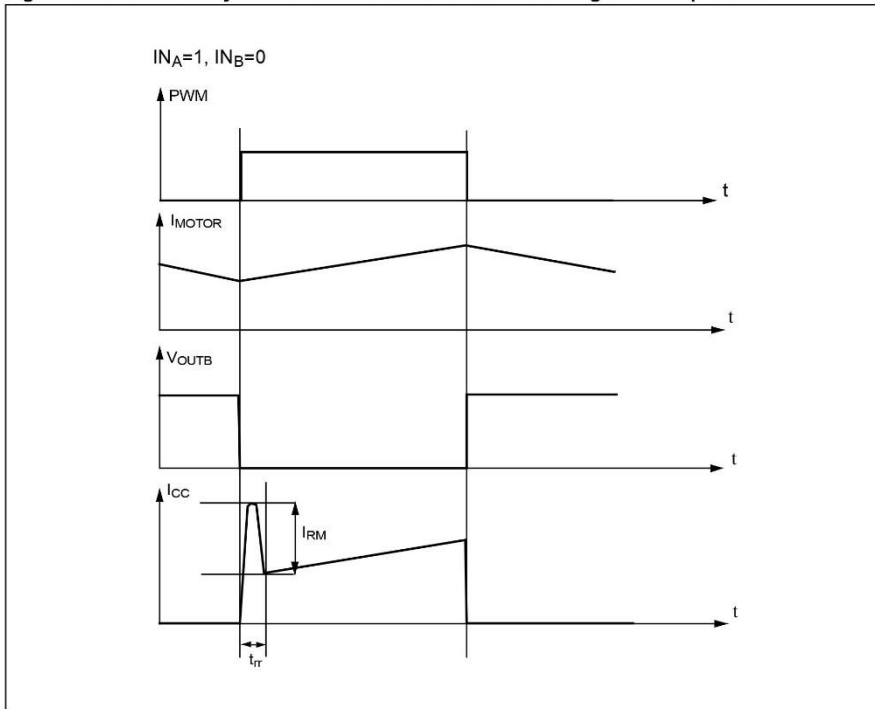




Figure 10. Waveforms in full bridge operation

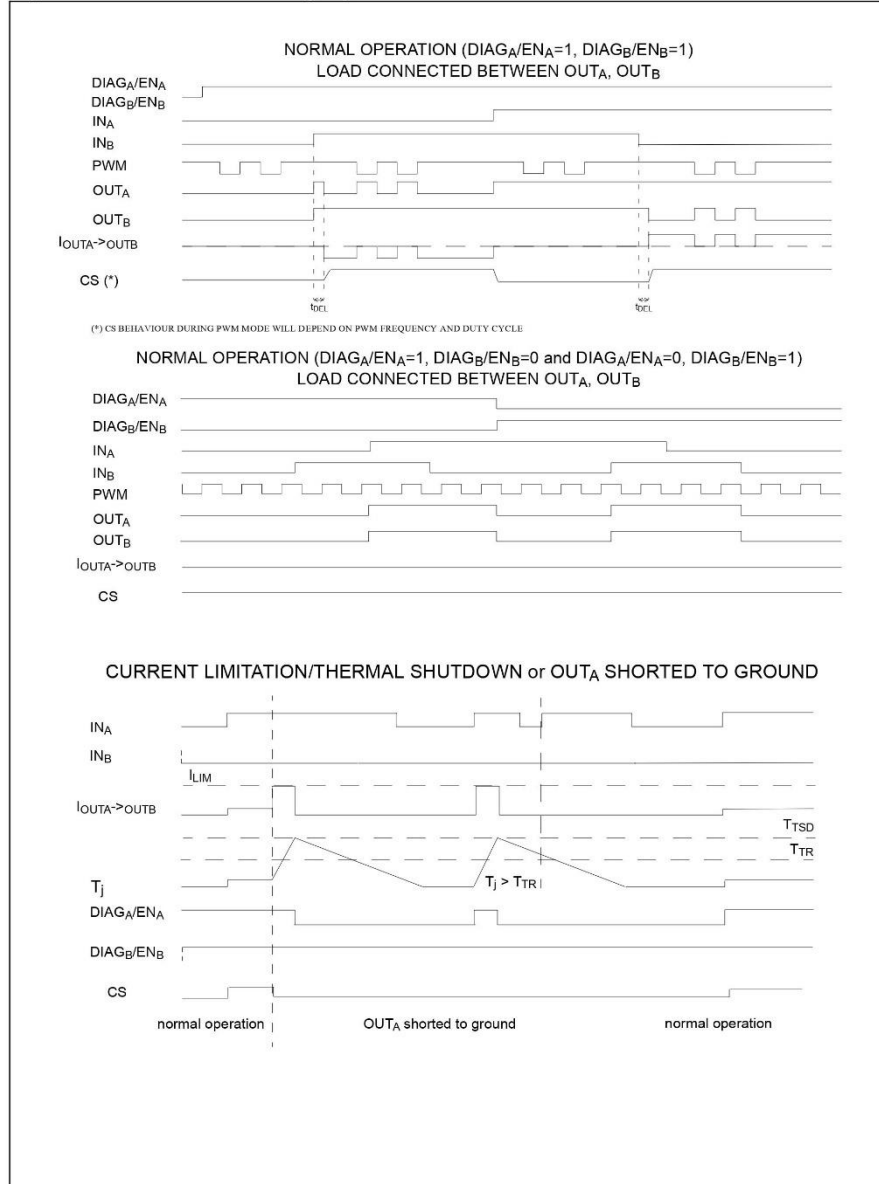
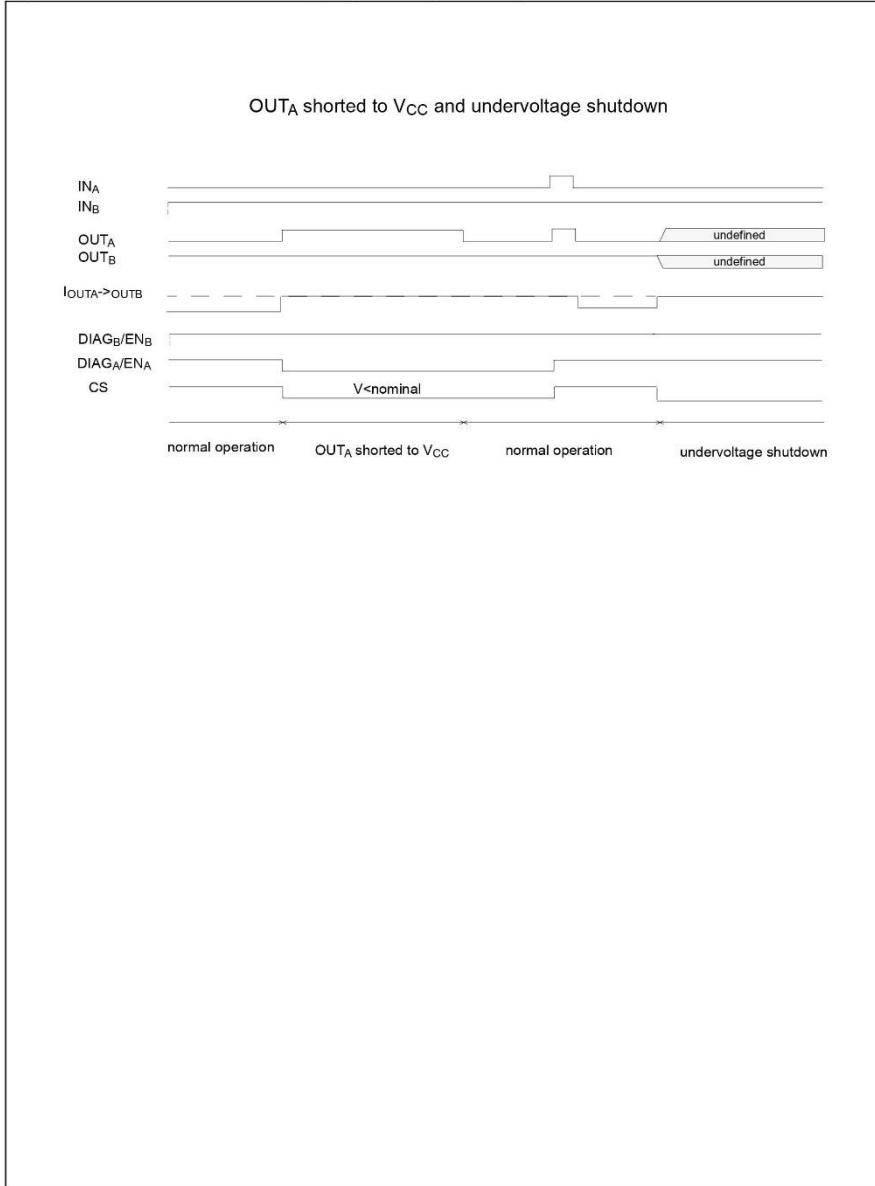


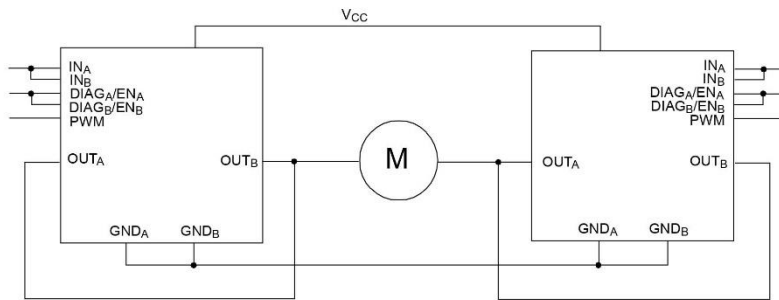
Figure 11. Waveforms In Full Bridge Operation (continued)



## VNH2SP30-E

**Figure 12. Half-bridge Configuration**

The VNH2SP30-E can be used as a high power half-bridge driver achieving an On resistance per leg of  $9.5\text{m}\Omega$ . Suggested configuration is the following:



**Figure 13. Multi-motors Configuration**

The VNH2SP30-E can easily be designed in multi-motors driving applications such as seat positioning systems where only one motor must be driven at a time.  $\text{DIAG}_X/\text{EN}_X$  pins allow to put unused half-bridges in high impedance. Suggested configuration is the following:

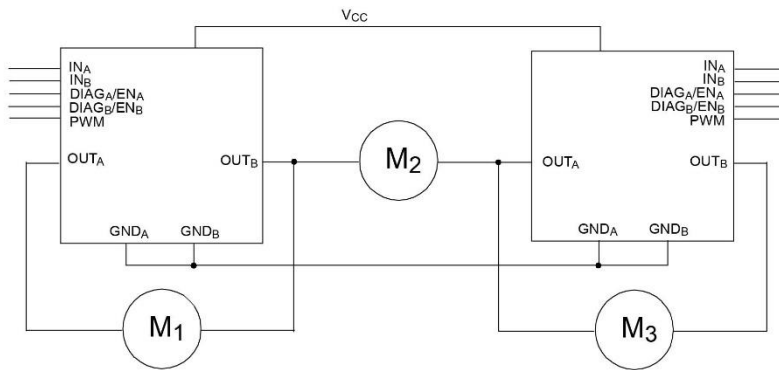


Figure 14. On State Supply Current

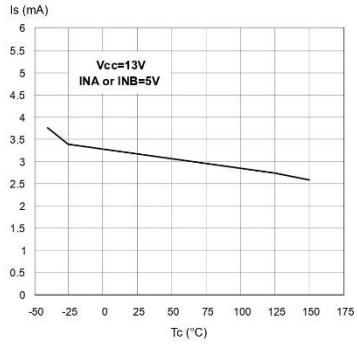


Figure 17. Off State Supply Current

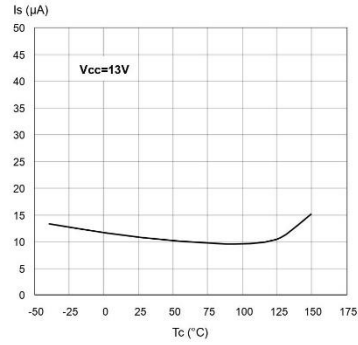


Figure 15. High Level Input Current

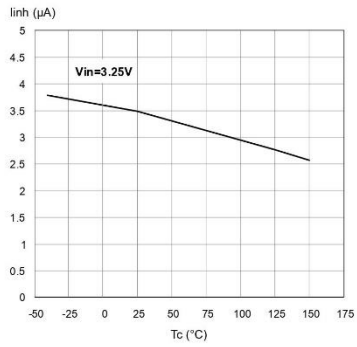


Figure 18. Input Clamp Voltage

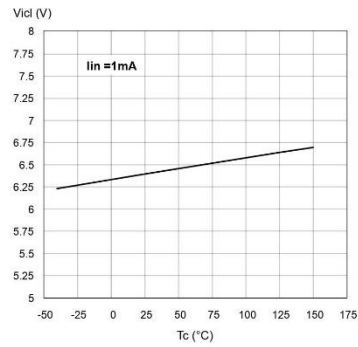


Figure 16. Input High Level Voltage

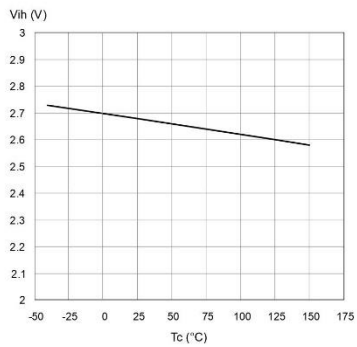


Figure 19. Input Low Level Voltage

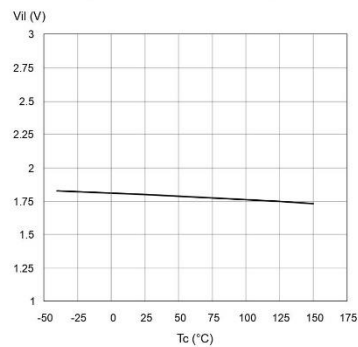


Figure 20. Input Hysteresis Voltage

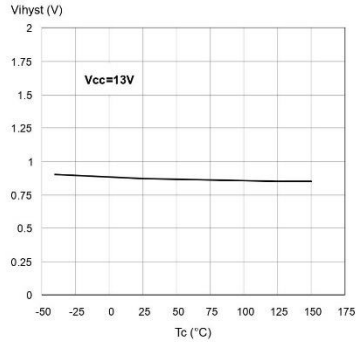


Figure 23. High Level Enable Pin Current

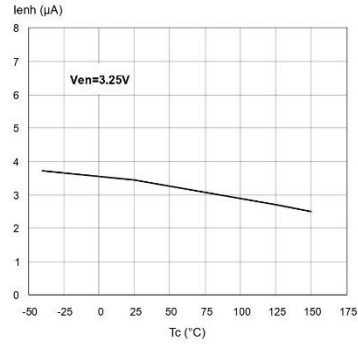


Figure 21. Delay Time during change of operation mode

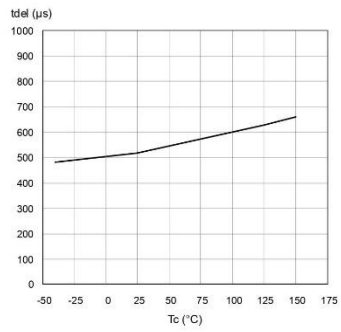


Figure 24. Enable Clamp Voltage

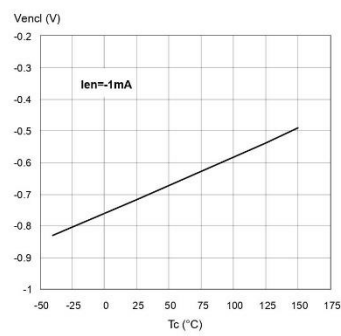


Figure 22. High Level Enable Voltage

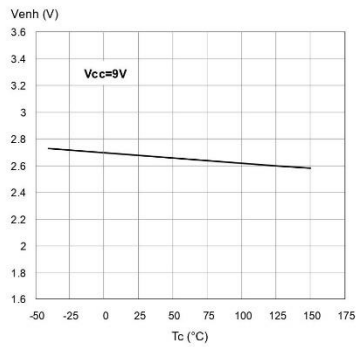


Figure 25. Low Level Enable Voltage

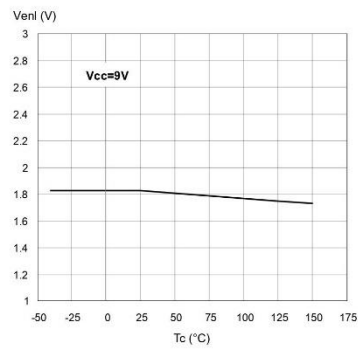


Figure 26. PWM High Level Voltage

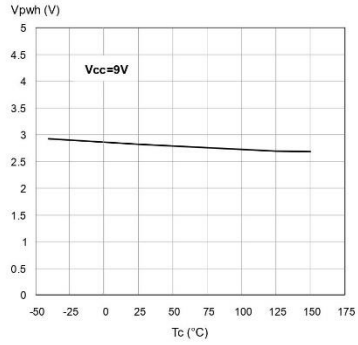


Figure 29. PWM Low Level Voltage

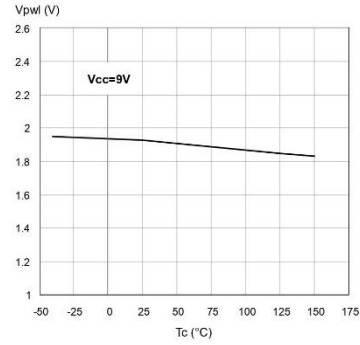


Figure 27. PWM High Level Current

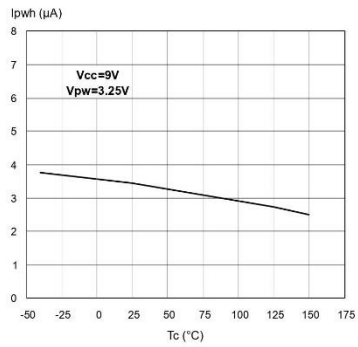


Figure 30. Overvoltage Shutdown

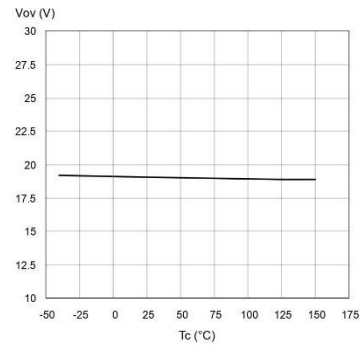


Figure 28. Undervoltage Shutdown

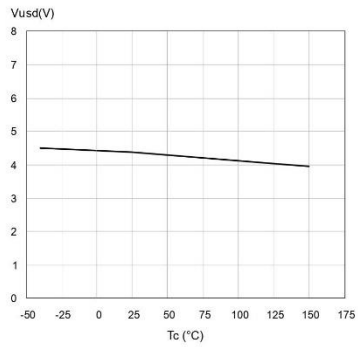


Figure 31. Current Limitation

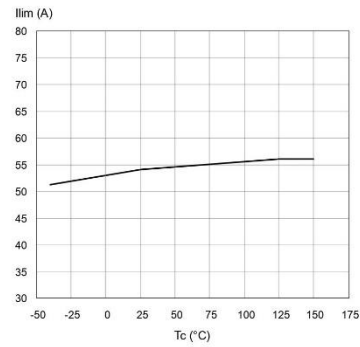


Figure 32. On State High Side Resistance Vs. Tcase

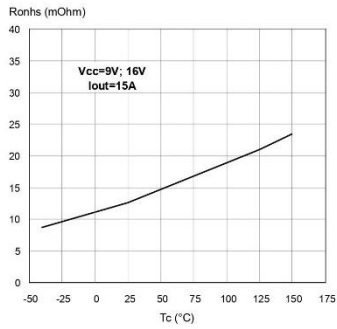


Figure 35. On State Low Side Resistance Vs. Tcase

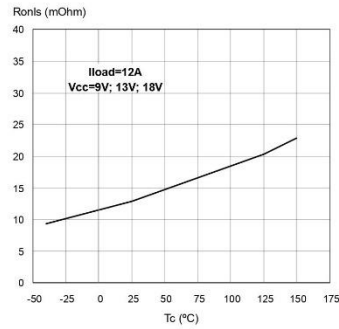


Figure 33. Turn-on Delay Time

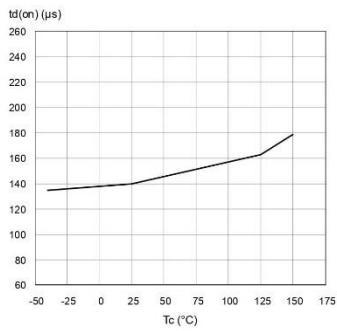


Figure 36. Turn-off Delay Time

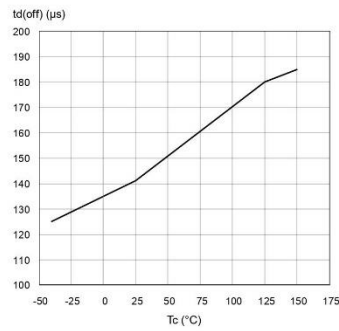


Figure 34. Output Voltage Rise Time

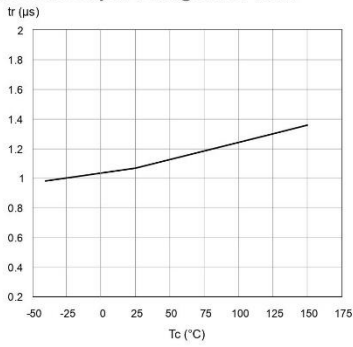
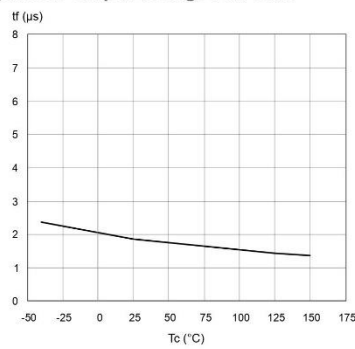


Figure 37. Output Voltage Fall Time



MultiPowerSO-30™ Thermal Data

Figure 38. MultiPowerSO-30™ PC Board

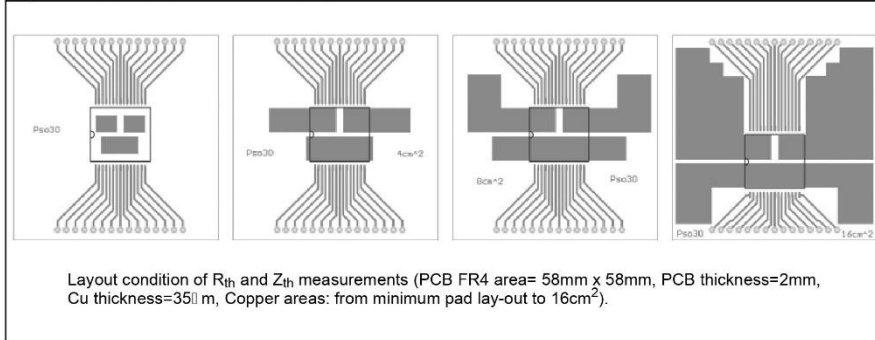


Figure 39. Chipset Configuration

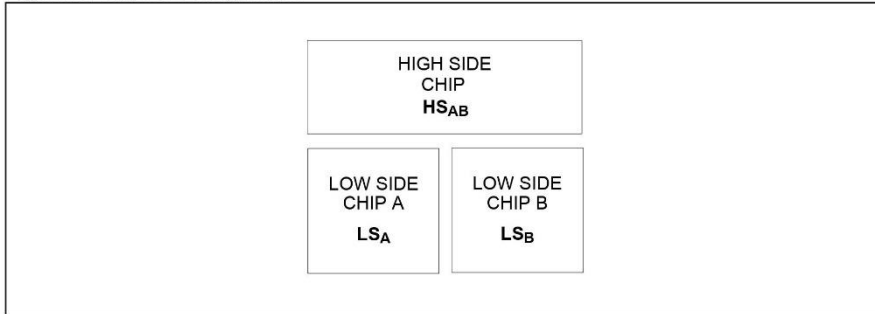
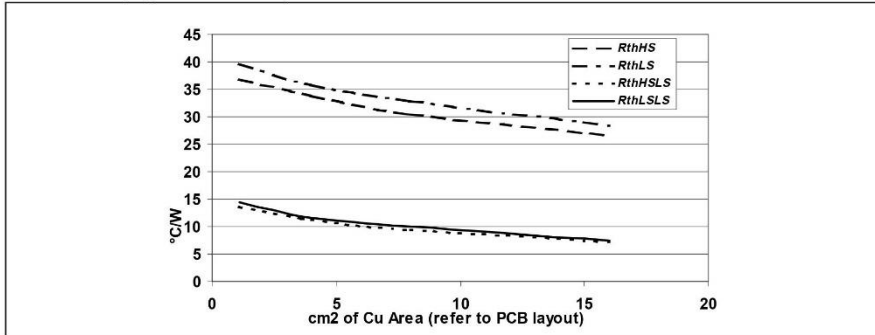


Figure 40. Auto and mutual  $R_{thj-amb}$  Vs PCB copper area in open box free air condition (according to page 20 definitions)





VNH2SP30-E

Table 17. Thermal Calculation In Clockwise And Anti-clockwise Operation In Steady-state Mode

HS <sub>A</sub>	HS <sub>B</sub>	LS <sub>A</sub>	LS <sub>B</sub>	T <sub>JHSAB</sub>	T <sub>JLSA</sub>	T <sub>JLSB</sub>
ON	OFF	OFF	ON	$\frac{P_{dHSA} \times R_{thHS} + P_{dLSB} \times R_{thHLS}}{R_{thHLS} + T_{amb}}$	$\frac{P_{dHSA} \times R_{thHLS} + P_{dLSB} \times R_{thLS}}{R_{thLS} + T_{amb}}$	$\frac{P_{dHSA} \times R_{thHLS} + P_{dLSB} \times R_{thLS}}{R_{thLS} + T_{amb}}$
OFF	ON	ON	OFF	$\frac{P_{dHSB} \times R_{thHS} + P_{dLSA} \times R_{thHLS}}{R_{thHLS} + T_{amb}}$	$\frac{P_{dHSB} \times R_{thHLS} + P_{dLSA} \times R_{thLS}}{R_{thLS} + T_{amb}}$	$\frac{P_{dHSB} \times R_{thHLS} + P_{dLSA} \times R_{thLS}}{R_{thLS} + T_{amb}}$

**Thermal Resistances Definition** (values according to the PCB heatsink area)

**R<sub>thHS</sub>** = R<sub>thHSA</sub> = R<sub>thHSB</sub> = High Side Chip Thermal Resistance Junction to Ambient (HS<sub>A</sub> or HS<sub>B</sub> in ON state)

**R<sub>thLS</sub>** = R<sub>thLSA</sub> = R<sub>thLSB</sub> = Low Side Chip Thermal Resistance Junction to Ambient

**R<sub>thHLS</sub>** = R<sub>thHSALS</sub> = R<sub>thHSBLS</sub> = Mutual Thermal Resistance Junction to Ambient between High Side and Low Side Chips

**R<sub>thLSLS</sub>** = R<sub>thLSALS</sub> = Mutual Thermal Resistance Junction to Ambient between Low Side Chips

**Thermal Calculation In Transient Mode (\*)**

$$T_{JHSAB} = Z_{thHS} \times P_{dHSAB} + Z_{thHLS} \times (P_{dLSA} + P_{dLSB}) + T_{amb}$$

$$T_{JLSA} = Z_{thHLS} \times P_{dHSAB} + Z_{thLS} \times P_{dLSA} + Z_{thLSLS} \times P_{dLSB} + T_{amb}$$

$$T_{JLSB} = Z_{thHLS} \times P_{dHSAB} + Z_{thLSLS} \times P_{dLSA} + Z_{thLS} \times P_{dLSB} + T_{amb}$$

**Single Pulse Thermal Impedance Definition** (values according to the PCB heatsink area)

**Z<sub>thHS</sub>** = High Side Chip Thermal Impedance Junction to Ambient

**Z<sub>thLS</sub>** = Z<sub>thLSA</sub> = Z<sub>thLSB</sub> = Low Side Chip Thermal Impedance Junction to Ambient

**Z<sub>thHLS</sub>** = Z<sub>thHSABLSA</sub> = Z<sub>thHSABLSB</sub> = Mutual Thermal Impedance Junction to Ambient between High Side and Low Side Chips

**Z<sub>thLSLS</sub>** = Z<sub>thLSALS</sub> = Mutual Thermal Impedance Junction to Ambient between Low Side Chips

**Pulse Calculation Formula**

$$Z_{TH\Omega} = R_{TH} \cdot \Omega + Z_{TH\Omega tp} (1 - \Omega)$$

where  $\Omega = t_p / T$

(\*) Calculation is valid in any dynamic operating condition. P<sub>d</sub> values set by user.



Figure 41. MultiPowerSO-30 HSD Thermal Impedance Junction Ambient Single Pulse

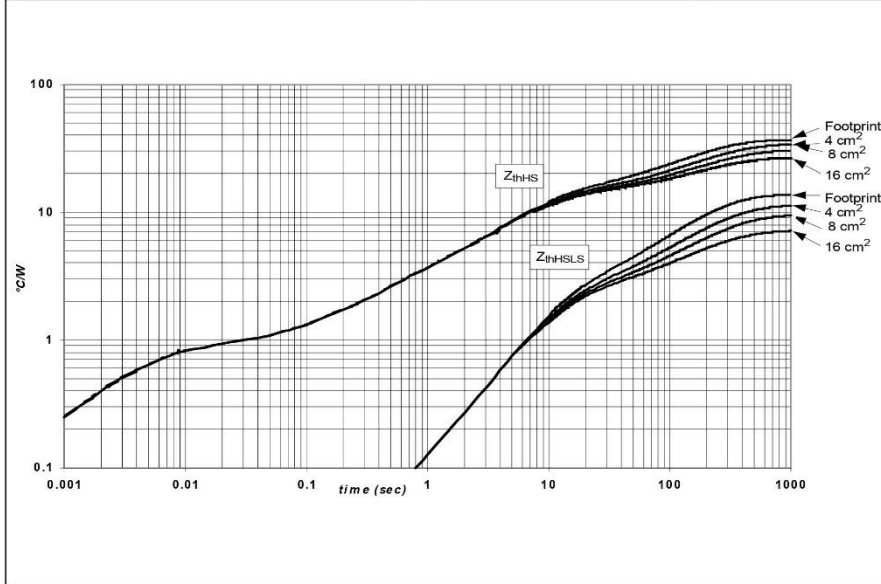


Figure 42. MultiPowerSo-30 LSD Thermal Impedance Junction Ambient Single Pulse

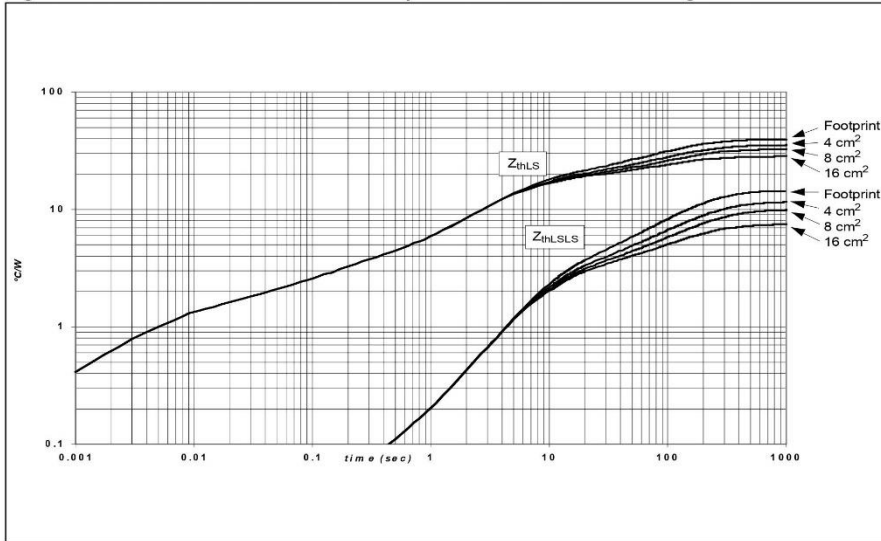


Figure 43. Thermal fitting model of an H-Bridge in MultiPowerSO-30

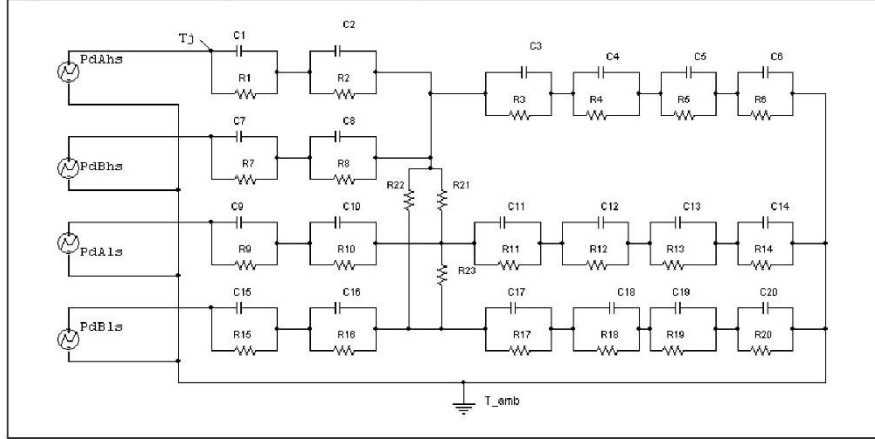


Table 18. Thermal Parameter (\*)

Area/Island (cm <sup>2</sup> )	Footprint	4	8	16
R1=R7 (°C/W)	0.05			
R2=R8 (°C/W)	0.3			
R3 (°C/W)	0.5			
R4 (°C/W)	1.3			
R5 (°C/W)	1.4			
R6 (°C/W)	44.7	39.1	31.6	23.7
R9=R15 (°C/W)	0.2			
R10=R16 (°C/W)	0.4			
R11=R17 (°C/W)	0.8			
R12=R18 (°C/W)	1.5			
R13=R19 (°C/W)	20			
R14=R20 (°C/W)	46.9	36.1	30.4	20.8
R21=R22=R23 (°C/W)	115			
C1=C7 (W.s/°C)	0.005			
C2=C8 (W.s/°C)	0.008			
C3=C11=C17 (W.s/°C)	0.01			
C4=C13=C19 (W.s/°C)	0.3			
C5 (W.s/°C)	0.6			
C6 (W.s/°C)	5	7	9	11
C9=C15 (W.s/°C)	0.003			
C10=C16 (W.s/°C)	0.006			
C12=C18 (W.s/°C)	0.075			
C14=C20 (W.s/°C)	2.5	3.5	4.5	5.5

Note: (\*) The blank space means that the value is the same as the previous one.

PACKAGE MECHANICAL

Table 19. MultiPowerSO-30 Mechanical Data

Symbol	millimeters		
	Min.	Typ	Max.
A			2.35
A2	1.85		2.25
A3	0		0.1
B	0.42		0.58
C	0.23		0.32
D	17.1	17.2	17.3
E	18.85		19.15
E1	15.9	16	16.1
e		1	
F1	5.55		6.05
F2	4.6		5.1
F3	9.6		10.1
L	0.8		1.15
N			10deg
S	0deg		7deg

Figure 44. MultiPowerSO-30 Package Dimensions

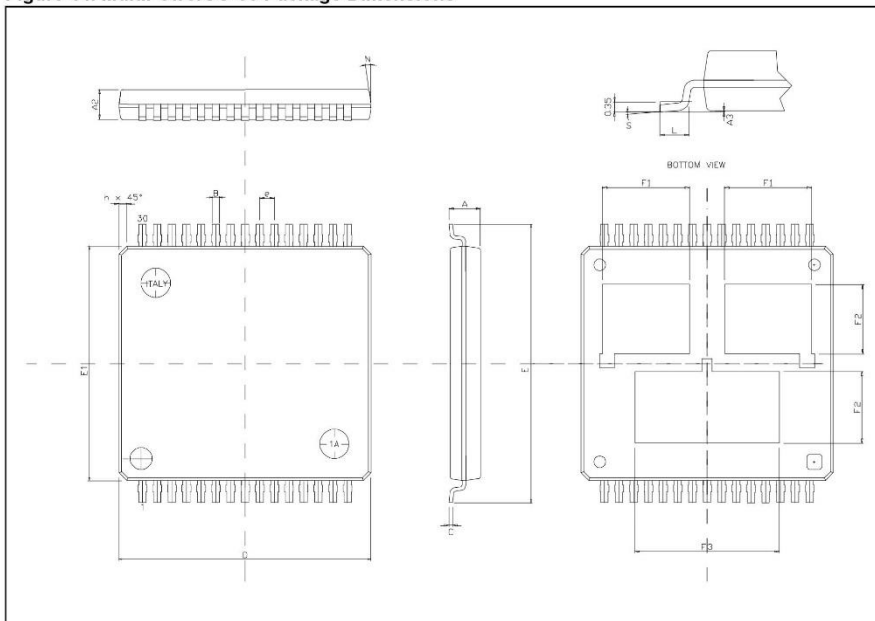
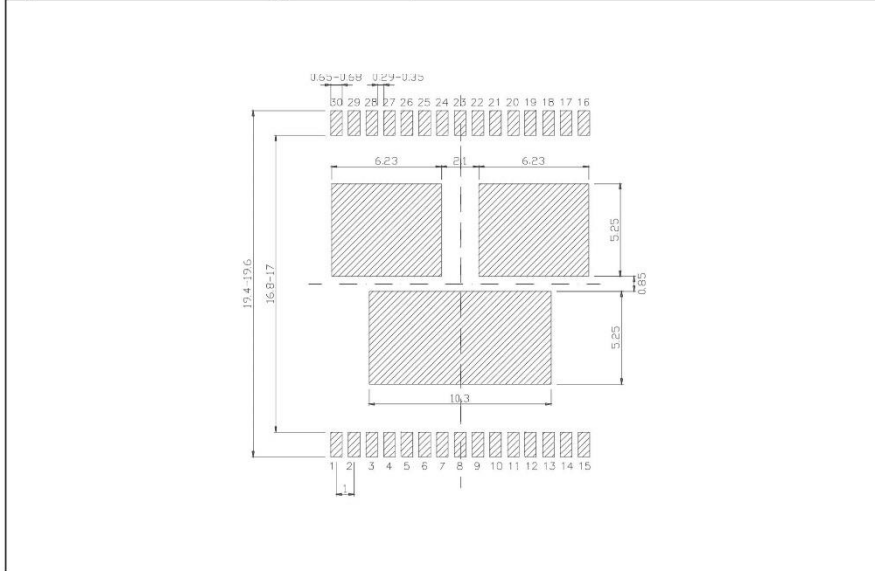


Figure 45. MultiPowerSO-30 Suggested Pad Layout



**REVISION HISTORY**

Date	Revision	Description of Changes
Sep. 2004	1	- First issue.

Information furnished is believed to be accurate and reliable. However, STMicroelectronics assumes no responsibility for the consequences of use of such information nor for any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of STMicroelectronics. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. STMicroelectronics products are not authorized for use as critical components in life support devices or systems without express written approval of STMicroelectronics.

The ST logo is a registered trademark of STMicroelectronics.  
All other names are the property of their respective owners

© 2004 STMicroelectronics - All rights reserved

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan -  
Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America  
[www.st.com](http://www.st.com)

## ANEXOS C: IMÁGENES

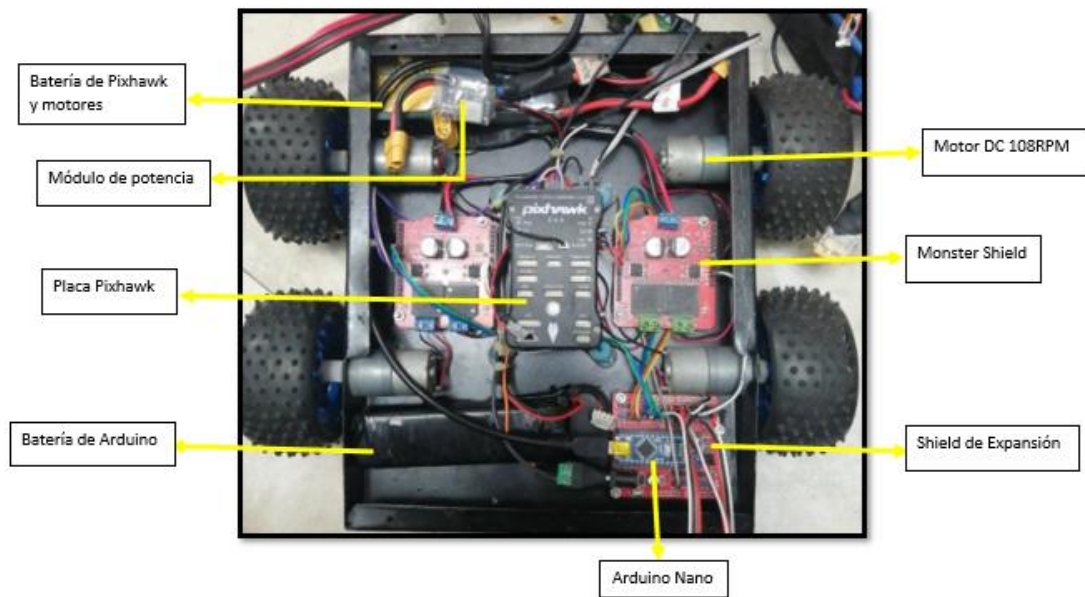


Fig. 73 Montaje de la parte inferior del prototipo, parte interna.

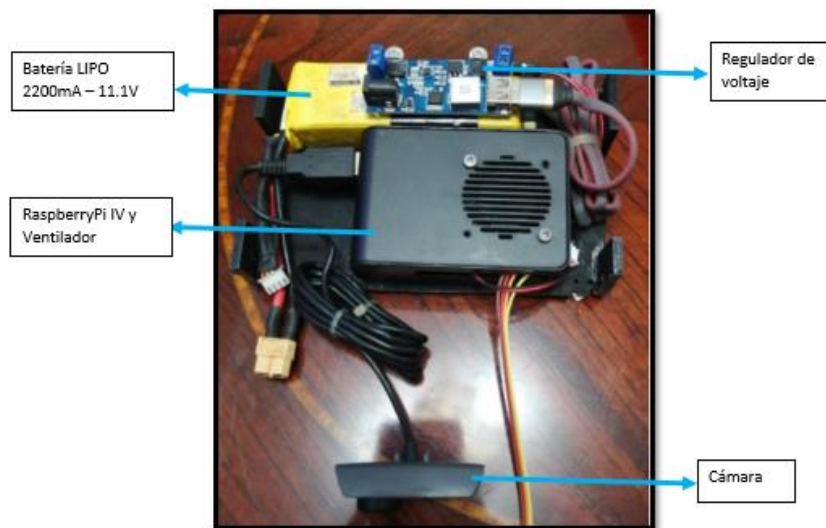


Fig. 74 Montaje de la parte superior del prototipo, nivel 1.



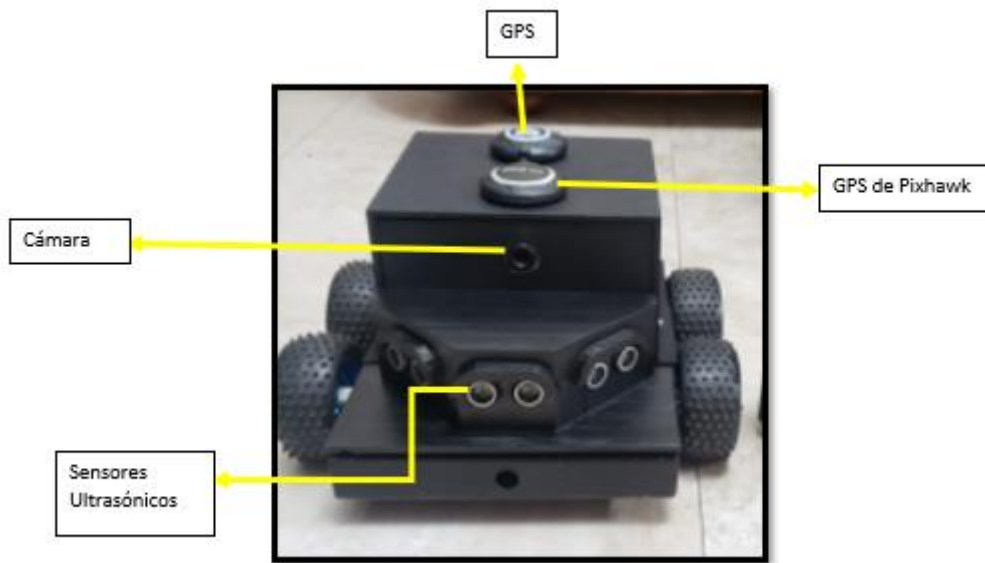


Fig. 75 Ensamble y montaje de la parte externa del prototipo.

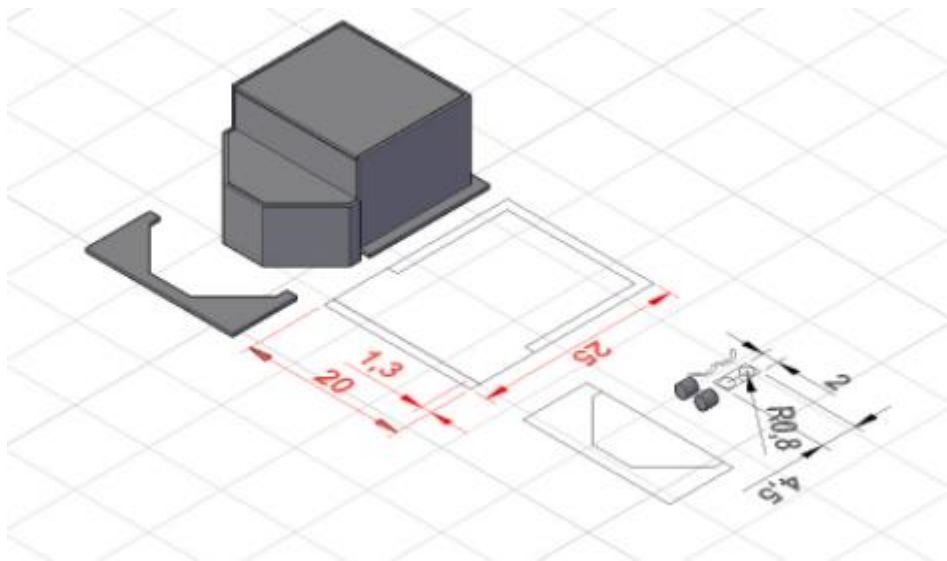


Fig. 76 Planos del diseño en 3D de la cubierta del prototipo

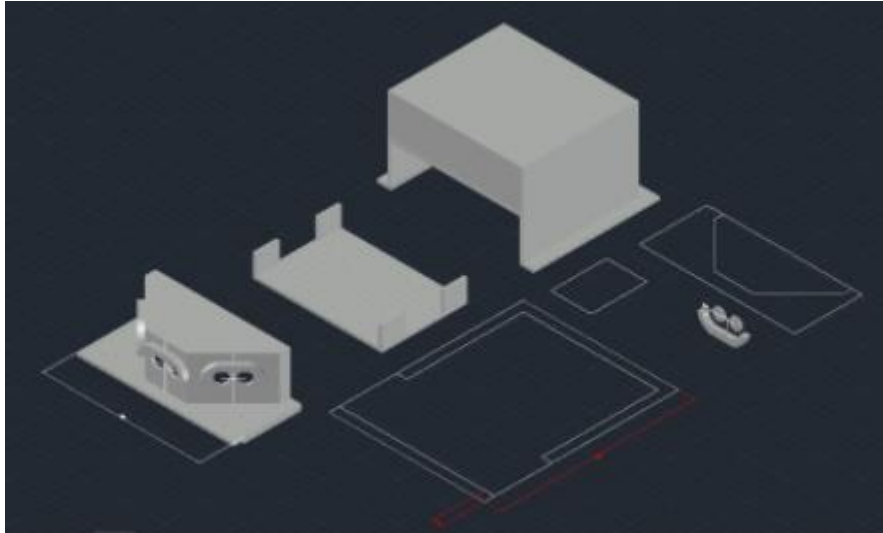


Fig. 77 Planos del diseño sólido en 3D de la cubierta y nivel 1 interno del prototipo

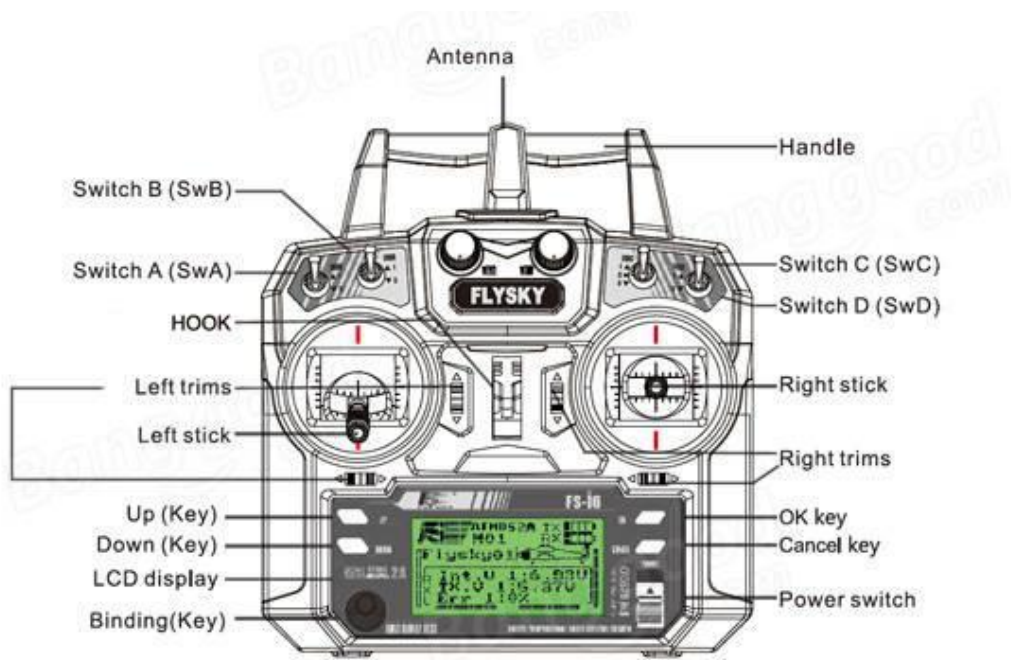


Fig. 78 Estructura de los controles del control RC.

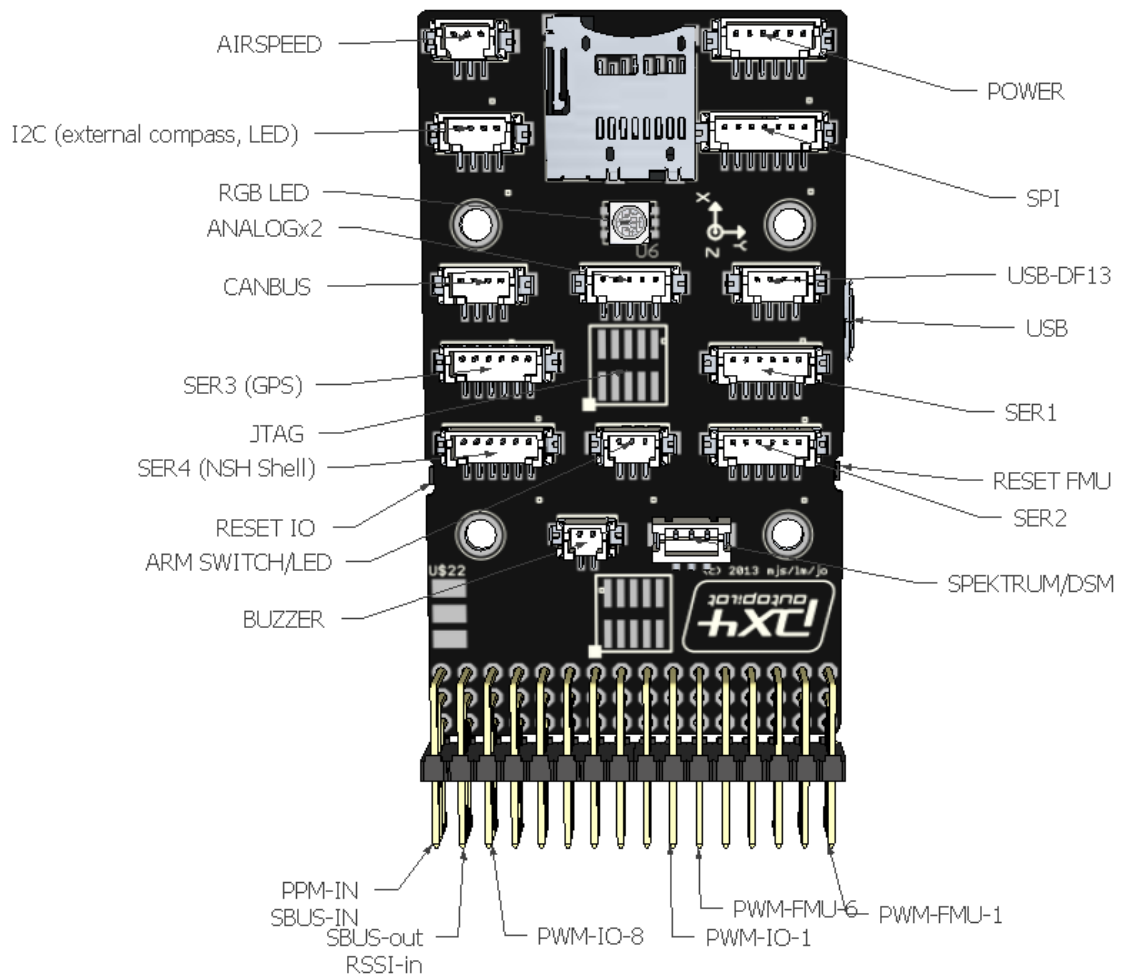
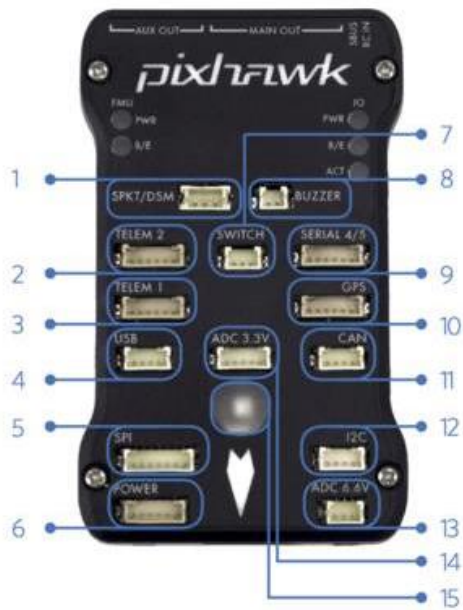


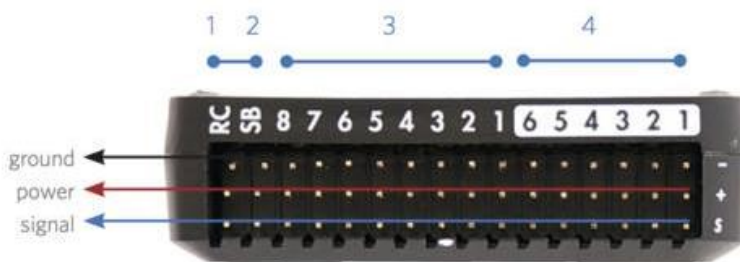
Fig. 79 Diagrama del conector de la placa Pixhawk



- 1 Spektrum DSM receiver
- 2 Telemetry (on-screen display)
- 3 Telemetry (radio telemetry)
- 4 USB
- 5 SPI (serial peripheral interface) bus
- 6 Power module
- 7 Safety switch button
- 8 Buzzer
- 9 Serial
- 10 GPS module
- 11 CAN (controller area network) bus
- 12 I<sup>2</sup>C splitter or compass module
- 13 Analog to digital converter 6.6 V
- 14 Analog to digital converter 3.3 V
- 15 LED indicator



- 1 Input/output reset button
- 2 SD card
- 3 Flight management reset button
- 4 Micro-USB port



- 1 Radio control receiver input
- 2 S.Bus output
- 3 Main outputs
- 4 Auxiliary outputs

Fig. 80 Conectores de la placa programadora Pixhawk

Alfiler	Señal	Voltio
1 (rojo)	VCC	+ 5V
2 (negro)	TX (SALIDA)	+ 3,3 V
3 (negro)	RX (EN)	+ 3,3 V
4 (negro)	CTS	+ 3,3 V
5 (negro)	RTS	+ 3,3 V
6 (negro)	GND	GND

Fig. 81 Asignación de pines del puerto utilizado TELEM1 o Telem2 de la placa programadora Pixhawk

Alfiler	Señal	Voltio
1 (rojo)	VCC	+ 5V
2 (negro)	TX (SALIDA)	+ 3,3 V
3 (negro)	RX (EN)	+ 3,3 V
4 (negro)	CAN2 TX	+ 3,3 V
5 (negro)	CAN2 RX	+ 3,3 V
6 (negro)	GND	GND

Fig. 82 Asignación de pines del puerto utilizado GPS de la placa programadora Pixhawk

Alfiler	Señal	Voltio
1 (rojo)	VCC	+ 5V
2 (negro)	SCL	+3,3 (dominadas)
3 (negro)	SDA	+3,3 (dominadas)
4 (negro)	GND	GND

Fig. 83 Asignación de pines del puerto utilizado I2C de la placa programadora Pixhawk



Fig. 84 Exposición y validación de proyecto de tesis en la Universidad Politécnica Salesiana a los estudiantes del curso de Control Automático.

### ENCUESTA DE MEDICIÓN DE IMPACTO

**PROYECTO:** DISEÑO DE UN PROTOTIPO DE ROBOT CON NAVEGACIÓN AUTOMÁTICA GPS PARA INSPECCIÓN DE RUTAS CON VISIÓN ARTIFICIAL.

**OBJETIVO:** Recabar información en cuanto a los niveles de impacto en la ejecución del Proyecto Previa-Titulación.

**INSTRUCCIONES:** Marque con una X en la opción que considere pertinente.

**PREGUNTAS:**

1. Usted como beneficiario, ¿Considera que el proyecto cumplió con los requerimientos solicitados de manera correcta?:

- Totalmente de acuerdo
- Parcialmente de Acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo

2. Bajo su percepción este prototipo mejoraría la calidad de la medida de rescate de la población y la reducción de pérdidas humanas posterior a la aplicación del proyecto.

- Totalmente de acuerdo
- Parcialmente de Acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo

3. ¿Cómo calificaría usted los componentes y/o tecnologías utilizadas en base al desarrollo del proyecto?

- Totalmente de acuerdo
- Parcialmente de Acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo

4. ¿Considera Usted que el proyecto realizado debería ser puesto en marcha de manera más profundizada y a mayor escala en un modelo ya concreto?

- Totalmente de acuerdo
- Parcialmente de Acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo

5. Bajo su percepción, ¿Qué otras tecnologías sería interesante proponer e implementar?

*Totalmente de acuerdo con la propuesta.*

Fig. 85 Encuesta de medición de impacto del proyecto de tesis en la Universidad Politécnica Salesiana a los estudiantes del curso de Control Automático.



Fig. 86 Exposición del proyecto de tesis en webinar desarrollado el martes 17 de agosto por la Universidad Politécnica Salesiana.