INSTITUTO TECNOLÓGICO DE CIUDAD GUZMÁN

TESIS

TEMA:

DISEÑO DE UN DISPOSITIVO DE CONTROL ASISTIDO PARA UAV

QUE PARA OBTENER EL TÍTULO DE:

MAESTRO EN INGENIERÍA ELECTRÓNICA

PRESENTA:

ING. JORGE LUIS SÁNCHEZ SERRANO

DIRECTOR DE TESIS:

DR. HUMBERTO BRACAMONTES DEL TORO

CODIRECTOR:

M.I.E. CARLOS ENRIQUE MACIEL GARCIA

CD. GUZMÁN JALISCO, MÉXICO, AGOSTO DE 2018

AGRADECIMIENTOS

Este trabajo representa la culminación años de estudio, de esfuerzo y perseverancia. Es por eso que quiero aprovechar para agradecerles:

En primer lugar a mi familia la cual me ha apoyado para continuar estudiando aunque existan adversidades.

A mi padre Jorge Sánchez Rodríguez el cual siempre ha visto por sus hijos antes de sus necesidades y mi nueva madre Faviola Buelna Alvarado que me ha adoptado como un hijo y dado su amor de madre.

A mi prometida E. Viridiana Miranda Barón y su familia que me han acogido en su casa, me han tratado de una manera maravillosa y aceptado ser casi parte de su familia.

A mi director de tesis el Dr. Humberto Bracamontes Del Toro y mi codirector el M.I.E Carlos Enrique Maciel García los cuales me motivaron a estudiar la maestría en Ingeniería Electrónica.

Al Instituto Tecnológico de Ciudad Guzmán y todos mis compañeros y amigos del mismo instituto.

Mención especial al Consejo Nacional de Ciencia y Tecnología (CONACYT) que apoyó mis estudios de maestría.

Y gracias a todos por creer en mí.

DEDICATORIA

Esta tesis la dedico primeramente a Dios por guiarme, porque me diste la oportunidad de vivir y de regalarme una familia y amigos maravillosos.

A mis padres, porque aunque yo les he dicho que soy independiente ellos están siempre para seguirme apoyando.

A mi madre que en paz descanse porque seguramente uno de sus sueños fue el de que yo fuera un hombre de bien y con formación académica.

A todos los maestros y personas que enseñaron algo en mi vida.

Y quiero hacer una dedicatoria muy especial a la persona que ha estado antes de comenzar esta tesis y el posgrado, una persona que amo, que me ha mostrado a no rendirme y me ha hecho ser una mejor persona, mi prometida, E. Viridiana Miranda Barón, gracias por todo.

LISTA DE SÍMBOLOS Y/O NOMENCLATURA

DIY:	Do iy yourself
DJI:	Dà-Jiāng Innovations
DRONE:	Dynamic Remotely Operated Navigation Equipment
ESC:	Electronic Speed Controller
GPS:	Global positioning system
I/0:	Input/Output
I2C:	Inter-Integrated Circuit
IMU:	Inertial measurement unit
LABVIEW:	Laboratory Virtual Instrument Engineering Workbench
LCD:	Liquid-crystal display
LiPo:	Litium Polymer
MPU:	Multiple Process Unit
OLED:	organic light-emitting diode
PAM:	Modulación por amplitud de pulsos
PCM:	Pulse Code Modulation
PDM:	Pulse-density modulation
PID:	Proporcional-Integral-Derivativo
PPM:	Pulse Position Modulation
PWM:	Pulse Width Modulation
RPAS:	Remotely Piloted Aerial System
RTH:	Return To Home
RTL:	Return To Launch
UART:	Universal Asynchronous Receiver-Transmitter
UAS:	Unmanned Aerial System
UAV:	Unmanned Aerial Vehicle
VANT:	Vehículo Aéreo No Tripulado

RESUMEN

Un vehículo aéreo no tripulado o comúnmente conocido como “drone” es una aeronave que vuela sin tripulación. Hoy en día se ha popularizado su uso en ámbitos de Fotografía, Vídeo así como muchas aplicaciones que involucran imágenes. Para su uso se han desarrollado algoritmos que permiten misiones planificadas o mediante una persona que realice la tarea de piloto. Hay actividades en la que un piloto no podría ser suficiente para controlar el dispositivo, o actividades donde una persona o un objeto deben desplazarse y un piloto humano se vuelve un inconveniente por lo que se propone un sistema que permitan sustituir a un piloto, por un control asistido para la grabación de video o de fotografía así como el seguimiento espacial del objeto, esta aplicación es también comúnmente conocida como “follow me”, ya que se encarga de seguir aquello que se le indica, mediante el control asistido.

Un control asistido o un piloto automático es una función que disponen algunos medios de transporte como los aviones, barcos, y recientemente coches, que permiten que el vehículo funcione por sí mismo sin la necesidad del control permanente del conductor o piloto, en un UAV comercial se requiere la necesidad de un piloto que este constantemente monitoreando y controlando el equipo.

Se diseñaran un par de dispositivos que sin alterar la programación interna de una controladora de vuelo para UAV ya sea de tipo comercial u open source, permitan a un usuario realizar la función de posición del UAV de forma automática en altura, orientación, latitud y longitud geográfica de tal forma que cumpla con una función de piloto automático.

Como posibles resultados se realizara un trabajo de investigación, programación y tendremos un prototipo UAV que sea capaz de ser preprogramado para seguir a una persona mediante diversa adquisición de datos con magnetómetros digitales, acelerómetros, giroscopios, GPS y barómetros, de forma precisa, que logre volar y hacer que el UAV realice la función de “follow me”.

ABSTRACT

An unmanned aerial vehicle or commonly known as "drone" is an aircraft that flies without a crew. Nowadays, it has become popular in photography, video and many applications that involve images. For its use, algorithms have been developed that allow planned missions or through a person who performs the pilot task. There are activities in which a pilot could not be enough to control the device, or activities where a person or an object must move and a human pilot becomes an inconvenience, so a system is proposed to replace a pilot, for a Assisted control for video recording or photography as well as spatial tracking of the object, this application is also commonly known as "follow me", as it is responsible for following what is indicated, through assisted control.

An assisted control or an autopilot is a function that some means of transport have, such as airplanes, ships, and recently cars, which allow the vehicle to operate on its own without the need for permanent control of the driver or pilot, in a commercial UAV the need for a pilot who is constantly monitoring and controlling the equipment is required.

A pair of devices will be designed that without altering the internal programming of a flight controller for UAV, whether commercial or open source, allow a user to perform the position function of the UAV automatically in height, orientation, latitude and longitude. geographically in such a way that it fulfills an automatic pilot function.

Possible results will be a research work, programming and we will have a prototype UAV that is capable of being preprogrammed to follow a person through diverse data acquisition with digital magnetometers, accelerometers, gyroscopes, GPS and barometers, in a precise way, that achieves fly and have the UAV perform the "follow me" function.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

INTRODUCCIÓN

El uso de vehículos aéreos no tripulados lleva muchos años desarrollándose, aunque trataban de realizar algunas de las funciones que se proporcionan hoy en día, su modelado era diferente, desde lo más simples como es el caso de los papalotes y globos aerostáticos, los cuales entran también a la categoría de bombas aéreas que se utilizarían para fines bélicos con el objetivo de observación y ataque, hasta la etapa actual donde se hace uso de las innovaciones tecnológicas, empleando sensores, motores, microprocesadores, sistemas de control, sistemas de comunicación inalámbrica y de almacenamiento de energía para llegar a un fin objetivo y potencial. Los UAVs pueden ser clasificados por su tamaño, su aplicación o los dos. El tamaño es un criterio dominante ya que dependiendo de este todo el sistema que lo involucrara deberá ser factible para su correcto funcionamiento. Y a la vez se clasifican en base a su funcionamiento aerodinámico. En los que encontramos los de ala fija (aeroplanos), ala rotatoria (helicópteros) y de ala flexible (parapente y ala delta).

La presente tesis propone un par de dispositivos que no requieran cambiar grandes aspectos de la programación interna de controladoras de vuelo comerciales para desarrollar una nueva aplicación en el mismo donde no se requiera de la intervención de un piloto para la manipulación de la misma, que un usuario pueda realizar alguna actividad y no requiera la atención completa en la aeronave, lo que implicó un estudio minucioso acerca de los movimientos de la aeronave, funcionamiento y control por medio de diversos sensores; a partir de lo anterior se diseñó un prototipo para la posterior integración de la electrónica mediante microcontroladores.

Se diseñó una tarjeta controladora basada en el uso de sensores como acelerómetro, giroscopio y magnetómetro para que fuera procesado por un microcontrolador, el cual fue programado para realizar el seguimiento de una persona u objeto. Una vez finalizado los dispositivos de control se incorporó en el vehículo de prueba para realizar experimentos de vuelo y ajustar ganancias de vuelo, asegurando un óptimo funcionamiento.

1.1. OBJETIVOS

1.1.1. Objetivo general

Diseñar un dispositivo que sea compatible con distintas plataformas, tanto comerciales como open source, que permita al usuario lograr un piloto automático de su UAV para tareas como grabado de video y manipulación del UAV.

1.1.2. Objetivos particulares

- Buscar y comparar las mejores alternativas en sensores como lo son magnetómetros, brújulas, GPS, giroscopios, acelerómetros y barómetros, así como de probar el funcionamiento de los mismos en diferente hardware de sistemas embebidos.
- Realizar censado de altura, campo magnético, latitud y longitud.
- Establecer un filtro de variables
- Ejercer comunicación bidireccional en protocolo Serial.
- Crear una aplicación de lectura de señales múltiples PWM.
- Establecer la escritura de señales PPM.
- Implementar un controlador para altura, dirección y posición geográfica de un UAV.
- Desarrollar el sistema en un microcontrolador de 32 bits.
- Registrar la visualización de caracteres en un display OLED.
- Elaborar PCBs para el sistema no invasivos respecto a tamaño y peso.
- Diseñar una plataforma para PC para visualizar el estado de los dispositivos.

1.2. PLANTEAMIENTO DE PROBLEMA

Existen en el mercado diferentes dispositivos que ofrecen en sus productos modos de funcionamiento para mantener la altura y posición geográfica sin embargo algunas están limitadas o son productos de costos muy elevados y enfocados solo para trabajar bajo sus especificaciones de UAVs.

En la actualidad se necesitan vehículos capaces de desempeñar actividades más complejas y específicas y que se logran mediante el desarrollo de más software y hardware, incorporando en ellos cámaras y sensores con la finalidad de vigilancia y reconocimiento facial y espacial.

Un vehículo aéreo no tripulado (UAV) presenta un alto consumo energético actualmente existe una gran variedad de vehículos tipo multirotor pero la solución que les dan es incluir baterías de mayor capacidad lo que va afectando peso soportado para elevación y deriva en diversos factores, según la necesidad final del consumidor, por lo que nuestra necesidad es buscar que el nuevo prototipo cumpla la necesidad de multicoptero y planeador asi que necesitaremos alterar el sistema de control convencional proponiendo nuevos modelos de vuelos materiales y comprobando la eficiencia que dará el desarrollo del prototipo e investigación.

1.3. JUSTIFICACIÓN

Los cambios o innovaciones tecnológicas demandan una ardua investigación para facilitar el trabajo que ejercen las personas durante su existencia laboral, ya sea militar o civil por lo que es necesario la mejora en el consumo eléctrico en drones o UAV ya que son una herramienta de trabajos indispensables para distintos trabajos como son, búsqueda de personas desaparecidas, así como de algunos objetos materiales, fotografía, video y cartografía aérea; prevención y control de incendios; seguridad y aplicaciones militares; en cuidado del medio ambiente; agricultura algunas aplicaciones; diseño de controladores o cualquier tarea en las que se requiera transportar un objeto de manera aérea.

Todo lo anterior es posible, debido a los grandes avances tecnológicos y que día a día se aporta una nueva investigación, que finalmente se pueden usar para llegar a un fin común y dar mejores alternativas mediante la aplicación de métodos sencillos y eficaces, disminuyendo el número de peligros en las actividades en las que se ponga en riesgo la vida de una persona.

1.4. HIPÓTESIS

Comprobar que el desarrollo de una plataforma móvil UAV y un sistema de control, proporcionan a un sistema de UAV más autonomía y que permite a un usuario realizar funciones como que el UAV sea capaz de un vuelo de manera más autónoma y este siempre en dirección a un usuario aunque esta se mueva, así como mantener una posición fija y pueda realizar un aterrizaje donde el usuario no tenga que realizar algún control, el sistema deberá poder ser capaz de ser agregado a un UAV que ya sea estable en vuelo y que no requiera modificación en el software interno de la controladora de vuelo.

1.5. METODOLOGÍA

Se realizó la revisión del estado del arte de los UAV, principios básicos de vuelo, características físicas de los UAV tipo QuadRotor, su clasificación por tamaño, evolución histórica y aplicaciones actuales, las tareas que cumple por definición, la descripción y características de los rotores, hélices, el sistema de posicionamiento global y tarjetas de telemetría. Se explicara el funcionamiento de un QuadRotor, se define su estructura, se describe la secuencia en que realiza el despegue, elevación/aterrizaje, movimiento sobre los ejes X, Y, Z, la descripción de su comportamiento dinámico y las ecuaciones de navegación que se utilizan para el control.

Posteriormente el "Modelado, Análisis, y Desarrollo del QuadRotor", se describen los componentes que conforman al QuadRotor, el fuselaje, rotores, la instalación y configuración de la electrónica interna del QuadRotor, la batería y su relación con los rotores y los controladores electrónicos de velocidad, así como la descripción e instalación del sistema de adquisición de video y su sistema de transmisión, además de la descripción y configuración del hardware y software a utilizar en la base terrena. Finalmente la "Implementación, pruebas y resultados" se realizaró una revisión de las reglas de seguridad, así como la calibración y pruebas de funcionamiento, las pruebas manuales y pruebas automáticas, llegando al informe de los resultados y las conclusiones.

1.6. METAS

- Diseñar la etapa de censado.
- Programación de los microcontroladores de la etapa del convertidor analógico – digital que realiza el censado.
- Diseñar la etapa del transmisor y receptor, para posteriormente hacer las pruebas correspondientes.
- Interpretación de los datos adquiridos.
- Desarrollo de sistemas de monitoreo remotos y locales de monitoreo.
- Implementar un sistema de monitoreo de consumo de energía.
- Realizar pruebas y recabar información necesaria.
- Realizar la redacción del reporte de protocolo.
- Publicar artículos en congresos

1.7. IMPACTO O BENEFICIO

1.7.1. Aplicación en Agricultura

Una de las aplicaciones con más potencial, además de la seguridad pública, es la agricultura de precisión y la monitorización de los campos. Así se recoge en el estudio "El impacto económico de la integración del Sistema Aeroespacial No Tripulado en la economía de EE UU"[51].

La agricultura de precisión, en síntesis, consiste en el empleo de nuevas tecnologías como es el caso de GPS, sensores planta clima e imágenes multiespectrales con la finalidad de otorgar un estudio detallado de la parcela, de manera que pueda dar objetividad a la actividad que desee realizarse, como es el caso de plagas o eterminado momento para la detección de incendios.

La agricultura de precisión empezó a estudiarse en los años ochenta, pero ha sido a partir del nuevo siglo cuando el desarrollo tecnológico y sobre todo el acceso barato a la tecnología han permitido su despegue definitivo.

Los beneficios de la agricultura de precisión son triples, permite reducir costes, mejora la rentabilidad de los cultivos y disminuye el impacto ambiental, ya que la aplicación de agroquímicos es dirigida y ajustada a los requerimientos reales de cultivo.

1.7.2. Ámbito fotográfico de recreación

En años recientes los drones se han popularizado como artefacto de recreación pero también de creatividad. En la medida en que permiten tomar fotografías y grabar videos, se trata de instrumentos que de algún modo han abierto nuevos caminos de exploración en el dominio de la expresión visual.

De cualquier modo, más allá del proyecto, las imágenes demuestran que, en efecto, cierto ámbito de la fotografía podría renovarse por completo por el uso del drone, lo mismo en la fotografía de paisaje que la de la naturaleza o la social y cultural, demostrándose así, una vez más, que la tecnología puede ser vehículo de la creatividad.

1.7.3. Seguridad privada y militar

En cuestión de seguridad privada, se han logrado grandes avances mediante el uso de un vehículo aéreo no tripulado que lleva consigo una cama de alta definición y sensores de gran alcance, con el objetivo lograr una búsqueda minuciosa de personas como es el caso de desastres naturales dónde es imposible ingresar al área, también tiene uso en vigilancia policiaca-actividades ilegales.

En cuanto al área militar tuvo inicio en los años 90´s durante la guerra de Irak con el

objetivo de realizar espionaje y crear ataques objetivos hay bibliografías que mencionar el uso de plataformas aéreas no tripuladas con inicio en el año de 1849 donde ejemplifican estas aeronaves como globos aerostáticos los cuales lanzaban explosivos por lo que refieren el concepto como "plataforma no tripulada que porta una carga útil".

El uso de los UAV's está siendo aplicada en países de primer nivel, como Japón el cual cuenta con una empresa denominada SECOM, el cual emplea la tecnología de tal manera que lo programan para que siga a personas sospechosas, utilizando un sistema de detección de movimiento y un láser para medir la distancia del objetivo [50]

1.7.4. Sistemas comerciales FOLLOW-ME

Follow-me, es una tecnología que se está siendo implementada en el área de fotografía, la cual hace lo que la persona física u objeto en movimiento realice y que se mantenga encima de él, realice giros y fotografié en intervalos de tiempos, para esta área resulta muy eficaz, ya que de esta manera el profesional puede estar dedicado a tomar fotografías sin preocuparse por el control del drone [47].

Capítulo 2

MARCO TEÓRICO

2.1. SENSORES

2.1.1. Barómetro

El barómetro fue inventado en 1634 por Evangelista Torricelli; es un instrumento meteorológico utilizado para medir la presión atmosférica. En la actualidad existen varios tipos de barómetros como lo son: Barómetro de mercurio, Barómetro de Fortín, Barómetro aneroide, Barómetro holostérico (ver figura 2.1); los cuales funcionan o miden la presión atmosférica basándose en la deformidad que se genera en una pequeña caja metálica y que puede verse en una superficie, dicha caja en su interior se encuentra al vacío, por lo que no hay una presión interna que contrarresta la presión de aire sobre ella, lo cual hace más sensible las variaciones.

Barómetro BPM085

El sensor BPM085 (ver figura 2.2) además de tener la función o la habilidad de medir la presión atmosférica también puede ser utilizado para conocer la altitud a la que ese encuentra el drone y al mismo tiempo este cuenta con un sensor de temperatura fiable9.

El barómetro BPM085, cuenta con las siguientes características [20]:

- Rango de alimentación: 1.8 a 3.6V
- Rango de operación de 300 a 1000 (9000 a 500m sobre el nivel del mar)
- Resolución de 0.03hPa/0.25m
- Temperatura de operación -40 a $>$ 85°C, precisión de $\pm$2°C

El barómetro BPM085 tiene las siguientes funciones:

Figura 2.1: Tipos de Barómetro. (Fuente: Elaboración propia)

- Read temperatura: devuelve una lectura de la temperatura, en grado centígrado.
- Read pressure: devuelve la lectura de la presión atmosférica en pascales.
- Read altitude: devuelve la altitud a la que se encuentra [3]

Este barómetro se basa en la tecnología piezo-resistiva de alta precisión, de gran y largo plazo de estabilidad y se conecta a un microcontrolador a través de un bus I2C y cuenta con una librería Arduino [46].

Barómetro MS5611

El MS5611 es una nueva generación de sensores de altímetro de alta resolución con interfaz de bus I2C y SPI. Cuenta con un sensor de presión optimizado para altímetros y variométricos, el módulo del sensor además incluye un sensor de presión además, el consumo energético es ultrabajo.

El MS5611 (ver figura 2.3) tiene las siguientes características:

- Módulo de alta resolución: 10 cm
- Conversión rápida de 1ms
- Baja potencia, 1μA (espera <0.15 μA)
- Paquete QFN de 5.0mm3 x 3.00mm3 x 1.0mm3
- Voltaje de alimentación: 1.8 V a3.6 V
- Sensores de presión digital integrado (ADC $\Sigma\Delta$ de 24bits)

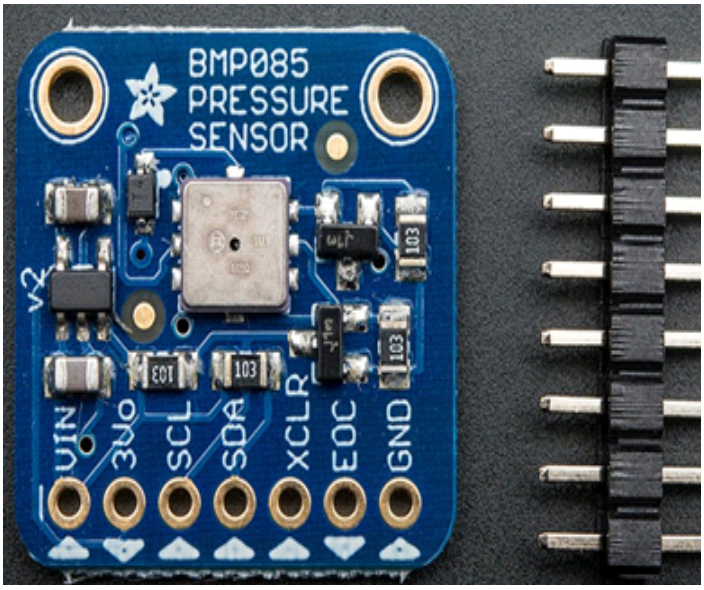

Figura 2.2: Barómetro BPM085. (Fuente: https://www.adafruit.com/product/391)

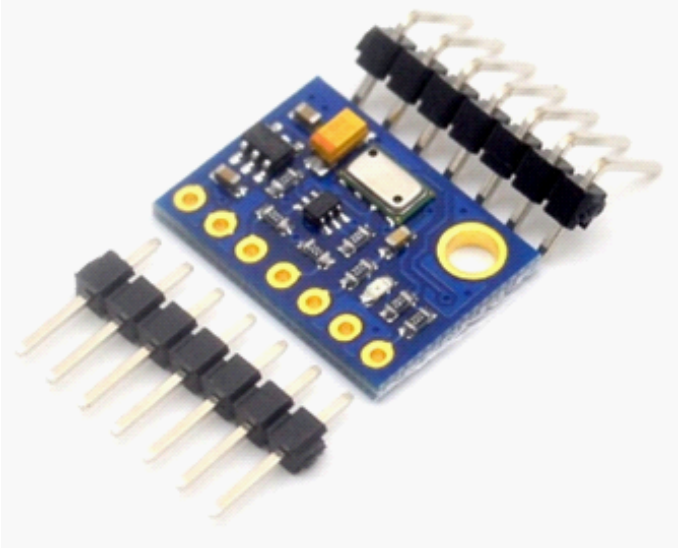

Figura 2.3: Barómetro MS5611. (Fuente: Elaboración propia)

- Rango de operación: 10 a 1200mbar, -40°C a +85°C
- Interfaz I2C y SPI de hasta 20 MHz
- Oscilador interno

Las aplicaciones del barómetro MS 5611 son variadas, entre ellas se encuentra su uso en sistemas de barómetro y alímetro móvil, computadoras para bicicletas, variómetros, detección de altura de alarmas médicas, navegación interior, entre otros. El MS5611 puede interconectarse a cualquier microcontrolador y posee tecnología MEMS.

2.1.2. Sistema de Posicionamiento Global (GPS)

El GPS es el conjunto de elementos (software y Hardware) que permiten realizar la posición, velocidad, y tiempo de usuario, además de los parámetros necesarios adicionales que requiera. El GPS tiene su inició en el año de 1957, cuando la Unión Soviética lanza un satélite al espacio el cual era monitorizado gracias al efecto doppler de la señal que transmitía. Posteriormente fue utilizado por las fuerzas armadas de los Estados Unidos para proveer la navegación de sus flotas, conocer la posición actual y precisa.

El sistema de navegación por satélite está compuesta por (ver figura 2.4) [23]:

- Segmento espacial: En el que se encuentran 24 satélites con trayectorias sincronizadas las cuales cubren el globo terrestre.
- Segmento control: Estaciones terrestres, que tienen la función de enviar información de control a los satélites con la finalidad de inspeccionar las orbitas y organizarlas.
- Segmento del usuario: El cual, representa al instrumento.

En la actualidad se han creado dispositivos de gran exactitud en tiempo y espacio, menor tamaño y que ofertan mejores opciones para el usos de esta tecnología dentro de la industria de investigación.

GPS NEO 6M

La serie de módulos NEO-6 (ver figura 2.5) es una familia de receptores GPS independientes que ofrecen el alto rendimiento en el motor de posicionamiento u-blox, este motor cuenta con con un Time-To-First-Fix (TTFF) de menos de un segundo [12].

El GPS NEO 6-M es compatible con el protocolo NMEA. Este GPS ya conectado a la interfaz serial, enviará cada segundo una serie de comandos siguiendo dicho protocolo (NMEA), de tal manera que el usuario debe implantar un software que sea capaz de reconocer dichos comandos[39]. En el cuadro 2.1 se muestran las especificaciones del GPS NEO 6-M [45].

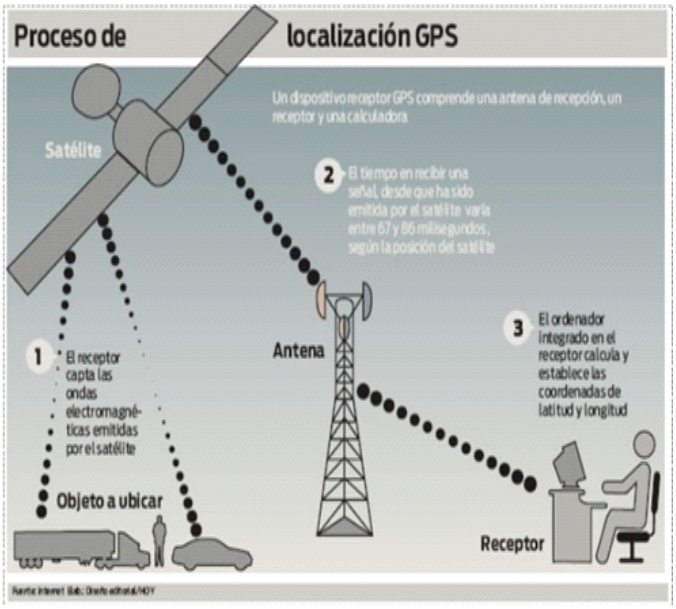

Figura 2.4: Proceso de localización GPS. (Fuente: https://es.slideshare.net/ ojmb7777777/ trabajo-de-gps)

Figura 2.5: GPS NEO 6-M. (Fuente: Elaboración propia)

Voltaje de alimentación: 3-5VDC
Interface : Serial UART 5V
Antena cerámica
Batería de respaldo: MS621FE
Frecuencia de refresco: 5Hz
Soporta SBAS
Indicador de señal con LED
Tamaño de antena: 25 mmx25mm
Tamaño de módulo: 25 mm x 35 mm
Tamaño de agujero de montaje: 3 mm
Baud rate por defecto: 9600 bps

Cuadro 2.1: Especificaciones de GPS NEO 6-M

GPS M8N

Este modelo (ver figura 2.6) incorpora una brújula digital HMC5883L, el cual usa un módulo de 8 series de Ubox, que proporciona un método útil para posicionar a la brújula lejos de fuentes de interferencia que pueden estar presentes en los confines del vehículo. Presenta un circuito activo para una antena de parche de cerámico y una batería de respaldo recargable para inicios cálidos.

El NEO M8N incluye un flash interno que permite actualizaciones de firmware para administrar sistemas GNSS adicionales (ver cuadro 2.2). La interfaz DDC compatible con I2C proporciona conectividad y permite sinergias de mayoría de celulares u-blox [24].

Magnetómetro

Un magnetómetro es un instrumento que mide la intensidad y, en ocasiones, la dirección de un campo magnético, fue inventado en 1833 por Carld Friedrich Gauss. Los magnetómetros se usan para la medición del campo magnético terrestre, y en estudios geofísicos se utilizan para detectar anomalías magnéticas de diferentes tipos.

Los magnetómetros se clasifican según las magnitudes que puedan medir:

- Magnetómetros escalares: Estos miden la intensidad total del campo magnético resultante en un punto, pero no aportan ningún dato sobre las componentes vectoriales de campo, como ejemplo se encuentra el magnetómetro A1391 (ver figura 2.7).

- Magnetómetros vectoriales: Estos tienen la capacidad de medir la intensidad del campo magnético a la que están sometidos además, de medir una dirección particular [42]

En la actualidad hay una amplia gama de dispositivos inteligentes como móviles, GPS, tabletas, etc. en los que se encuentran incorporada una brújula mediante la cual, se logra

Figura 2.6: GPS M8N. (Fuente: Elaboración propia)

GNSS FEATURES	
GNSS	BeiDuo, Galileo, GLONASS, GPS/QZSS
Number of concurrent GNSS	3
Concurrent GNSS	si
Oscillator	TCXO
ANTENA	
Pin (active) for external	Si
Pin (passive) for external	Si
INTERFACE	
UART, USB, SPI, DDC compatible I2C	Sí
ELECTRICAL DATA	
Minimum supply	2.7
Maximum supply	3.6
EVIRONMENTAL DATA, QUALITY, & RELIABILITY	
Maxima temperature	85
Minima temperature	-40

Cuadro 2.2: Comparaciones

Figura 2.7: Magnetómetro A1391. (Fuente: https://www.5hertz.com/ index.php?route= tutoriales/ tutorial&tutorial_id=1)

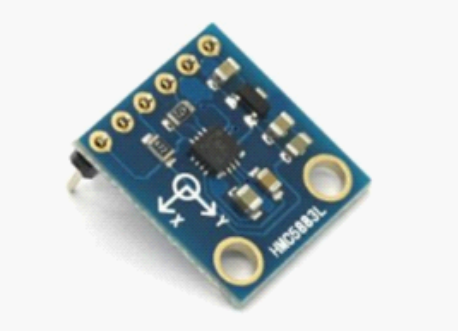

Figura 2.8: Módulo brújula digital HMC5883L. (Fuente: Elaboración propia)

orientar el dispositivo, detectando la posición actual y la transforma a su posición vertical que tiene el móvil o la dirección del polo norte en el GPS.

El magnetómetro digital es un sensor que mide el valor del campo magnético en función de tomar tres ejes, con lo que es posible estimar la orientación del dispositivo respecto al campo magnético de la tierra.

En el proyecto usaremos el magnetómetro HMC5883L (ver figura 2.8), el cual es un sensor de 3 ejes, se encuentra integrado en módulos como GY-273 para integrarlo a Arduino.

Este módulo concentra 3 sensores de magnetorresistencia, cancelación de desfases y conversores de 12 bits, lo que provee una precisión de ±2°C. El módulo de Honeywell HMC5883L, se comunica a través del bus I2C, cuya dirección es ox1E y demanda dos hilos de comunicación: uno para un reloj (SCL) y otro para los datos (SDA)16 (ver figura 2.9).

Especificaciones:

- Interfaz I2C simple

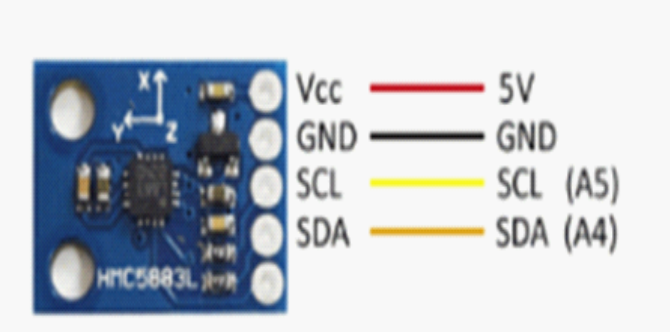

Figura 2.9: Conexiones. (Fuente: Elaboración propia)

- Rango de suministro de 2.16-3.6VDC
- Bajo consumo de corriente: 3 a 5 volts
- Resolución de 5mili-gauss
- Dimensiones: 0.7 x 0.7" (17.78 x 17.7mm)[44]

Aplicaciones

- Magnetometría
- Brújula electrónica de bajo costo
- Robots con navegación autónoma
- Servicio basados en localización (LBS)[21]

Se designa el nombre de acelerómetro a todo aquel instrumento consignado a medir aceleraciones, todos estos están construidos con piezas móviles con la finalidad de facilitar los cambios de posición de acuerdo a la aceleración que se aplica.

El acelerómetro mide la aceleración, inclinación o vibración y transforma la magnitud física de aceleración en otra magnitud eléctrica.

Un giroscopio es un dispositivo mecánico formado básicamente por un cuerpo con simetría de rotación, es decir, que tenga la capacidad de girar alrededor de su eje. La estructura básica de un giroscopio es una rueda que gira a gran velocidad [40].

Cuando la rueda gira a alta velocidad hace que su eje de rotación permanezca en una posición estable y sea difícil moverlo de esta posición [41]. El giroscopio muestra el cambio de rango en rotación en sus ejes X, Y, Z23.

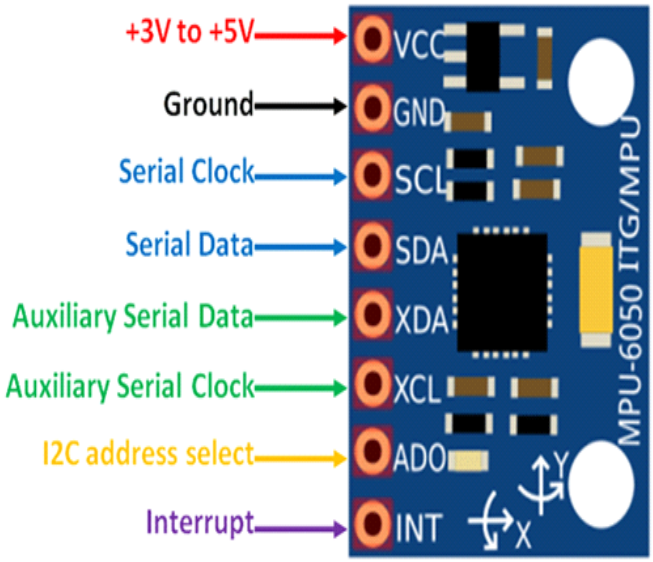

Figura 2.10: Acelerómetro y giroscopio MPU6050. (Fuente: https://components101.com/sensors/ mpu6050-module)

Estos sensores son utilizados en algunas aplicaciones como: Estabilización, pilotos automáticos, navegación, plataformas de vuelo.

El MPU6050 es una unidad de medición inercial, por sus siglas en inglés (IMU) de 6 grados de libertad ya que este circuito armoniza un giroscopio de 3 ejes, un acelerómetro de 3 ejes y un procesador digital de movimiento DMP capaz de procesar complejos algoritmos de 9 ejes de movimiento[2].

El circuito integrado MPU6050 (ver figura 2.10) contiene un acelerómetro y giroscopio MEMS en un solo empaque. Cuenta con una resolución de 16 bits , lo cual significa que se divide en 65536 fracciones, estos aplican para cada eje X, Y y Z, al igual que en la velocidad angular [25]. Dicho sensor es muy utilizado en navegación, gonometría, estabilización, etc.

Y es ideal para diseñar control de robótica, medición de vibraciones, sistemas de medición inercial (IMU), detector de caídas, sensor de distancias y velocidad (ver cuadro 2.3).

2.2. PROTOCOLOS DE COMUNICACIÓN

Las comunicaciones con el drone y la emisora pueden a pertenecer a diferentes protocolos de comunicaciones, existen 2 tipos:

Salida digital de 6 ejes
Giroscpio con sensibilidad de $\pm 250, \pm 500, \pm 1000, \pm 2000 dps$
Acelerómetro con sensibilidad de $\pm 2g, \pm 4g, \pm 8g, y \pm 16g$
Algoritmos embebidos par la calibración
Sensor de temperatura digital
Entrada digital de video FSYNC
Interrupciones programables
Voltaje de alimentación: 2.37 a 3.46V
Voltaje lógico 1.8V $\pm$ 5 % o VDD
100000G tolerancia de aceleración máxima

Cuadro 2.3: Especificaciones MPU 6050

- Trasmisores o emisoras que envían los movimientos
- Receptores que se comunican con la controlador de vuelo

2.2.1. Comunicación Serial

Las comunicaciones se remonta al año de 1800 cuando fue descubierta por primera vez por Von Sommering, quien utilizó 26 cables pegados a la parte inferior de un acuario y se demostraba la transmisión de energía a través de ellos cuando se generaban burbujas. De esta misma manera se produjo el código Morse, el cual es un código binario: espacio y marca; y en términos de computadora se usan los valores 0 y 1 dependiendo de cada bit; todo lo anterior nos lleva a que se habla de comunicación serial porque los bits se reciben uno seguido de otro o en serie.

La comunicación serial tiene dos categorías (ver figura 2.11):

- Comunicación serial síncrona: Requiere de una línea de reloj y una línea de datos
- Comunicación serial asíncrona: Los dispositivos se ponen de acuerdo en la velocidad de la comunicación, uno de ellos recibe y otro trasmite y viceversa.

En la Figura 2.11 Se muestra la comunicación asíncrona el cual consta de dos sistemas y cada uno tiene un reloj, su sincronización es mediante un bit de Start y stops , por lo que solo requiere un hilo, en la imagen inferior se ejemplifica como es una comunicación serial ,a cual tiene un reloj común y que está posicionando en el sistema que funge como maestro , este sistema requiere de dos hilos.

Para la sincronización se precisa de un bit extra el cual es manipulado por el emisor y el receptor, de tal manera que por aquí se intercambiara la señal de pulso; es importante, recordar que la trasmisión serial no será ejecutable atreves de un cable de dos líneas debido que ambos están ocupados por los datos y la tierra, por tal motivo es necesario el uso del

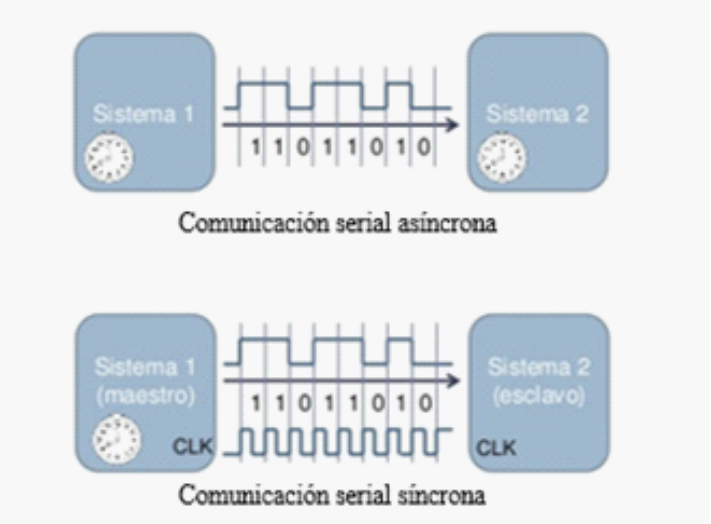

Figura 2.11: Comunicaciones seriales. (Fuente: https://es.slideshare.net/ JonathanRuizde-Garibay/ 09comunicacin-serie)

protocolo RS-232 para realizar la programación correspondiente de los puertos a partir de los siguientes datos:

- Bit de inicio: El receptor al detectar el bit de inicio reconoce que la transmisión a comenzado y de esta manera sabe que debe leer las señales a distancia, tiempo y velocidad.
- Bit de parada: Se refiere a la finalización de la transmisión.
- Bit de paridad: Este bit es utilizado para conocer los errores de trasmisión [15].

En la figura 2.12 se muestra una transmisión asíncrona donde por cada carácter se envía al menos un bit de inicio y un bit de parada; así como opcional un bit de paridad.

En la figura 2.13 Se muestra una transmisión síncrona, donde el primer dato que se envía es un octeto de sincronismo ("sync"), dicho octeto realiza la misma función que un bit de inicio en el caso de una transmisión asíncrona , el cual indica al receptor que va realizar enviado el mensaje y la señal se muestrea con frecuencia lo que permite sincronizar los relojes del transmisor y el receptor.

Un puerto serial es un módulo de comunicación digital para un sistema embebido, es decir, aprueba la comunicación entre dos dispositivos digitales y la función primordial del puerto serial es empacar y desempacar paquetes de datos binarios. Al módulo serial también se le conoce como:

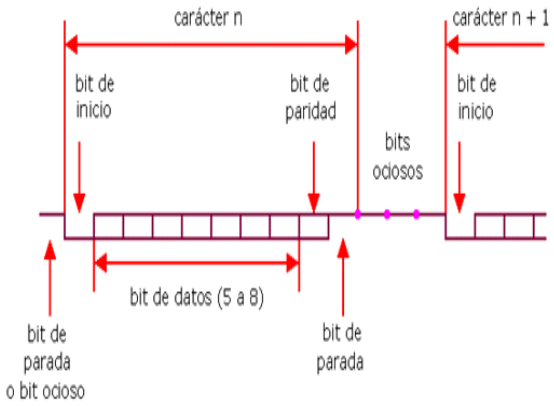

Figura 2.12: Transmisión asíncrona. (Fuente: http:// perso.wanadoo.es/ pictob/ comserie.htm)

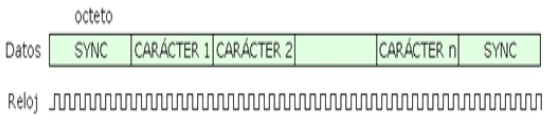

Figura 2.13: Transmisión síncrona. (Fuente: http:// perso.wanadoo.es/ pictob/ comserie.htm)

- Universal Asyncronos Receiver and Transmitter (UART): Transceptor asincrono
- Universal syncronos and asyncronos Receiver and Transmitter (USART): Transceptor sincrono y Asincrono
- Enhanced Universal Asyncronos Receiver and Transmitter: Transceptor Asíncrono Universal Mejorado[26]

El modo de comunicación de un IUART tiene separada las líneas de transmisión y recepción, de tal manera que esta característica le permite trabajar de 3 modos de comunicación asíncrona:

- Full-duplex: Transmite y recibe simultáneamente
- Half-duplex: Solo transmite o solo recibe
- Simplex: Solo transmite información binaria[36].

2.2.2. Comunicación I2C

I2C significa Inter Integrated Circuit, es decir comunicación entre circuitos integrados, fue fundado a inicios de la década de 1980 por Philips y desde entonces, se ha convertido en el bus serial estándar. Este sistema contiene un canal semibidireccional el cual se utiliza para que el sistema central defina como se comportará tal circuito, es decir como transmitir órdenes y por otro lado para trasmitir los datos que sean precisos en uno u otro sentido[1].

Está interfaz contiene 2 señales:

- Serial Data (SDA). Es la línea de datos de serie.
- Serial Clock (SCL).Es la señal de sincronía.

En la figura 2.14 Se muestra una comunicación I2C dónde se observa la capacidad de tener un maestro y múltiples esclavo en un m ismo bus. El bus requiere dos líneas para trasmitir la información: una para datos (SDA) y una segunda para la señal de reloj (SCL). La misma línea de datos enviará la información en las dos direcciones (half-duplex), por lo que se necesita un control de acceso y un direccionamiento de cada elemento.

Además la interfaz de comunicación I2C puede encontrarse en varios estado, es decir puede ser libre, en el cual no se expresen transferencias; inicio, donde se realiza un cambio generado por la transacción y hace un cambio de alta a baja SDA y SCL permanece intacto; dato, se refiere a que no puede haber ningún cambio después de iniciarse una transacción, y finalmente parada, cuando se hace la regresión de SDA de baja a alta. La

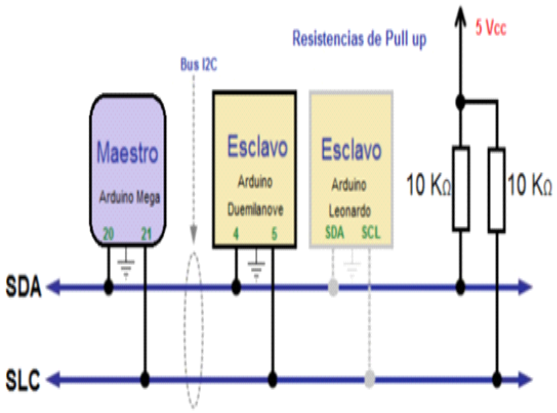

Figura 2.14: Esquema eléctrico de comunicaciones I2C. (Fuente: https://ardubasic. wordpress.com /2014/07/28/ comunicacion-i %C2 %B2c/)

interfaz o comunicación I2C tiene la función de comunicarse y configurar los sensores que son compatibles con este protocolo de comunicación.

El I2C es usado en una gran variedad de micro controladores y aplicaciones de comunicaciones como en control, diagnóstico y administración de potencia[35].

2.3. COMUNICACIONES

2.3.1. Comunicaciones digitales

El término de comunicaciones digitales abarca una gran área de técnicas de comunicaciones, que incluye la transmisión digital y el radio digital.

La radio digital es la transmisión de portadoras analógicas moduladas digitalmente entre dos o más puntos de un sistema de comunicaciones. Los sistemas digitales de transmisión demandan una instalación física entre el transmisor y el receptor, es necesario para realizar esta conexión tener un par de hilos metálico, un cable coaxial o un cable de fibra óptica. Aunque también en los sistemas digitales de radio, el medio de transmisión podría ser el espacio libre, la atmósfera terrestre o una instalación física.

La transmisión digital es un sistema digital verdadero, donde los impulsos digitales (con valores discretos, como +5V y tierra) se transfieren entre dos o más puntos en un sistema de comunicaciones[49].

En la figura 2.15, muestra diagramas de bloques de un sistema de transmisiones digitales y uno de radio digital. En la figura a) la fuente original de información puede estar de forma digital o analógica , de tal manera que si la información se encuentra de forma analógica

esta debe convertirse a pulsos digitales antes de transmitirse , y reconvertirse a la forma analógica en al llegar al extremo de recepción; en cambio en la imagen b), teniendo un sistema digital de radio de manera inicial , simplemente la señal moduladora de entrada y salida será en pulsos digitales sin tener que convertirse la señal en ningún momento del sistema.

Modulación analógica de impulsos

Un sistema analógico de comunicaciones es aquel en el cual la energía se transmite y se recibe en forma analógica: una señal de variación continua, como por ejemplo una onda senoidal. En estos sistemas tanto la información como la portadora son señales analógicas.

En la modulación analógica de impulso cada valor de muestra de la señal mensaje hace abrir equitativamente uno de los parámetros de cada impulso, el tren de impulso, así modulado, puede trasmitirse y en el destino se le puede extraer la información o mensaje contenido en ella.

Cada impulso dispone para su transmisión de todo el ancho de banda del canal pero sólo lo ocupa para darte un intervalo $\tau < T_b$ es el periodo del tren de impulso sin modular y τla duración del impulso.

Existen varias formas de modulación analógica de impulsos, pero son 3 las más conocidas y usadas:

- La modulación de Amplitud de Impulsos (Pulse-Amplitude Modulation, PAM), en la cual la altura o amplitud de cada impulso varía en función del valor de muestra de la señal mensaje. El periodo y la duración de los impulsos no cambian.
- La Modulación de Duración o anchura de Impulsos 8Pulse-Duration (Width) Modulatción, PDM o PWM), en el cual la duración de cada impulso varia e función del valor de muestra de la señal de mensaje. El periodo y la amplitud de los impulsos no cambian.
- La Modulación de Posición de Impulsos (Pulse-Position Modulación (PPM), en la cual la posición década impulso varía, respeto a un punto de referencia, en función del valor de muestra de la señal de mensaje. La amplitud y la duración de impulso no cambian.
- Modulación por código de impulso (Pulse code modultion, PCM), en la cual se muestrea la señal analógica y se convierte en un número binario en serie de logitud fija para su transmisión.

La modulación por código de pulso es la única técnica de modulación por codificación digital de la figura 2.16 que se utiliza para la transición digital; las técnicas PWM, PPM,

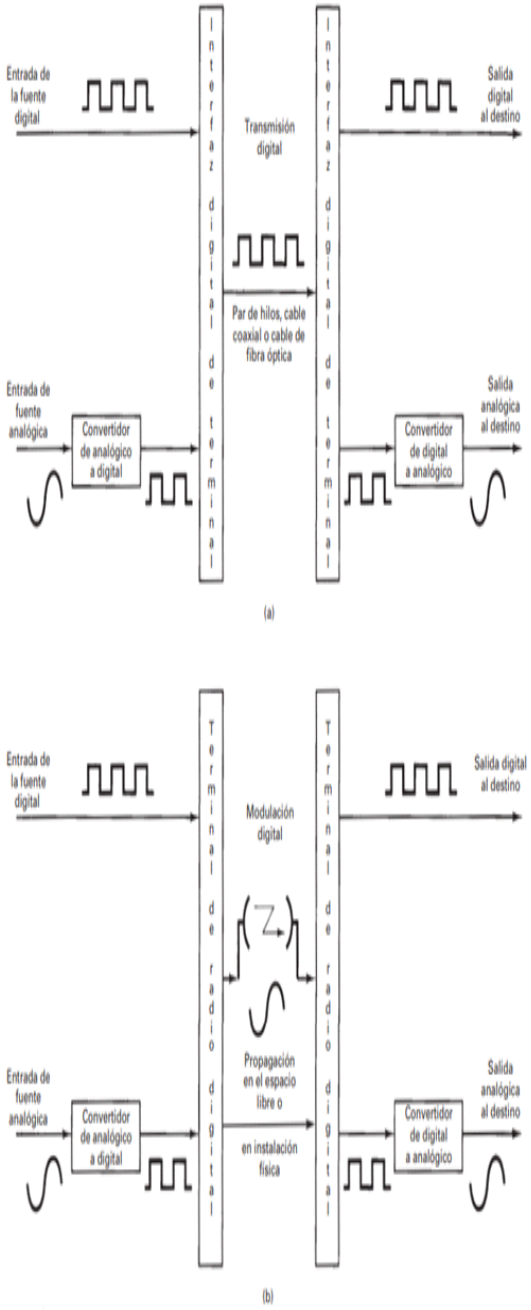

Figura 2.15: Sistemas digitales de comunicaciones: a) Transmisión digital , b) radio digital. (Fuente: https://www.monografias.com/ docs113/comunicacion-digital-redes/ comunicacion-digital-redes.shtml)

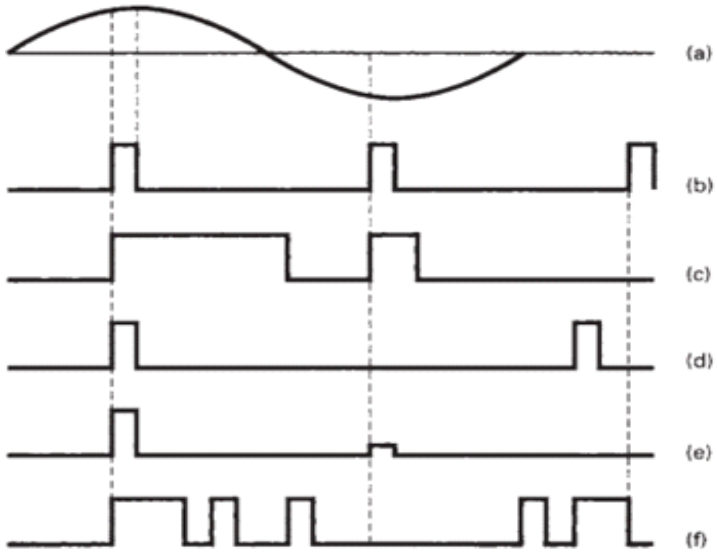

Figura 2.16: Modulación de pulso: a)señal analógica; b) pulsos de muestra; c) PWM; d) PPM; e) PAM; f) PCM. (Fuente: https://www.docsity.com/es/ pam-y-pcm-para-modulaciones- en-los-sistemas-digitales/ 2284245/)

Y PAM son digitales, pero casi nunca son binarias ya que un pulso no representa a un solo digito binario (bit).

2.3.2. Señal RC PWM

Modulación de a la duración o anchura de impulsos (PDM O PWM)

En el caso de la modulación PWM (ver figura 2.17) la duración de los impulsos varia proporcionalmente a los valores de muestra de la señal mensaje m(t).

El valor más positivo de m(t) corresponde al impulso ms ancho y el valor más negativo se representa como un impulso negativo.

Una señal PWM tiene la formula siguiente

$$X_{PWM}(T) = A \sum_{n=-\infty}^{\infty} \prod \left[\frac{t - nT_b}{\tau(nT_b)} \right] \tag{2.1}$$

Donde

$$\tau(t) = \tau_o + \tau_1[1 + m_t + m(t)] = \tau_o[1 + \Delta(t)] \tag{2.2}$$

Siendo

$$m_t = \frac{\tau_1}{\tau_0} \leq 1 \tag{2.3}$$

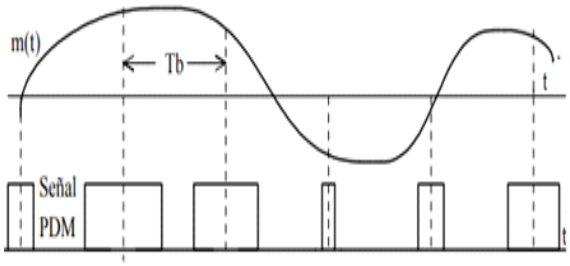

Figura 2.17: Modulación de la duración de Impulsos PDM. (Fuente: https://www.docsity.com/ es/pam-y-pcm-para-modulaciones- en-los-sistemas-digitales/2284245/)

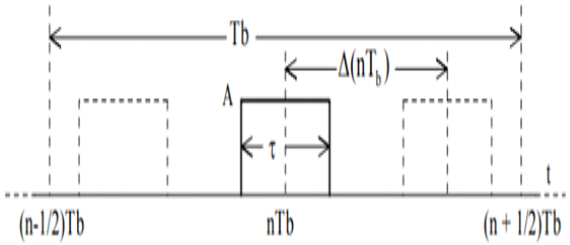

Figura 2.18: Relación de duración en PDM. (Fuente: https://www.docsity.com/ es/pam-y-pcm-para-modulaciones- en-los-sistemas-digitales/2284245/)

el índice de modulación PWM y

$$\Delta(t) = m_t m(t) \tag{2.4}$$

es la señal de los impulsos no modulados.

$$\tau_o$$

Los valores τ_0 y τ_1 se eligen de acuerdo a las siguientes condiciones, que la variación de la duración del impulso se efectué simétricamente alrededor del punto de muestra nTb , pero que también se pueda mantener fijo un borde del impulso mientras el otro orden es el que se desplaza como se muestra en la figura 2.18 :

2.3.3. Señal RC PPM

Modulación por posición de impulso (PPM)

La modulación del impulso de acuerdo a un punto dado varía según los valores de muestra de la señal de mensaje. En este sistema la información está contenida en los

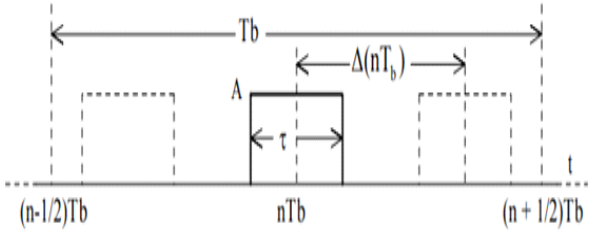

Figura 2.19: Modulación PPM.(Fuente: https://sites.google.com/site/cursodetelecomavanzada/ modulacin-de-pulsos- en-posicin-y-duracin)

desplazamientos de los impulsos de un tren de impulso, la portadora.

En la modulación por posición de impulso la amplitud y la duración de los impulsos se mantienen constantes, la información posicional es trasmitida por la posición del borde frontal del impulso.

Una señal PPM se puede expresar en la forma

$$X_{PPM(t)=} \sum_{n=-\infty}^{\infty} A \prod \left[\frac{t - nT_b - \Delta(nT)_b}{\tau} \right] \tag{2.5}$$

Donde $\Delta(t) = m_t m(t)$ es el desplazamiento instantáneo del impulso respecto al instante de referencia $t = nT_b$, como se observa en la figura 2.19.

La posición del impulso respecto a $t = nTb$ es proporcional a m(t); el desplazamiento máximo será:

$$|\Delta(t)|_{\text{máx}=} m_t \, |m(t)|_{\text{máx}} \leq \frac{1}{2}(T_b - \tau) \tag{2.6}$$

Dónde $|m(t)|_{\text{máx}}$ es el valor de $m(t)$y m_t el índice de modulación PPM[19].

Las modulaciones PPM Y PWM están íntimamente relacionadas, a tal punto que la modulación PPM se puede obtener directamente a partir de la modulación PDM como se muestra en la figura 2.20:

Ahora bien como la información reside en la posición temporal de los bordes del impulso y no en el impulso mismo, y como potencia es proporcional a la duración de los impulsos, es conveniente trasmitir impulsos muy angostos medidos en PPM, por lo que la potencia requerida por PPM es inferior a la requerida para PWM, lo cual nos indicaría una ventaja que se reflejara en las relaciones S/N.

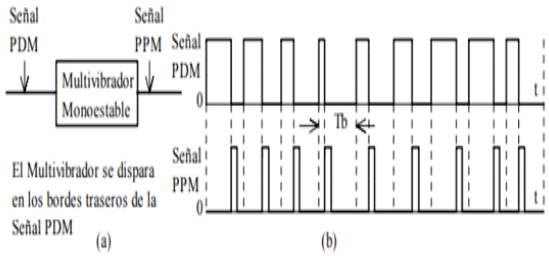

Figura 2.20: Generación de señales PPM a partir de señales PDM. (Fuente: https:/ /sites.google.com/ site/ cursodetelecomavanzada/ modulacin-de-pulsos-en -posicin-y-duracin)

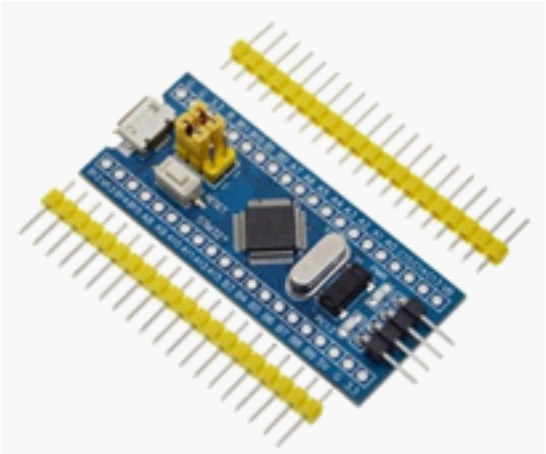

Figura 2.21: Microcontrolador STM32F103. (Fuente: Elaboración propia)

2.4. COMPONENTES FISICOS UAV

2.4.1. Microcontrolador STM32F103

Este microcontrolador ofrece una base para construir una amplia gama de sistemas integrados desde dispositivos básicos de seguridad alimentados por una batería hasta sistemas complejos como pilotos automáticos de helicóptero6. Este componente proporciona una diversidad de opciones de tamaño de memoria, rendimiento y potencia [29].

El microcontrolador STM32f103D (ver figura 2.21) fue elegido por tener un procesador ARM, 3 interfaces, por su velocidad 5 veces mayor a un Arduino nano, una micro USB para alimentación de la placa y comunicaciones y por contener 3 puertos UART que son de gran utilidad para este proyecto.

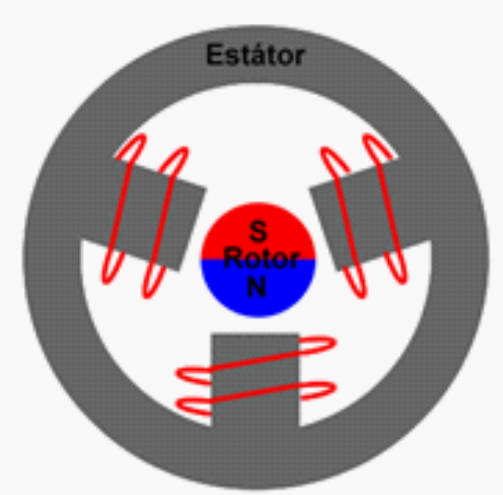

Figura 2.22: Estator de motor Brushless. (Fuente: http://motores.nichese.com/ brushless.htm)

2.4.2. Motor Brushless

Los motores brushless están compuestos por una parte móvil que es el rotor, que es donde se localizan los imanes permanentes, y una parte fija, denominada estator o carcasa, sobre la cual van dispuestos los bobinados de hilo conductor.

En la figura 2.22, se muestra como la corriente eléctrica pasa directamente por los bobinados del estator o carcasa, esta corriente produce un campo electromagnético que interacciona con el campo magnético originado por los imanes permanentes del rotor, haciendo que surja una fuerza que hace girar al rotor y por tanto al eje del motor. El rotor es controlado por el variador eléctrico, que funciona observando la posición en que se encuentra el rotor a cada instante para hacer que la corriente que llega sea la adecuada para provocar el movimiento de rotación que le corresponde. La velocidad del rotor y su eje va de la velocidad de secuencia del variador, por tanto el variador no se limita únicamente a cambiar la velocidad de secuencia de alimentación, sino que además aumenta o disminuye la tensión de alimentación de las bobinas para extraer el máximo rendimiento al motor brushless[33].

Para conocer la posición del rotor en cada momento se pueden utilizar procedimientos, y dependiendo de cuál utilice el motor será sensored o sensorless.

- Motor Brushless Sensored (ver figura 2.23). Estos motores cuentan con sensores que determinan la posición durante el giro del rotor y da la oportunidad de conocer el momento más idóneo para emplear el valor de tensión adecuado en la bobina

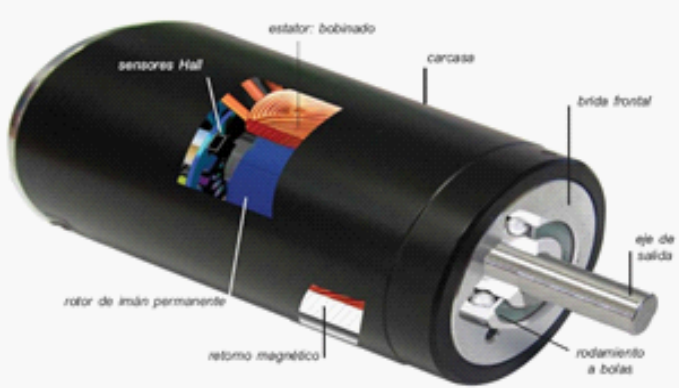

Figura 2.23: Componenetes del Motor Brushless Sensored. (Fuente: https://www. orientalmotor.com/ brushless-dc-motors- gear-motors/ technology/ brushless-dc-motors-servomotors -inverter.html)

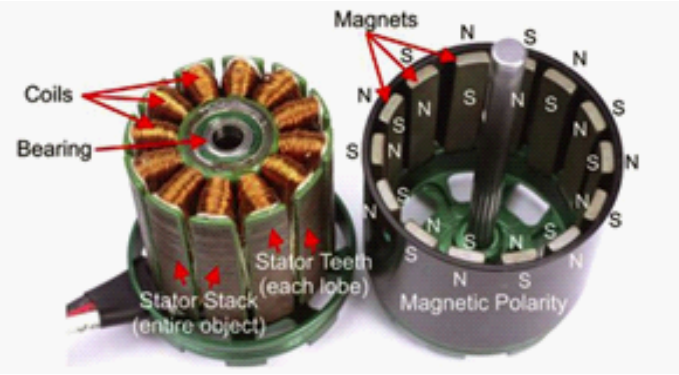

Figura 2.24: Componentes del Motor Brushless sensorless. (Fuente: https://www. dronetrest.com/ t/ brushless- motors-how-they- work-and-what-the- numbers-mean/564)

adecuada. Este motor debe ir asociado a un variador electrónico el cual se lleva mediante el cableado para enviar los niveles de tensión y otro conector que permitirá recibir información y a partir de ello se ajusta sus señales de salida a cada bobinado.

- Motor Brushless Sensorless (ver figura 2.24). En estos motores no existen sensores, para conocer la posición del rotor de estos motores se realiza monitorización de los impulsos o señales que envían a motor [30].

Los motores brushless se emplean en sectores industriales como: automóvil, aeroespacial, consumo, médico, equipos de automatización e instrumentación. En la industria los podemos encontrar en los robots de montaje, robot de pintura, robots de soldadura, etc [33], [30].

	Motor BLDC	Motor con escobillas
Conmutación	Conmutación electrónica basada en sensores de posición de efecto Hall	Conmutación por escobillas
Mantenimiento	Mínimo	Periódico
Durabilidad	Mayor	Menos
Curva velocidad/par	Plan. Operación a todas las velocidades con la carga definida	Moderada. A altas velocidades la fricción de las escobillas se incremnetra, reduciendo el par
Eficiencia	Alta. Si caída de tensión por las escobillas	Moderadas
Inercia del rotor	Bajada. Debido a los imanes permanentes en el rotor	Alta. Limta las caracteristicas dinámicas
Rango de velocidad	Alto. Sin limitaciones mecánicas impuestas por escobillas	Bajo. Límite lo impone princiálmente las escobillas
Ruido eléctrico generado	Bajo	Arcos en las escobillas
Coste de construcción	Alto. Debido a los imanes permanentes	Bajo
Control	Complejo y caro	Simple y barato
Requisitos de control	Un controlador es requerido siempre para mantener el motor funcionando. El mismo puede usarse para variar la velocidad	No se requiere control si no se necista una variación de velocidad

Cuadro 2.4: Comparación motores brushless vs brushless con escobilla

2.4.3. Controlador Electronico de Velocidad (ESC)

Es un dispositivo electrónico que sirve para controlar la velocidad del motor brushless, mediante la generación de pulsos compatibles con este tipo de motores. Independiente del tipo de motor eléctrico al que conectemos el variador, el ESC interpreta información de control. Anteriormente los variadores eran mecánicos y actuaban a través de servos.

La mayoría de los ESC incorporan un sistema BEC, el cual, hace posible regular un voltaje estable para poder hacer funcionar el receptor y los servos, lo que elimina la necesidad de llevar una batería extra dentro del drone para alimentar los componentes[19].

En la figura 2.25 se muestra la estructura de un ESC detallada de la siguiente manera:

- A la izquierda:
  - Cable rojo y negro: Alimentación del motor va conectado a la batería lypo.
  - Conector de 3 pins

Figura 2.25: Estructura de ESC. (Fuente: Elaboración propia)

- Amarillo: Señal PWM 50Hz desde el Arduino
- Rojo y negro: Alimentación del ESC

- A la derecha:
  - 3 cables de trifásica para la alimentación del motor brushless

El ESC funciona al recibir la señal PWM de 50 Hz y dependiendo de la longitud de pulso entregará más o menos potencia al motor, esta longitud varia de 1 ms a 1.5ms, parado y a máxima potencia respectivamente.

2.4.4. Gimbal

Es una plataforma monitorizada, se controla mediante una placa con varios sensores. Generalmente son acelerómetros y compás magnético que se encargan de mantener u objeto estabilizado en todo momento, la mayoría de las veces se trata de cámaras.

Características

- Aislamiento de vibraciones de montaje
- Motor integrado y del marco del diseño
- GoPro 3 montaje directo
- Protección de alimentación tensión de polaridad y la compensación de voltje
- Protección contra cortocircuito

Figura 2.26: Gimbal Tarot 2D. (Fuente: Elaboración propia)

- Ángulo de inclinación inicial de la aduana
- Soporte de ajuste de parámetros de sensibilidad y el software
- Soporta el modo de tasa de joystick y el modo de posición
- Doble procesador de 32 bits de alta velocidad del núcleo ARM
- Fácil de usar software [27]

2.4.5. DJI Naza v2 + GPS

DJI Naza para multirotores es un sistema de autopiloto diseñado para entusiastas de los multirotores proporcionando un excelente auto nivelación y estabilización vertical.

DJI NAZA está diseñado para funcionar con DJI F550 o S800 a la perfección, pero también puede ser usado con multirotores de cualquier tipo con configuraciones de hasta 8 rotores.

NAZA GPS es la computadora o FC que integra giroscopios, barómetro, y GPS y controla a través de los algoritmos avanzados y patentados por DJI para dar una autoestabilización inigualable a cualquier multirotores que lo utilice, NAZA puede medir altitud y posición por lo que puede ser usado para vuelo en automático [16].

El DJI NAZA GPS se caracteriza por:

- Tener un algoritmo avanzado de control que permite un control de vuelo sobresaliente

Figura 2.27: Componentes de DJ NAZA a) BEC; b) conexiones; c) cable USB; d) LED indicador; e) tarjeta principal; f) GPS NAZA; g) base GPS; h) extensor GPS. (Fuente: Elaboración propia)

a otros dispositivos del mercado, la excelente maniobrabilidad y precisión de NAZA permiten una inigualable experiencia de vuelo.

- El sistema permite 3 modos de control, GPS Atti Mode, Atti Mode y Manual Mode. El piloto puede cambiar al sistema durante el vuelo para ajustarse a las condiciones del lugar o dejarlo en automático para que el equipo elija el modo adecuado cuidado de seguridad
- El GPS permite mantener una posición fija, regresar a base RTH y controlado inteligente de orientación, con este sistema el multirotor podrá permanecer en el aire en una posición y altitud fijas en condiciones de tiempo. El rango de precisión es de menos de 2.5 m horizontales y 0.8m verticales.
- La función de orientación inteligente permite establecer el frente del multirotor de acuerdo a nuestra sección, a diferencia de los helicópteros y otros multirotores sin DJI este puede establecer el frente como cualquier brazo o a la dirección de casa, de esta última forma al avanzar siempre irá hacia el lugar de donde despejo.
- El sistema permanecerá estable si pierde señal del control remoto por cualquier razón
- Para prevenir choques por problemas de voltaje, existen dos sistemas de protección de bajo voltaje.

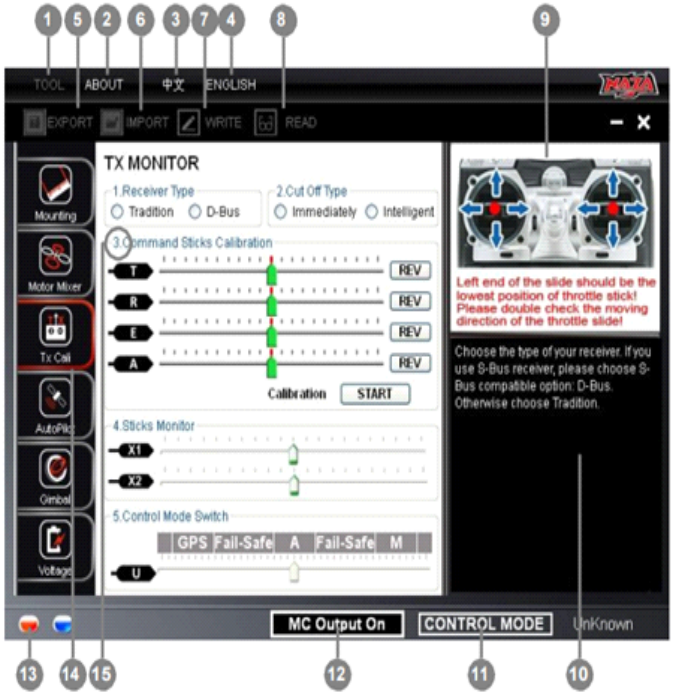

Figura 2.28: Partes del Software DJI NAZA. (Fuente: http://www.aeromodelismofacil.com/ Descargas/copteros/NAZA/Manual %20Naza_spain %20By_pfram

- El sistema permite mandar información a monturas de cama de dos servos para la corrección del roll y pitch de la cámara

  La instalación del Software y el Driver es mediante 4 etapas.

- Etapa 1. Descargar assistant software y driver de la website, luego descomprimirla
- Etapa 2: Conector MCy PC via USB cable y encender MAC
- Etapa 3. Si el sistema opertive intenta instalar el driver automáticamente , cancelarlo
- Etapa 4. Abrir la carpeta DJI_USB_Driver, seguir el manual Driver Installation hasta acabar[23].

Las partes del software son las siguientes (ver figura 2.28):

1. TOOL

   a) Firmware upgrade: actualiza firmware desde el serve, manteniendo el sistema autopilot al día

b) Disable all Knob

c) Check for Updates: Revisa la última versión del assitant software y firmware.

2. ABOUT

a) Info: Información de su producto

b) Error code

3. Interfaz chino

4. Interfaz ingles

5. Exporta datos de configuración

6. IMPORT: Importa datos de configuración compatibles con la versión

7. WRITE: Escribe los datos de la página actual al MC. El parámetro o título que se modifique se volverá rojo y remarcado, se sebe pulsar en el botón write o enter para actualizar el sistema.

8. READ: lee los parámetros desde el MC a la página actual

9. Guía gráfica

10. Guía texto

11. Control Mode: Indicación del modo de control

12. MC Output On: Indica salida hacia los ESCs; cuando se establece la comunicación entre MC y el assitant software vía cable USB cable, MC Output Off: indica que no hay salida hacia los motores, de modo que se puede configurar el multirotor con el assitant software con mayor seguridad

13. Simbolismo de colores:

a) Luz roja: MC↔PC ha sido desconectado

b) Luz verde: MC⟷PC ha sido conectado

c) Luz azul: MC⟷PC comunicándose

14. En estas posiciones se encuentran todos los contenidos de configuración en cada uno de los capítulos

15. Etapa de la configuración [23]

Figura 2.29: Pantalla OLED LCD 0.96. (Fuente: Elaboración propia)

OLED LCD 0.96”

La pantalla OLED LCD (ver figura 2.29) es un nuevo OLED de pequeño tamaño, se caracteriza por ser una pantalla que mide 128 x 64 y cuenta con dos tipos de interfaz IIC y SPI las cuales se encuentran en dos tarjetas diferentes.

Especificaciones:

- Voltaje de operación : 3.3-5 VCD
- Comunicación I2C
- Driver IC: SSD1306
- Angulo de vision: 160°
- Tamaño pantalla: 0.96”
- Resolución : 128*4
- Color: azul
- Temperatura de operación: ˜30˜80°C

Conexión:

- VCC: 3.3 -5 VCD

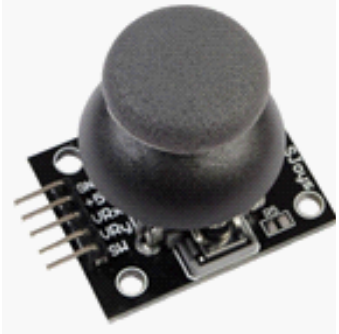

Figura 2.30: Vista general de un joysticks. (Fuente: Elaboración propia)

- GND: Tierra
- SCL: Serial Clock
- SDA: Serial Data[43]

Joysticks

Un joystick, o palancas de mando es un elemento de entrada para programas digitales. Los joysticks pueden ser digitales (solo indican estados de encendido y apagado) o análogos (indican posición mediante potenciómetros).

Los joysticks análogos está formado por dos potenciómetros que permiten medir el movimiento del stick en 2D, por lo que este movimiento es capturad por los potenciómetros, de tal manera que para cada movimiento en cada dirección será regulado un potenciómetro.

Con este módulo joystick se pueden utilizar el cambio de valor resistivo para tomar lectura con dos entradas analógicas en el Arduino. Este módulo de joystick cuenta con 5 pines los cuales se enumeran tomando de referencia de izquierda a derecha (ver figura 2.30):

1. GND: Pin conectado a tierra
2. +5V: pin de alimentación (5v)
3. VRx: pin de lectura de potenciómetro para el eje de las x´s
4. VRy: pin de lectura de potenciómetro para el eje de las y´s
5. SW: es un pin adicional que se utiliza para un push button en la parte inferior[24].

Los potenciómetros son de resistencias variables y actúan como sensores, proporcionando un volteje variable, esto dependiendo de la rotación del dispositivo alrededor de

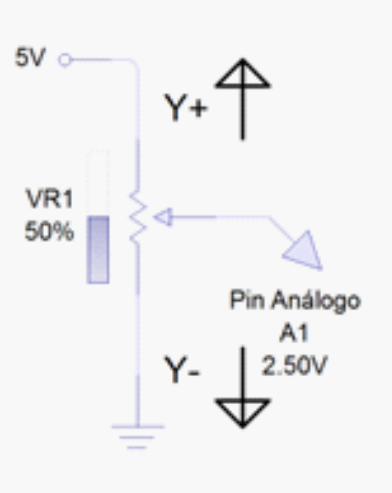

Figura 2.31: Potenciómetro vinculado al eje Y. (Fuente: Elaboración propia)

su eje, aunado a ello la configuración que se le otorgue al dispositivo; por ejemplo, un potenciómetro vinculado al eje Y, ejemplificado en la siguiente figura (ver figura 2.31):

Donde:

- Y+: Pin superior del potenciómetro
- Y-: Pin inferior del potenciómetro
- Pin análogo (A1): Punto de medición conectado a Pin análogo 1

Los joysticks usan, generalmente, potenciómetros tipo B de $10k\Omega$, en los cuales el valor del voltaje es directamente proporcional al ángulo de giro con respecto al pin conectado a tierra (GND), es decir entre mayor sea el ángulo con respecto a el pin conectado a tierra, mayor el voltaje en el punto de medición, ver figura 2.32:

Donde:

- Vin: Voltaje de entrada (5v)
- Vout: Voltaje de salida en el punto de medición
- R1: Resistencia de positivo al punto de medición.
- R2: Resistencia del punto de medición a GND

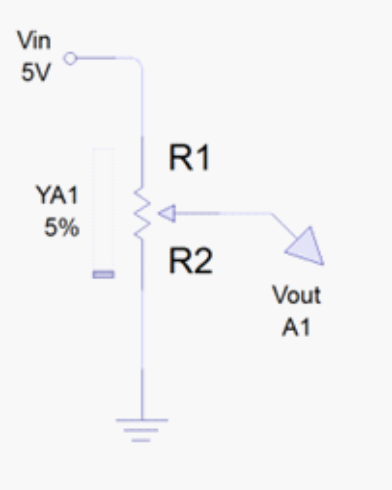

Figura 2.32: Potenciómetro tipo B. (Fuente: Elaboración propia)

En este ejemplo el valor de la resistencia R2 es equivalente al 5 % de la resistencia total, por lo que, R1 equivale al 95 % de la resistencia total, por lo que, si tenemos $10k\Omega$, el valor de R2 es equivalente a 500Ω y R1 a $9{,}5k\Omega$. Conociendo estos datos y utilizando un divisor de voltaje podremos obtener el voltaje en el punto de medición:

$$Vout = \frac{R2}{R1 + R2}(Vin) \tag{2.7}$$

Por lo que el voltaje en el punto de medición con las condiciones mencionadas equivale a 0.25 V que es igual al 5 % de 5V (Vin).

Se debe tomar en cuenta que los valores se obtienen del joystick en reposo, debido a que no todos ellos se encuentran en un estado de reposo en un punto medio , por lo que la configuración también tendrá modificaciones, por lo que se deben corregir los valores en map usando la siguiente ecuación

$$map(x, 0, 2x, 1, -1) : \tag{2.8}$$

Donde x es el valor de lectura en reposo, dado el caso que el valor en reposo es menor a 512.

En caso de que el valor en reposo sea mayor o igual a 512 se debe utilizar la siguiente formula[32]

$$map(x, y, inMax, 1, -1) \tag{2.9}$$

Donde:

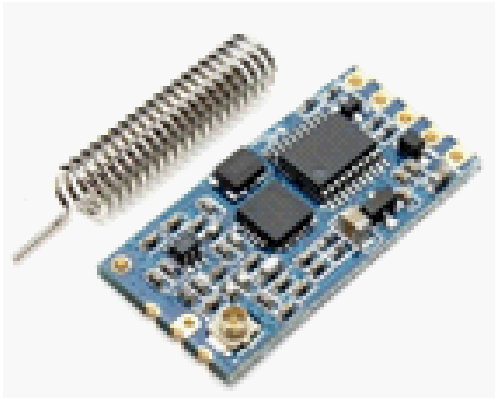

Figura 2.33: Módulo de comunicación HC-12. (Fuente: Elaboración propia)

$$y = \frac{x(utMax - outMin) + outMin(inMax)}{outMax} \tag{2.10}$$

2.4.6. Módulo de comunicación de puerto serie inalámbrico HC-12

El módulo de comunicación de puerto serial inalámbrico HC-12 es un multicanal de nueva generación de módulos de transmisión de datos inalámbricos incorporados.

El módulo HC 12 (ver figura 2.33) e puede aplicar en sensor inalámbrico, seguridad de la construcción comunitaria, control inalámbrico del robot, control remoto industrial y telemetría, adquisición automática de datos, gestión automática de datos, gestión de la información del contenedor, sistema POS, adquisición inalámbrica de datos del mediador de gas, sistema de entrada sin llave del vehículo y red inalámbrica de PC .

Las características del HC 12 son:

- Transmisión inalámbrica de larga distancia (1000 m en espacio abierto/tasa aud 5000 bps en el aire)
- Rango de frecuencia de trabajo (433.4-473.0 MHz, hasta 100 canales de comunicación)
- Potencia de transmisión máxima de 100mW
- Cuatro modos de trabajo, adaptados a diferentes situaciones de aplicación
- La MCU incorporada realiza la comunicación con un dispositivo externo a través de una serie puerto, no requiere programación o configuración para uso básico
- El número de bytes transmitidos continuamente es ilimitado

Las especificaciones de dicho dispositivo son:

- Frecuencia de trabajo : 433.4-473.0 MHz
- Tensión de alimentación: 3.2V a 5.5vdc
- Distancia de comunicación: 1000 m en el espacio abierto
- Tasa de baud serie: 1.2 kbps a 115.2 kbps (9.6kbps por defecto)
- Sensibilidad de recepción:-117dbm a -100dbm
- Potencia de transmisión: -1dbm a 20dbm
- Protocolo de interfaz: UARTttl
- Temperatura de trabajo:-4DA + 85
- Dimesiones:27.8 mm x 14.4mm x 4mm[4]

El módulo HC-12 tiene cuatro modos de transmisión transparente en un puerto serie, expresados como FU1, FU2, FU3, FU4, los cuales en funcionamiento actúan ocultando los detalles de comunicación inalámbricas desde dispositivos conectados. Los módulos se operan generalmente en pares, trasmitiendo datos por medio de un enlace semiduplex. Para una transmisión inalámbrica exitosa, el modo de transmisión, la velocidad en baudios del puerto serie y el canal de comunicación inalámbrica de los dos pares de módulos deben configurarse de la misma manera.

El modo FU1 es un modo de ahorro de energía moderado, con una corriente de trabajo inactiva de aproximadamente 3.6mA. Este modo se puede configurarse en cualquiera de los puertos, pero la velocidad en baudios en el aire es uniforme de 250 000bps.

El modo FU2 es un modo moderado de ahorro de energía extremo, con una corriente inactiva de aproximadamente 80uA. FU2 admite una velocidades en baudios de 1200bps, 2400bps y 4800bps, con una velocidad de baudios en el aire de 250 000 bps.

El modo FU4 es útil para un alcance máximo de hasta 1.8km. se admiten 1200bps con la velocidad en baudios de aire reducida a 500bps par distancia de comunicación mejorada, de tal manera que solo puede usar pequeñas cantidades de datos y los intervalos de tiempo entre los paquetes de envío no deben ser demasiado cortos para evitar la pérdida de datos.

El cuadro 2.5 proporciona valores de referencia típicos para los distintos modos:

Modo	FU1	FUI2	FU3	FU4	OBSERVACIONES
Corriente de reposo	3.6mA	80uA	16A	16mA	Valor promedio
Transmisión, tiempo de retardo	15-25mS	5000mS	4-80mS	1000mS	Enviando un byte
Prueba de bucle invertido, Tiempo de retraso 2	31mS				Velocidad en baudios del puerto serie 9600, enviando un bit
Operando rango en lleno potencia	100metros	1000 metros	600m en 9600bps 1000 m en 2400bp	1800m en 1200bps	Línea de visión clara enree los módulos bajo condiciones ideales

Cuadro 2.5: Comparación de FU1, FU2, FU3 y FU4

2.5. CONTROLADORES

2.5.1. PID

El controlador PID (Proporcional, Integral y Derivativo), es un controlador realimentado cuyo propósito es hacer que el error en estado estacionario, entre la señal de referencia y la señal de salida de la planta, sea cero de manera asintótica en el tiempo, lo que se logra mediante el uso de la acción integral. El controlador tiene además la capacidad de anticipar el futuro a través de la acción derivativa que tiene un efecto predictivo sobre la salida del proceso (ver figura 2.34).

Los controladores PID se pueden describir mediante sus funciones de transferencia, relacionando el error E(s) con la salida U(s) del controlador:

Calcula la variable Proporcional de PID

$$C_p(s) = K_p \tag{2.11}$$

Calcula la variable proporcional e integral de PID

$$C_{PI}(s) = K_p(1 + \frac{1}{T_r S}) \tag{2.12}$$

Calcula la variable proporcional y derivativo de PID

$$C_{PD}(s) = K_p(1 + \frac{T_d S}{T_D S + 1}) \tag{2.13}$$

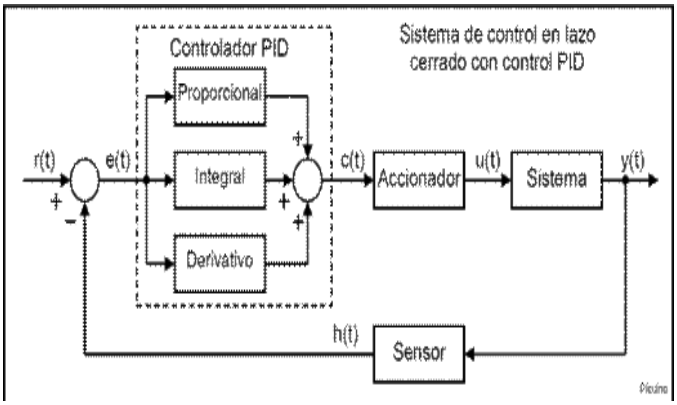

Figura 2.34: Sistema de controladores PID. (Fuente: https://www.info-transistor.info/ biblioteca/ Control %20Pid.pdf)

Calcula las tres variables de PID

$$C_{PID}(s) = K_p(1 + \frac{1}{T_r S} + \frac{T_d S}{T_D S + 1}) \tag{2.14}$$

Donde:

- Tr = constante denominada "tiempo de Reset"
- Td = constante denominada "tiempo Derivativo"

La función CPID (s) es conocida como la forma estándar, existen otras alternativas como:

$$C_{serie}(s) = K_s(1 + \frac{1_s}{s})(1 + \frac{D_s S}{y_s D_s S + 1}) \tag{2.15}$$

Calcula la función CPID en forma serie

$$C_{paralelo}(s) = K_p(1 + \frac{1_p}{s})(1 + \frac{D_p S}{y_p D_p S + 1}) \tag{2.16}$$

Calcula la función CPID en forma paralela
Los efectos de las acciones son:

- Proporcional: Contribuye con valores presente del error de control. Se usa la expresión "banda proporcional (PB)" para describir la acción proporcional. La equivalencia es $PB(\,\%) = (\frac{100(\%)}{K_p})$. El Cual se efine como el error requerido para alcanzar un cambio del 100 % en la salida de control.

- Integral: Contribuye con los valores proporcional al error acumulado o errores pasados (sumatoria).
- Derivativa: Contribuye con los valores proporcionales a la razón de cambio de los errores de control. Es un modo de reacción rápida a los cambios, el cual desaparece en presencia de errores de constantes. Su principal limitación es generar grandes señales en presencia de errores de control de frecuencia altas[10].

El control proporcional tiene la desventaja de que, en la mayoría de los casos, resulta un error estático o de estado estacionario diferente a cero. Los algoritmos de control usados en la práctica so más complejos que el del controlador proporcional. El algoritmo de PID se describe como[34]:

$$u(t) = K\,[e(t)+]\,\frac{1}{T_i}\int_0^I e(\tau)d\tau + T_d\frac{de(t)}{dt}) \tag{2.17}$$

Donde:

- E(t) es el error de la señal
- U(t) salida del controlador y entrada al proceso
- Kp es la ganancia proporcional
- Ti es la constante de tiempo integral
- Td es la constante de tiempo derivativo

En la figura 2.35, se muestra ejemplificada la ecuación general de los controladores PID.

De la ecuación **2.17** se describe de la siguiente manera, donde u es la variable de control y e es el error de control dado por $e = y_{sp} - y$. De esta manera, la variable de control es una suma de tres términos: el término P, que es proporcional al error; el término I, que es proporcional a la integral del error; y el término D, que es proporcional a la derivada del error. Los parámetros del controlador son: la ganancia proporcional K, el tiempo integral T_i y el tiempo derivativo T_d.

Acción proporcional

La ley de control de la ecuación se reduce a:

$$u(t) = Ke(t) + u_b \tag{2.18}$$

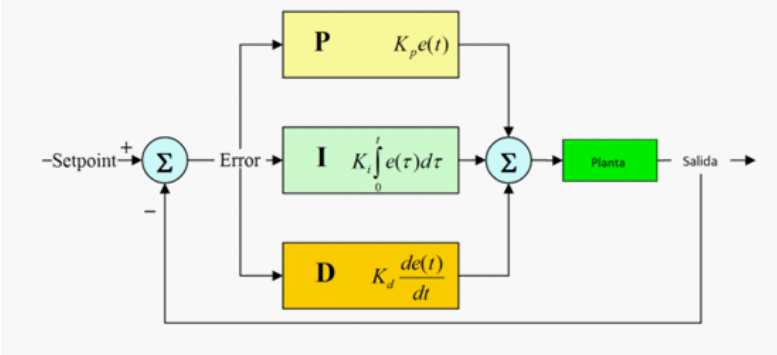

Figura 2.35: Algoritmo de PID. (Fuente: http://studylib.es/doc/289502/ limitaciones-de-un-control-pid)

Calcula la acción de control proporcional.

La acción de control es simplemente proporcional al error de control. La variable ub es una señal de polarización o un reset. Cuando el error del control e es cero, la variable de control toma el valor $u(t) = u_b$. La polarización ub a menudo se la fija en $\frac{(u_{max}+u_{min})}{2}$, pero algunas veces, puede ser ajustada manualmente de forma que el error de control en estado estacionario sea cero en una referencia dada[5].

Acción integral

La función de la acción integral es asegurar que la salida del proceso concuerde con la referencia en estado estacionario, Con la acción integral un pequeño error positivo siempre producirá un incremento en la señal de control y, un error negativo dará una señal decreciente8. La señal de control está dada por:

$$u_{0=K}(e_0 + \frac{e_0}{T_i}t) \tag{2.19}$$

Calcula la acción integral

Como se tiene que $e_0 \neq 0$, claramente se contradice el supuesto de que la señal de control u_0 se mantiene constante. Por tanto, como resultado de esto, un controlador con acción integral siempre dará un error en estado estacionario cero.

La acción integral también puede ser vista en un dispositivo que automáticamente restablece el término de polarización u_b de un controlador proporcional.

En la figura 2.36 se muestra un controlador proporcional con un "reset" que se ajusta automáticamente. El ajuste se hace realimentando una señal, que es un valor filtrado de la salida del controlador, a un punto de suma. A partir de diagrama de bloques, se pueden deducir las siguientes ecuaciones[31]:

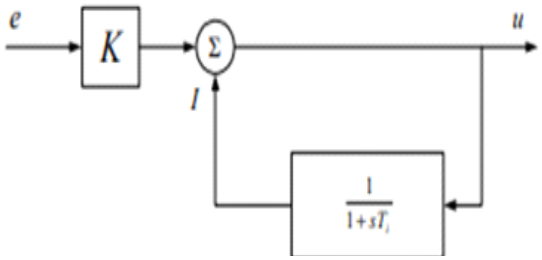

Figura 2.36: Acción integral de un controlador tipo PID. (Fuente: Elaboración propia)

$$u = Ke + l \tag{2.20}$$

$$T_i \frac{dL}{dt} + I = u \tag{2.21}$$

De donde, la eliminación de u entre otras ecuaciones produce:

$$T_i \frac{dL}{dt} + I = Ke + I \tag{2.22}$$

Y, de aquí:

$$T_i \frac{dL}{dt} = Ke \tag{2.23}$$

Con la que se muestra que el controlador de PI la figura 2.36. Es un controlador de tipo PI

Acción derivativa

El propósito de la acción derivativa es mejorar l estabilidad de lazo cerrado. El mecanismo de inestable se puede describir debido a la dinámica del proceso, es decir, pasa un tiempo antes de que la variable de control se note en la salida del proceso, por lo que de esta, manera el sistema de control tarda en corregir el error.

La predicción se hace por la extrapolación del error de control en la dirección de la tangente a su curva respectiva como se muestra en la figura 2.37

La estructura básica de un controlador PD está dada por:

$$u(t) = K(e(t) + T_d(\frac{de(t)}{dt})) \tag{2.24}$$

Calcula el controlador de variables PD

La expansión en series de Taylor de $e(t + T_d)$ da

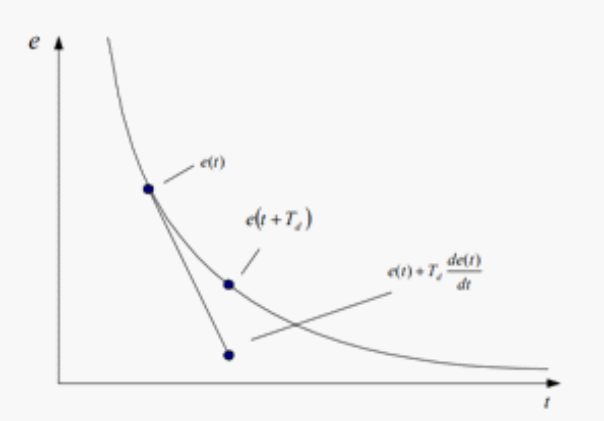

Figura 2.37: Interpretación geometrica de la acción derivativa como un control predictivo. (Fuente: Elaboración propia)

$$e(t + T\_d) \approx e(t) + T_d(\frac{de(t)}{dt}) \tag{2.25}$$

Calcula el estimado de error

De esta manera, la señal de control es proporcional a un estimado del error de control en él un tiempo Td hacia adelante, donde el estimado se obtiene mediante la extrapolación lineal, como se mostró en la figura 2.37.

2.6. PRESENTACION Y VISUALIZACIÓN

2.6.1. Laboratory Virtual Instrument Engineering Workbench LabVIEW

El LabVIEW es un lenguaje de programación de alto nivel, de tipo gráfico, y enfocado al uso en instrumentación. LabVIEW es una herramienta de programación, originalmente este programa está orientado a aplicaciones de control de instrumentos de electrónica usadas en el desarrollo de sistemas de instrumentación, lo que se conoce como instrumentación virtual.

LabVIEW proporciona un entorno único e integrado donde el software y el hardware trabajan juntos para ayudarle a desarrollar programas de manera más eficiente[38].

La primera versión de LabVIEW fue lanzada en 1986, la cual ha continuado evolucionando adaptándose cada vez a las necesidades tecnológicas que se han ido desarrollando a través de los tiempos.

Los programas de LabVIEW son llamados instrumentos virtuales o Vis ya que su apariencia y operación imitan a los instrumentos físicos, como osciloscopios multímetros. LabVIEW contiene una extensa variedad de herramientas para adquirir, analizar, visualizar y

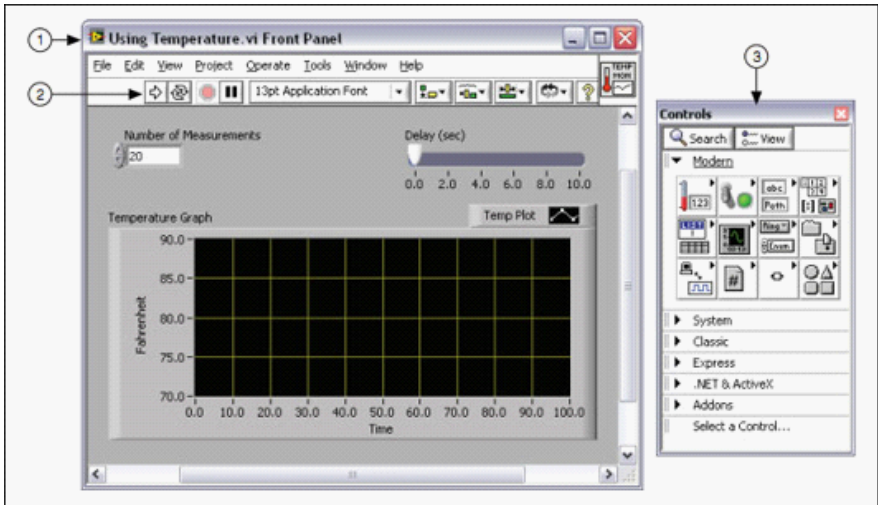

Figura 2.38: Ejemplo de Panel frontal. 1) Ventana de panel frontal; 2) Barra de herramientas y 3) Paleta de controles. (Fuente: Elaboración propia)

almacenar datos, así como herramientas para ayudarle a solucionar problemas en el código que escriba.

Los Vis tiene una parte interactiva con el usuario y otra arte de código fuente y aceptan parámetros procedentes de otros Vis.

Al crear un nuevo instrumento virtual se observan dos ventanas (ver figura 2.38):

- Panel frontal: Es la interfaz del usuario para el Instrumento virtual. Esta interfaz recoge las entradas procedentes dl usuario y representa las salidas proporcionadas por el programa.

Controladores e indicadores: Son terminales interactivas de entrada y salida del VI.

Los controladores pueden ser perillas, botones, barra deslizantes, estos simulan dispositivos de entrada de instrumentos y suministran datos al diagrama de bloques del VI.

Los indicadores son gráficas, y estos simulan dispositivos de salida de instrumentos y muestran datos que el diagrama de bloques adquiere o genera.

Los controladores e indicadores pueden ser :

- Numéricos (ver figura 2.39): Los datos numéricos se clasifican en 12 representaciones para los controladores e indicadores. Pueden representarlo números de varios tipos como entero o real.

    - Tipo entero (I, integer) : 8, 16, 31 bits

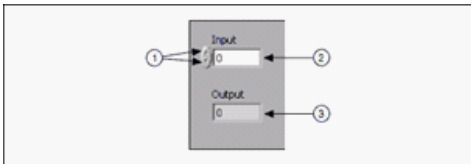

Figura 2.39: Controladores e indicadores numéricos: 1) botones de incremento/reducción; 2) Control numérico; 3) Indicador numérico. (Fuente: Elaboración propia)

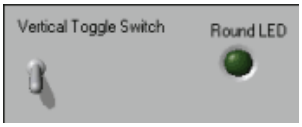

Figura 2.40: Controles e indicadores Booleano. (Fuente: Elaboración propia)

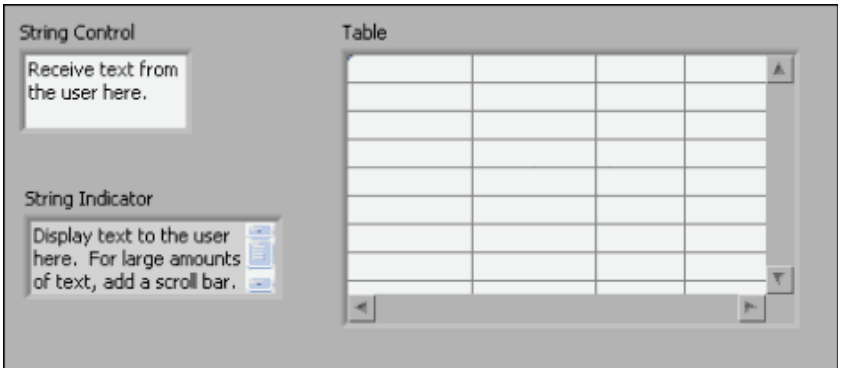

Figura 2.41: Controles e indicadores de cadena de caracteres. (Fuente: Elaboración propia)

- Tipo sin signo (U, Unsigned): 8, 16 y 32 bits
- Unto flotantes: 32 SGL, 64 DBL, Y 80 ET bits.
- Tipo complejos: Simples (CSG), dobles (CDB) y extendidos (CXT).
    - Booleano (ver figura 2.40): Definidos por enteros de 16 bits. True y false u on y off
    - Cadena de caracteres (ver figura 2.41): Es una secuencia de caracteres ASCII, esta se usa para recibir texto del usuario como una contraseña o nombre de usuario[37].

- Diagrama de bloques Constituye el código fuente del VI para controlar o realizar cualquier procesado de las entradas y salidas que se crean en el panel frontal15. Contienen terminales, subVis, funciones, constantes, estructuras y cables, lo cuales transfieren datos junto con otros objetos del diagrama de bloques (ver figura 2.42).

    - Ventana del diagrama de bloques: Contiene un código de fuente gráfica
    - Terminales de diagrama de bloques: Son puertos de entrada y salida que intercambian información entre el panel Fontal y diagrama de bloques.
    - Alambres: Es el camino de los datos entre los nodos, se colorean según el tipo de datos que llevan (ver figura 2.43).
        - Azules: Enteros
        - Naranjadas: Números en punto flotante
        - Verdes: Boleanos
        - Rosas: Cadenas de caracteres
    - Controladores, indicadores y constantes: Se comportan como entradas, y salidas del algoritmo del diagrama de bloques.
    - Nodos: Son objetos que en entrada y/o salida y realizan operaciones cuando el VI se ejecuta, estos pueden ser funciones, subVis o estructuras.
    - Funciones: Son los elementos de operación fundamentales de LabVIEW
    - SubVis: Son Vis que crean dentro de otro VI o que tiene acceso en la paleta de funciones. Es un lenguaje de programación basado en texto.
    - Estructuras: Estás controlan el flujo de datos en un VI. Los controladores tienen varias estructuras:
        - Bucle while_ Esta repite una sección de código hasta que se cumpla una condición determinada

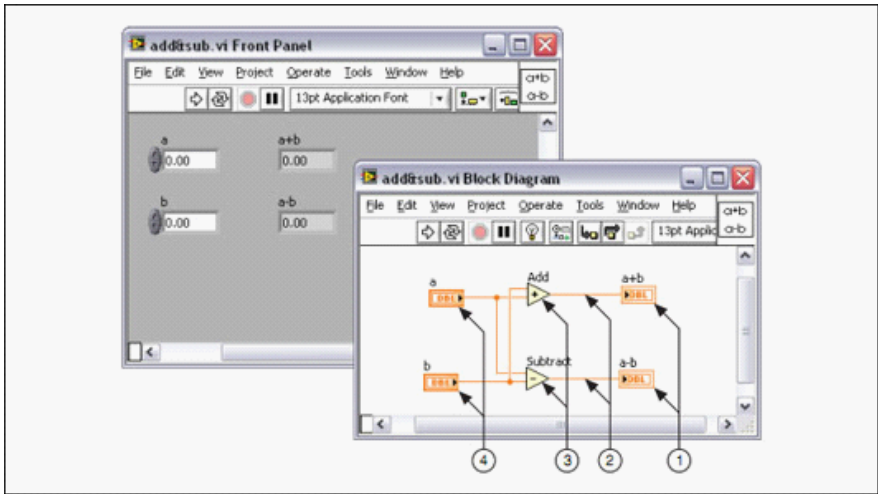

Figura 2.42: Diagrama de bloques y panel frontal: 1) Terminales de Indicador; 2) Cables; 3) Nodos, y $) terminales de control. (Fuente: Elaboración propia)

- Case: Son dos o más subdiagramas de manera que solo uno de ellos se ejecuta cuando la estructura se ejecuta.
- Nodo fórmula: Se utiliza para ejecutar fórmulas matemáticas directamente.
- Estructura secuencia: Ejecuta el diagrama de bloques de forma secuencial.
- Graficadores

- Paletas

  - Paleta de herramientas (Tools palette). Contiene las herramientas necesarias para editar y depurar los objetos del panel frontal y del diafragma
  - Paletas de controles (ver figura 2.44): Contiene controladores e indicadores, y está dividida en varias categorías.
  - Paleta de funciones: Se emplea en el diseño del diafragma de bloques, contiene todos los objetos empleados en la implementación de programa del VI, ya sean funciones aritméticas, de entrada/salida de señales, entrada salida de datos a fichero, adquisición de señales, temporizador de la ejecución del programa[37].

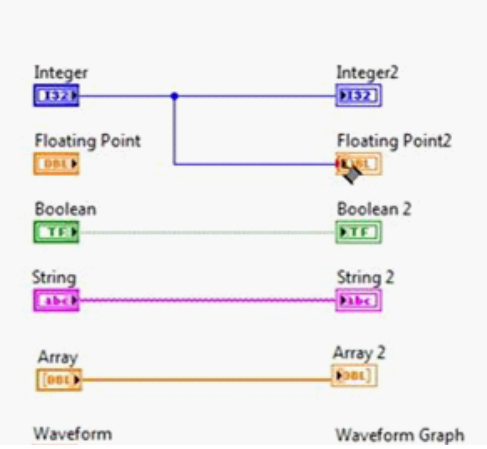

Figura 2.43: Ejemplificación de significado del alambre según el color. (Fuente: Elaboración propia)

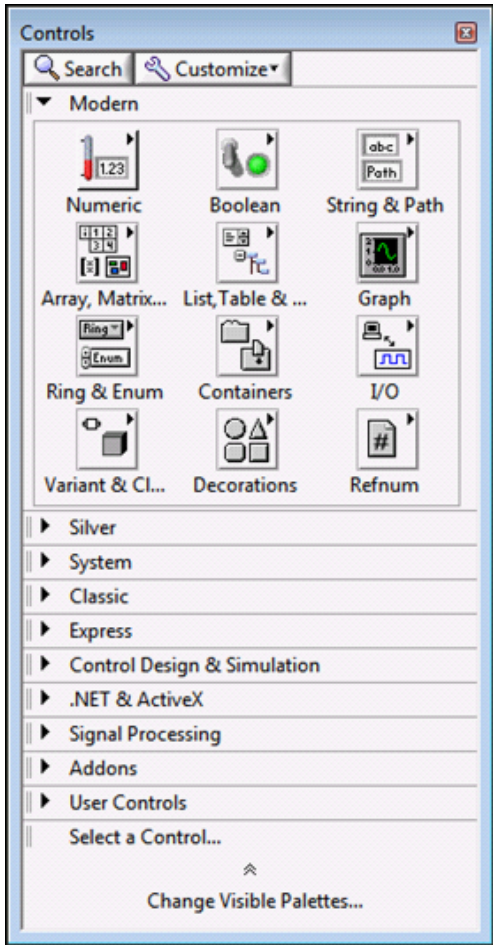

Figura 2.44: Paleta de controles. (Fuente: Elaboración propia)

Capítulo 3

DISEÑO DEL SISTEMA

3.1. MICROCONTROLADOR

3.1.1. Elección del microcontrolador

Para la realización del prototipo se realizaron investigaciones y conforme el proyecto se fue implementando todos los dispositivos para su correcto funcionamiento, se utilizaron diferentes microcontroladores como lo son Arduino Uno, Arduino Nano, Arduino Mega, Arduino Leonardo, Arduino Pro Micro y STM32F103. Con el avance del proyecto se eligió por el que fuera más adecuado, seleccionando así el microcontrolador STM32F103, por su capacidad de procesamiento, memoria, tamaño, costo; superando así a uno de los más prácticos para aplicaciones en UAV's como los es el Arduino Nano. Las características de los dos microcontroladores fueron la del cuadro 3.1:

	ARDUINO NANO	STM32F103
Microcontrolador	ATMega328	ARM 32-bot Cortex$^{TM} - M3$
Voltaje de operación	5V	3.3V
Voltaje de alimentación	7-12V	5V
I/O Digitales	14 I/O	26 I/O
Memoria Flash	32Kbytes	64 Kbytes
EEPROM	1KB	I
Frecuencia de trabajo	16MHz	72MHz (1.25 DMIPS/MHz)
I2C	1	2
UART	1	3
Interrupciones	2	Todos los pines I/O

Cuadro 3.1: Comparación de microcontrolador Arduino NANO y STM32F103

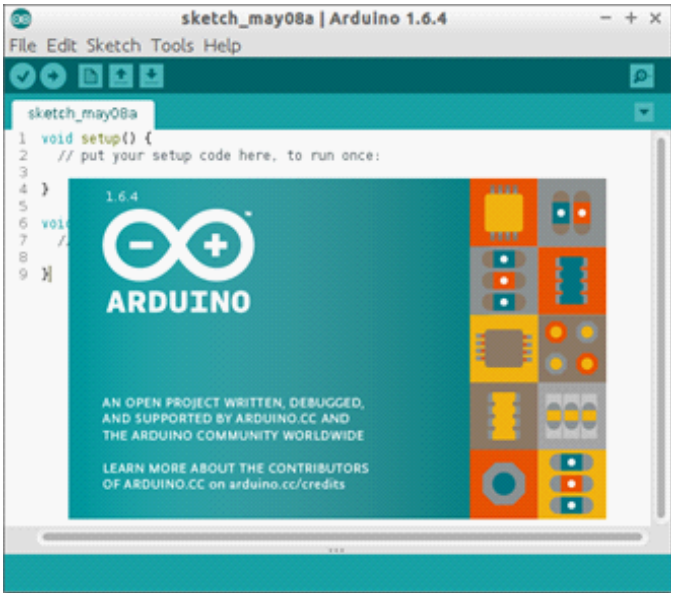

Figura 3.1: Hardware Arduino ADE. (Fuente: Elaboración propia)

3.2. PROGRAMACIÓN DEL PROCESADOR Y DE LAS INTERFACES GRAFICAS DE OPERACIÓN

3.2.1. Programación del microcontrolador

Los microcontroladores son programados con instrucciones propias del chip en lenguaje ensamblador. Para facilitar la programación, empresas como Microchip, NXP o Arduino crearon compiladores de código C. El código C tiene la facultad de ser más lógico e intuitivo que el ensamblador además de necesitar menos instrucciones para logar una aplicación.

Para la programación del chip STM32F103 en este proyecto se requirió del compilador desarrollado por la compañía open source y open hardware “Arduino” como se muestra en la figura 3.1 , antes conocida como Genuino.

La comunidad de desarrolladores Open Source de Arduino desarrollaron librerías para el microcontrolador STM32F103 y drivers para poder utilizar Arduino como plataforma de programación.

3.2.2. Instalación de Arduino y configuración

Para poder compilar y cargar un programa se tienen que realizar los siguientes pasos:

1. Descargar el ambiente de desarrollo Arduino

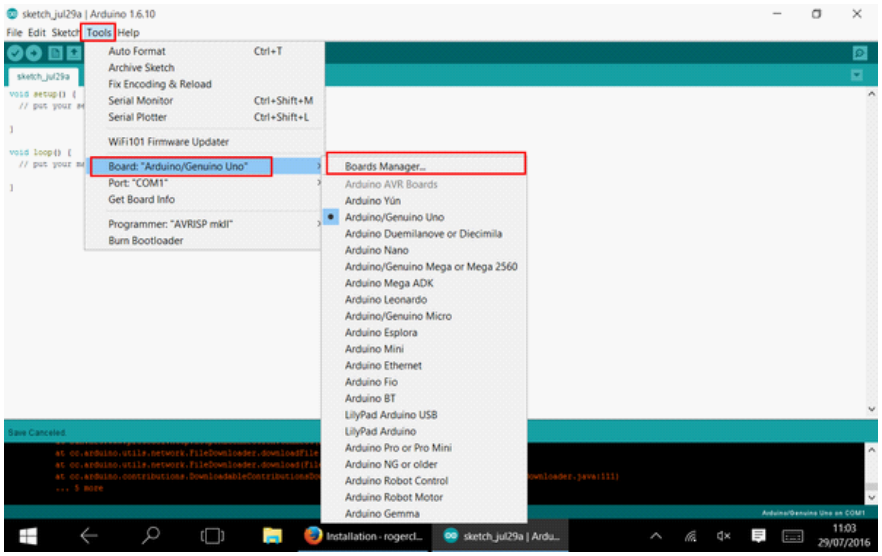

Figura 3.2: STM32 EN Arduino IDE 01. Ventana Boards Manager. (Fuente: Elaboración propia)

Para programar la tarjeta necesitas el ambiente de desarrollo Arduino, el cual lo puedes descargar del siguiente link (esta es la última versión disponible en este momento, pero siempre es bueno revisar la página oficial de Arduino http://www.arduino.cc para ver si hay actualizaciones)

2. Descargar plugin
   Para usar el microcontrolador STM32F103 en el IDE Arduino usaremos el plugin de Roger Clark:
   El repositorio git está en GitHub https://github.com/rogerclarkmelbourne/Arduino_STM32

3. En el IDE Arduino, vamos a la Ventana: "Tools-> "Board-> "Boards Manager..."(ver figura 3.2)

4. Instalamos el soporte a placas .Arduino/Genuino Zero", esto instala las herramientas .arm-none-eabi-g++"que también se usan en los STM32 (ver figura 3.3).

5. Vamos al menú .Archivo> "Preferencias"(imagen)

6. En el cuadro de texto "Gestor de URLs Adicionales de Tarjetas ponemos:
   http://dan.drown.org/stm32duino/package ... index.json
   Si ya se tiene algún escrito, se podrá darle al botón de la derecha para que se pueda abrir un área de texto y poner cada URL en una línea.

   STM32 en Arduino IDE 05 - Anadir URL de tarjetas B

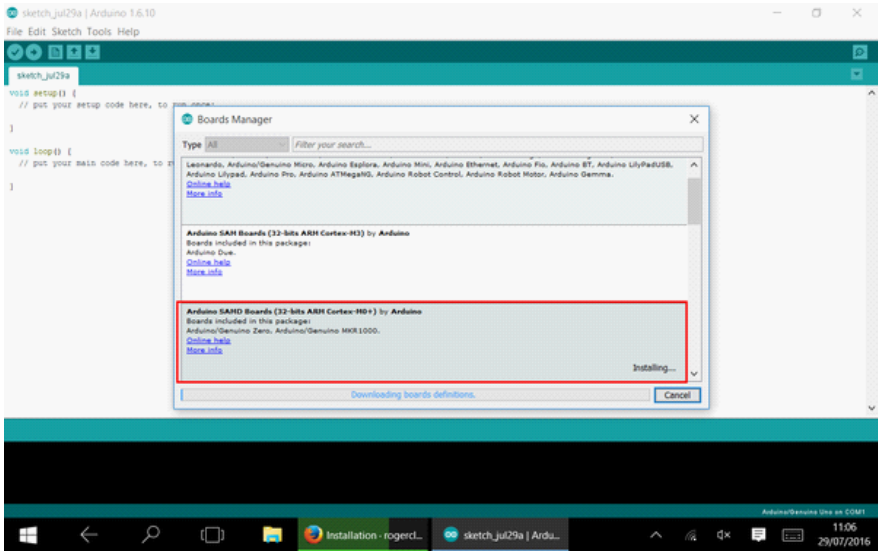

Figura 3.3: STM32 en Arduino IDE 02- Boards Manager Cortex M0. (Fuente: Elaboración propia)

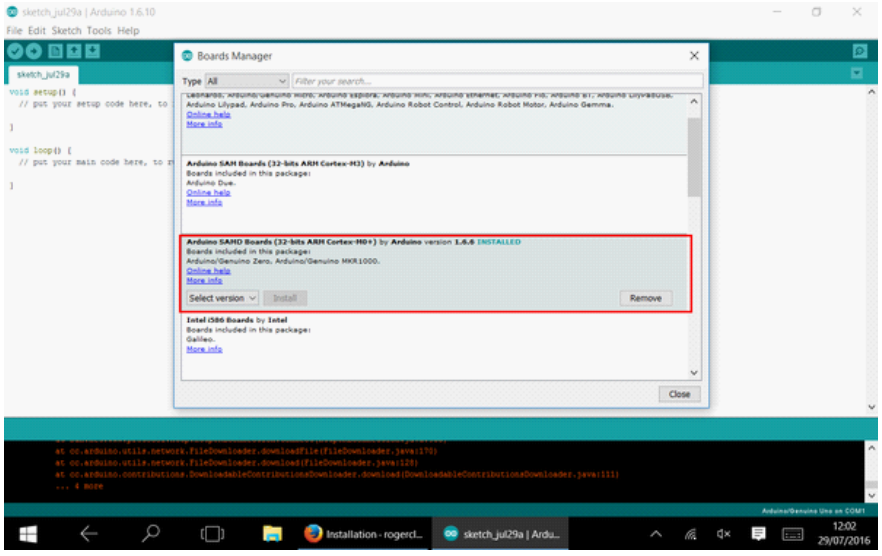

Figura 3.4: STM32 en Arduino IDE 03- Boards Manager Cortex M0 instalado. (Fuente: Elaboración propia)

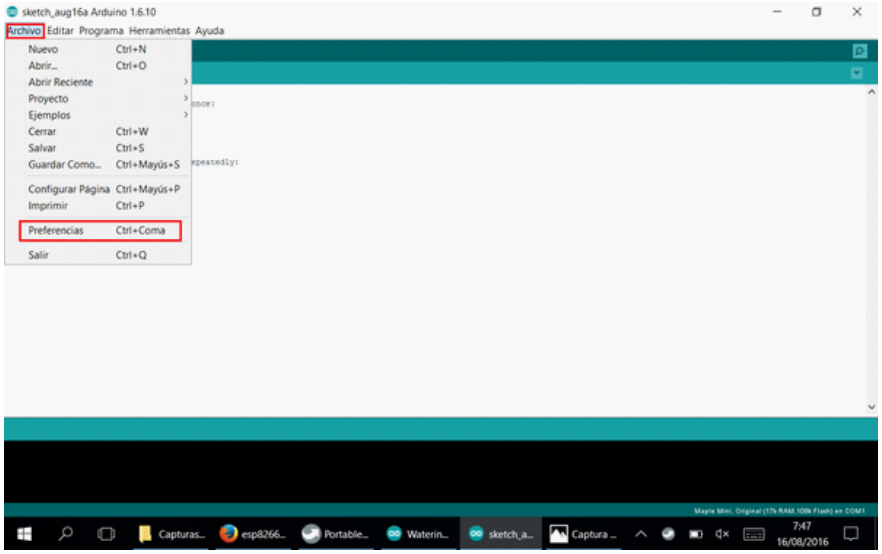

Figura 3.5: STM32 en Arduino IDE 04. (Fuente: Elaboración propia)

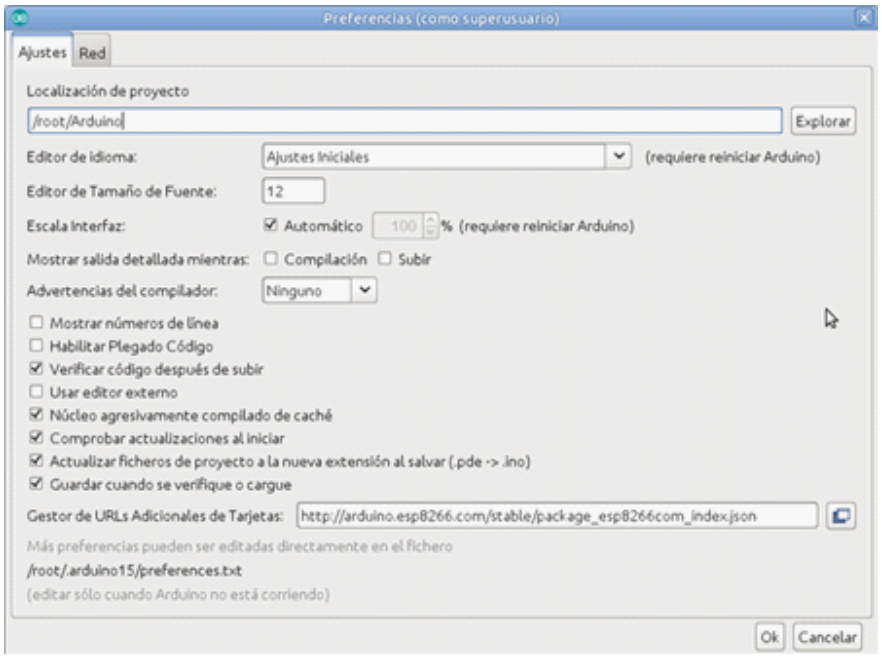

Figura 3.6: STM32 en Arduino IDE 05. (Fuente: Elaboración propia)

Figura 3.7: STM32 con URL. (Fuente: Elaboración propia)

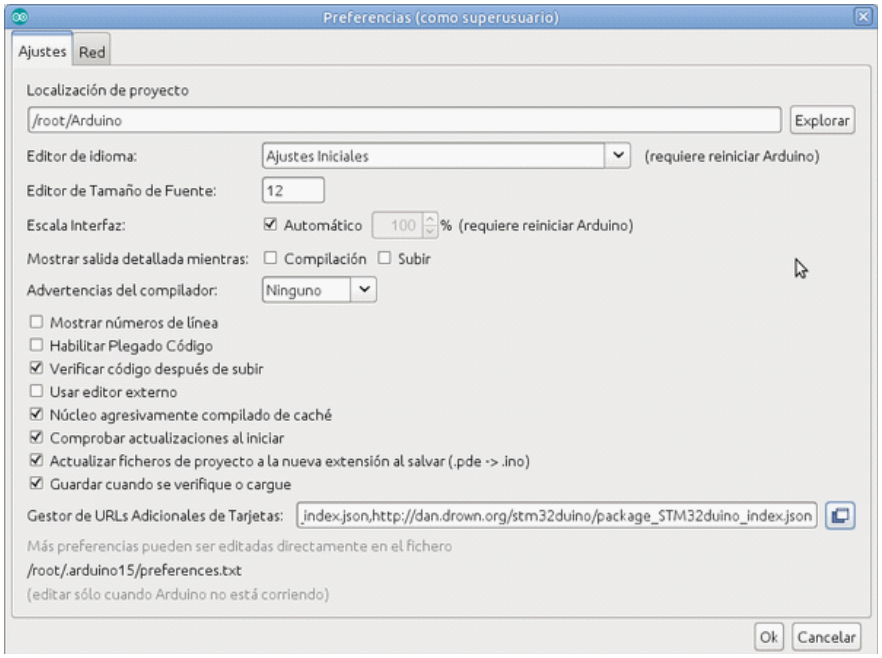

Figura 3.8: STM32 en Arduino IDE 05 - se añade URL de tarjetas C. (Fuente: Elaboración propia)

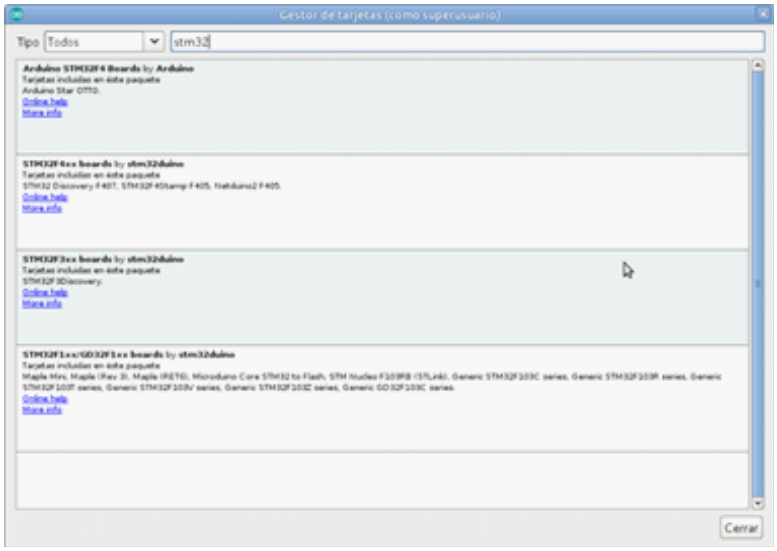

Figura 3.9: STM32 en Arduino IDE06- Ventana Boards Manager. (Fuente: Elaboración propia)

Pin	Tarjeta	ST-Link
1	3.3V	VDD
2	SWCLK	SWCLK
3	GND	Ground
4	SWDIO	SWDIO

Cuadro 3.2: Pines para dispositivos STM32F

7. Abrimos el menú "Herramientas> "Placa> "Gestor de tarjetas...". Una vez que se haya actualizado la información (se hace automáticamente) buscamos "stm32". Instalamos todas las opciones que salgan de "stm32duino".

   Si lo requiere solo se puede instalar lo que necesite el proyecto que se esté realizando

   Comprobamos que está instalando

8. Ya estarán disponibles las placas en el menú "Herramientas> "Placa"

   Para cargar un el código en la tarjeta necesitaremos un cable ST-link (ver figura 3.12) para conectarlo se utiliza el puerto USB de la computadora, es necesario descargar el driver compatible con el sistema operativo.

   Solo necesitamos tres pines para la programación básica y la depuración de los dispositivos STM32FXXX, (cuatro si está alimentando la placa del programador) usando Serial Debug Wire (SWD) (cuadro 3.2).

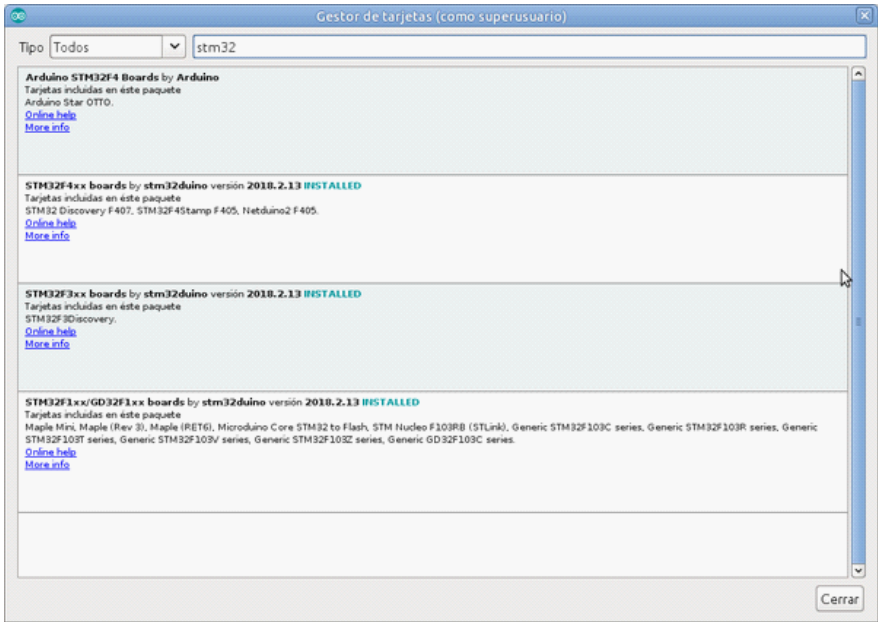

Figura 3.10: STM 32 en Arduino IDE 07- Ventana Boards Manager instalado. (Fuente: Elaboración propia)

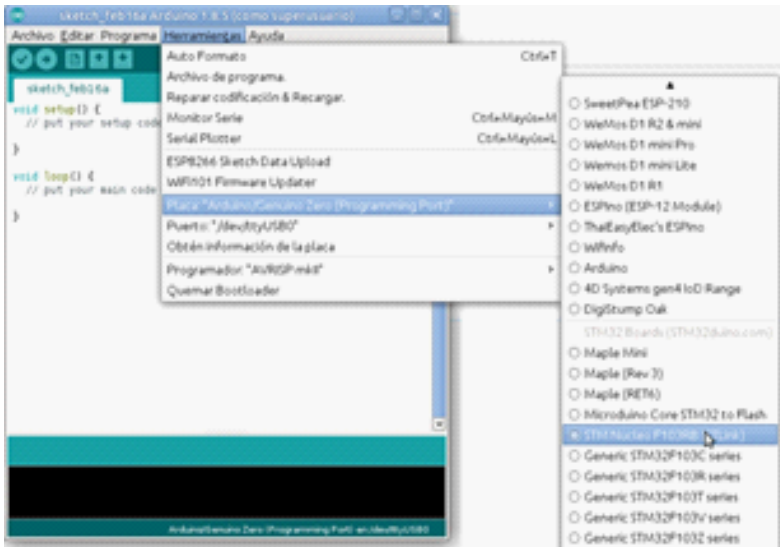

Figura 3.11: STM32 con placas disponibles. (Fuente: Elaboración propia)

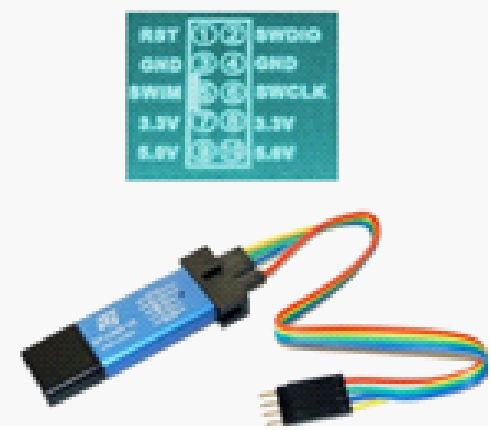

Figura 3.12: Cable ST-link. (Fuente: Elaboración propia)

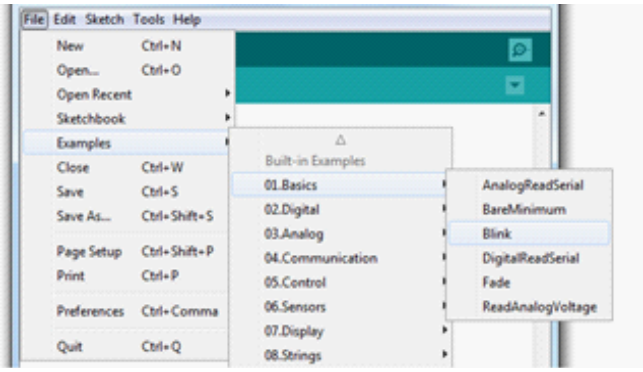

Figura 3.13: Blink. (Fuente: Elaboración propia)

Una vez realizadas las conexiones podemos efectuar una prueba para revisar que todo este correctamente.

9. Abra el boceto Blink desde .Archivo> Ejemplos> 01. Básicas> Parpadear (ver figura 3.13).

10. Haga clic en el botón subir (ver figura 3.14).

    Al finalizar los pasos anteriores el LED deberá parpadear en el STM32F103C como en la figura 3.15.

    El código es el siguiente:

```
1.  /*
2.  Blink
```

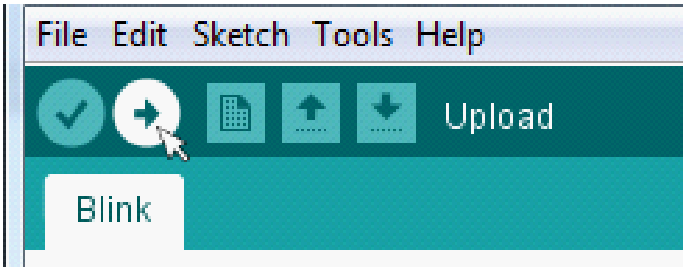

Figura 3.14: Cargar. (Fuente: Elaboración propia)

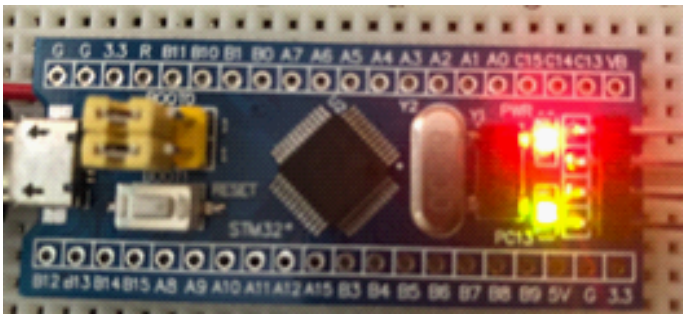

Figura 3.15: LED Pin PC13 funcional. (Fuente: Elaboración propia)

3. Turns on an LED on for one second, then off for one second, repeatedly.
4.
5. Most Arduinos have an on-board LED you can control. On the Uno and
6. Leonardo, it is attached to digital pin 13. If you're unsure what
7. pin the on-board LED is connected to on your Arduino model, check
8. the documentation at http://arduino.cc
9.
10. This example code is in the public domain.
11.
12. modified 8 May 2014
13. by Scott Fitzgerald
14.
15. Modified by Roger Clark. www.rogerclark.net for Maple mini 25th April 2015 , where the LED is on PB1
16.
17. */

```
18.
19.
20.   // the setup function runs once when you press reset or power the
board
21.   void setup() {
22.   // initialize digital pin PB1 as an output.
23.   pinMode(PB1, OUTPUT);
24.   }
25.
26.   // the loop function runs over and over again forever
27.   void loop() {
28.   digitalWrite(PB1, HIGH); // turn the LED on (HIGH is the voltage
level)
29.   delay(1000); // wait for a second
30.   digitalWrite(PB1, LOW); // turn the LED off by making the voltage
LOW
31.   delay(1000); // wait for a second
32.   }
```

3.3. GPS N8M

Este sensor es utilizado en la estación tierra como ya se describió en el capítulo 3. El GPS o Sistema de Posicionamiento Global (Global Positioning System de acuerdo a sus abreviaturas en inglés) se utiliza para determinar la posición de un objeto, un vehículo o persona con mucha precisión. Existen diferentes versiones comerciales con características no tan distantes, para este proyecto se necesitó una versión que fuera portátil y con un buen posicionamiento así que se investigaron diferentes tipos de GPS entre los que solo iban cambiando sus características de acuerdo a la generación.

Se estudió la precisión de los datos obtenidos por el GPS NEO-6M y el NEO-M8N. Y de igual manera se tomaron muestras para determinar cuál sería el más preciso para el proyecto.

En el cuadro 3.3 se muestra la desviación estándar de latitud, longitud, altitud y rumbo de la brújula. Como se puede observar las condiciones de recepción y la interferencia pueden tener un impacto en los resultados obtenido en 200 muestras.

El error de localización fue calculado utilizando la siguiente formula:

$$d = R\sqrt{a^2 + b^2} \tag{3.1}$$

Receptor GPS	STDEV de latitud (segundos)	STDEV de longitud (segundos)	STDEV de altitud (m)	Localización error (m)
NEO-6M (Buenas condiciones)	<0.18	<0.18	10.44	<6.783
NEO-6M (Condiciones dificiles)	0.688	1.547	8.29	39.597
NEO-6M (Buenas condiciones)	<0.18	<0.18	4.29	<6.783
NEO-6M (Condiciones dificiles)	0.218	0.784	19.78	18.22+

Cuadro 3.3: Comparación de DE (latitud, longitud, altitud y localización

Calcula el error de localización

Donde R es el radio de la tierra (6371 Km) y

$$a = \Delta\lambda * \cos\varphi_m \tag{3.2}$$

$$b = \Delta\varphi \tag{3.3}$$

Donde φ indica la latitud, λ la longitud y φ_m denota el valor inicial de la latitud

En la figura 3.16se muestra la gráfica de 200 muestras de la altitud obtenidas por los GPS

Como se observa dichos valores de altura no son muy buenos para utilizar el dato obtenido por el GPS para obtener la altura por lo que se optó por un sensor que mida la presión para obtener un dato más preciso.

Los datos experimentales muestran que las condiciones de recepción tienen impacto significativo en la precisión de los resultados.

Para la posición y la altitud, el receptor NEO-M8N (ver figura 3.17) proporciona una mayor precisión buenas condiciones por lo que se optó por utilizar este para el proyecto.

3.3.1. Librería GPS NEO-M8N

Para utilizar este GPS es necesario la librería TinyGPS ++, la cual, es una nueva biblioteca Arduino para analizar flujos de datos NMEA provistos por módulos GPS. Dicha biblioteca proporciona métodos compactos y fáciles de usar para extraer la posición, fecha, hora, altitud, velocidad y rumbo de los dispositivos GPS del consumidor.

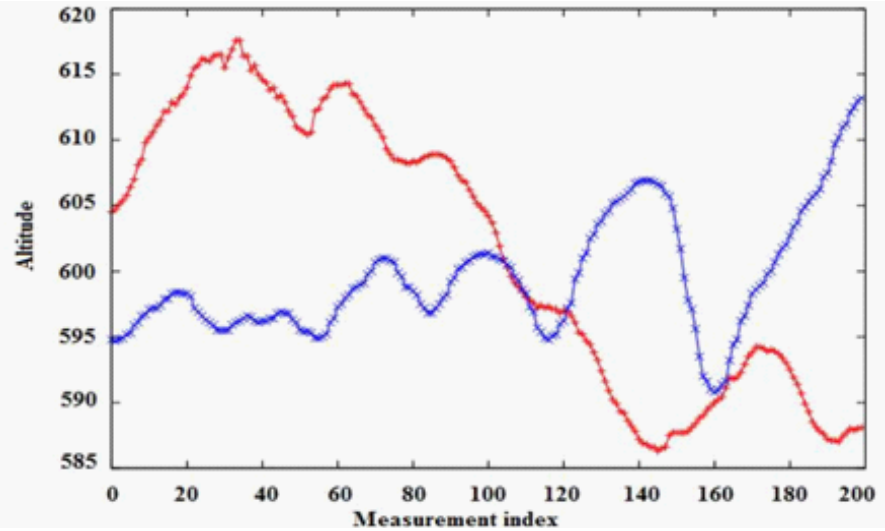

Figura 3.16: Grafica de Altitud: Rojo) GPS NEO 6M y Azul) GPS NEO M8N. (Fuente: Elaboración propia)

Figura 3.17: GPS NEO M8N. (Fuente: https://www.robotshop.com/ media/files/pdf2/ radiolink_se100_gps_user_manual 2016.7.13.pdf)

Para instalar esta biblioteca, se descarga en https://github.com/mikalhart/TinyGPSPlus, descomprimimos el archivo en la carpeta Arduino "librariesz reinicie Arduino. Debe cambiar el nombre de la carpeta "TinyGPSPlus".

3.3.2. Uso

En el IDE de Arduino simplemente crearía una instancia de TinyGPS ++ como esta:

1. #include "TinyGPS++.h"
2. TinyGPSPlus gps;

Recibir un nuevo dato:

1. while (ss.available() > 0)
2. gps.encode(ss.read());

Mostrar nuevos datos:

1. if (gps.altitude.isUpdated())
2. Serial.println(gps.altitude.meters());

En el anexo se encuentra el código final de pruebas.

3.4. DJI NAZA GPS/COMPASS

Para el GPS del drone se utiliza el GPS que utiliza la tarjeta NAZA, para ello se intercepta la información del GPS de forma paralela, esta es conectada al puerto Serial de nuestra tarjeta, ahí debemos codificar los caracteres que envía el GPS para obtener los datos necesarios para el prototipo (ver figura 3.18).

3.4.1. Librería

Para la obtención de los datos es utilizada la librería NazaDecoder, esta librería adquiere los datos de la latitud, longitud, angulo de rotación entre otros. La información sobre cada aspecto de la librería se encuentra en el siguiente link: https://github.com/dalmirdasilva/ArduinoNazaDecoder/blo

El GPS trabaja a con una comunicación en serie a 115200 baudios y los mensajes binarios se envían en el siguiente formato:

55 AA XX YY <carga útil> ZZ ZZ

Donde:

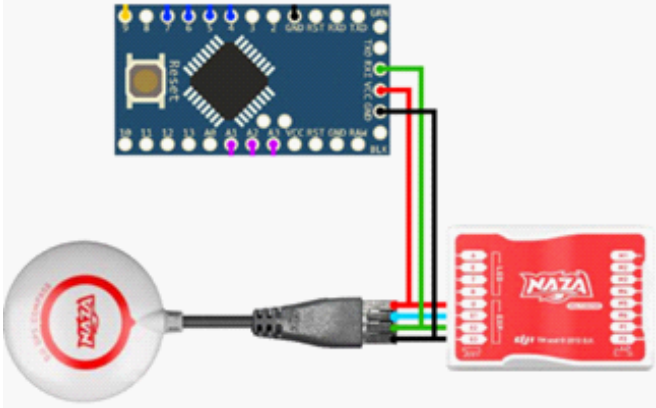

Figura 3.18: Conexiones GPS NAZA. (Fuente: Elaboración propia)

- XX es la longitud del mensaje
- YY es la ID
- ZZ ZZ parece ser suma de comprobación

ID 10 que contiene datos de GPS, enviados cada 250 ms
ID 20 que contiene datos de la brújula, enviados cada 30ms
ID 30 que contiene los números de versión del módulo GPS, enviados cada 2 segundos
El código utilizado para solo obtener los datos de latitud, longitud y rotación es el siguiente:

```
1.  /*
2.  DJI Naza (v1, v1 Lite, V2) data decoder library example
3.  (c) Pawelsky 20141109
4.  Not for commercial use
5.
6.  Refer to naza_decoder_wiring.jpg diagram for proper connection
7.  */
8.
9.  #include "NazaDecoderLib.h"
10.
11.
12.   void setup()
13.   {
14.   Serial.begin(115200);
```

```
15.  }
16.
17.  void loop()
18.  {
19.  if(Serial.available())
20.  {
21.  uint8_t decodedMessage = NazaDecoder.decode(Serial.read());
22.  switch (decodedMessage)
23.  {
24.  case NAZA_MESSAGE_GPS:
25.  Serial.print("Lat: "); Serial.print(NazaDecoder.getLat(), 7);
26.  Serial.print(", Lon: "); Serial.print(NazaDecoder.getLon(), 7);
27.  Serial.print(", Alt: "); Serial.print(NazaDecoder.getGpsAlt(), 7);
28.  Serial.print(", Fix: "); Serial.print(NazaDecoder.getFixType());
29.  Serial.print(", Sat: "); Serial.println(NazaDecoder.getNumSat());
30.  break;
31.  case NAZA_MESSAGE_COMPASS:
32.  Serial.print("Heading: "); Serial.println(NazaDecoder.getHeadingNc(),
2);
33.  break;
34.  }
35.  }
36.  }
```

3.5. BARÓMETRO

Para el proyecto se necesitó un sensor preciso con una resolución de alrededor de 10 cm. Hay disponibles algunas presiones barométricas que son pequeñas, baratas y tienen la precisión requerida.

Se realizó una comparación entre:

- LPS25H
- MS5611
- BMP180
- BMP280

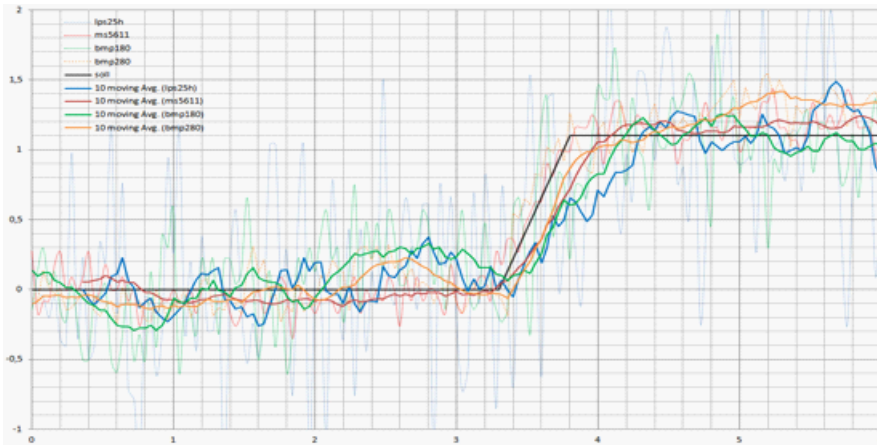

Figura 3.19: Lectura de altura. (Fuente: Elaboración propia)

Con el Arduino se leyeron todos los sensores a 25Hz. De las hojas de datos podemos obtener los siguientes datos:

- LPS25H: ADC de 24 bits, 128 avg, la mejor tasa de actualización: 25Hz
- MS5611: ADC de 24 bits, 4096 osr, la mejor tasa de actualización: 50Hz
- BMP180: ADC de 19 bits, 8 osr, mejor tasa de actualización: 33Hz
- BMP280: 20 bits ADC, 16 osr_p, 2osr_t, la mejor tasa de actualización: 26Hz

Si se usa la configuración anterior y mira los datos de salida, puede calcular el número efectivo de bits. Para hacer esto, se trazaron los valores de presión bruta como binarios y busco los bits que cambian constantemente en los datos. Solo se consideraron 4 muestras consecutivas para eliminar drifft o cambiar la presión atmosférica como un factor. Ahora las resoluciones se ven de la siguiente manera:

LPS25H: 14 bits

MS5611: 17 bits

BMP180: 14 bits

BMP280: ...

Ahora bien, para conocer cual sensor sería el más tangible se realizó un experimento y compararon todos los sensores entre sí. En donde la

línea negra indica la .ªltura verdadera". Todos los sensores se han filtrado con una media móvil de 10 muestras (ver figura 3.19).

Se puede ver que el MS5611 ofrece los mejores resultados en términos de precisión. En el cuadro 3.4 se muestra la comparación de la división estándar y los valores pico a pico de los sensores:

Sensores	Sigma	pkpk
LPS25H	60cm	2,85 m
BMP180	35 cm	1,52 m
BMP280	14 cm	0,65 m
MS5611	12 cm	0,62 m

Cuadro 3.4: Comparación de DE y valores pico de diferentes sensores

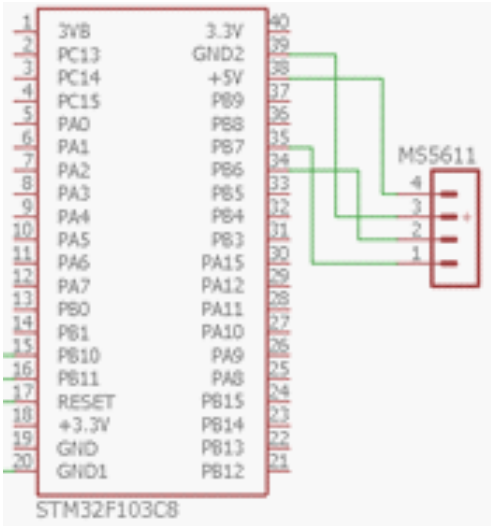

Figura 3.20: STM32F103C Conexiones barómetro. (Fuente: Elaboración propia)

1.

3.6. CONEXIONES

Hay varios módulos listos en el mercado con sensores MS5611, que difieren principalmente con el nivel de voltaje de suministro (ver figura 3.20). La mayoría de las veces estos son módulos accionados por 3.3V. En este proyecto se usará el módulo IMU GY-86, ya que puede suministrarse con voltaje de 5V y 3.3V. Si se elige suministrar 5V, es conveniente prestar especial atención a la conexión al pin apropiado, ya que la conexión al pin marcado 3.3V puede dañarse.

3.6.1. Librería MS5611

Para admitir módulos con sistemas MS5611 se puede descargar desde el repositorio de Git : https://github.com/jarzebski/Arduino-MS5611

Ejemplo de pruebas:

```
1.  /*
2.  MS5611 Barometric Pressure & Temperature Sensor. Simple Example
3.  Read more: http://www.jarzebski.pl/arduino/czujniki-i-sensory/czujnik-
cisnienia-i-temperatury-ms5611.html
4.  GIT: https://github.com/jarzebski/Arduino-MS5611
5.  Web: http://www.jarzebski.pl
6.  (c) 2014 by Korneliusz Jarzebski
7.  */
8.
9.  #include <Wire.h>
10.  #include <MS5611.h>
11.
12.  MS5611 ms5611;
13.
14.  double referencePressure;
15.
16.  void setup()
17.  {
18.  Serial.begin(9600);
19.
20.  // Initialize MS5611 sensor
21.  Serial.println("Initialize MS5611 Sensor");
22.
23.  while(!ms5611.begin())
24.  {
25.  Serial.println(Çould not find a valid MS5611 sensor, check wiring!");
26.  delay(500);
27.  }
28.
29.  // Get reference pressure for relative altitude
30.  referencePressure = ms5611.readPressure();
31.
32.  // Check settings
33.  checkSettings();
34.  }
35.
36.  void checkSettings()
```

```
37.     {
38.     Serial.print(.oversampling: ");
39.     Serial.println(ms5611.getOversampling());
40.     }
41.
42.     void loop()
43.     {
44.     // Read raw values
45.     uint32_t rawTemp = ms5611.readRawTemperature();
46.     uint32_t rawPressure = ms5611.readRawPressure();
47.
48.     // Read true temperature & Pressure
49.     double realTemperature = ms5611.readTemperature();
50.     long realPressure = ms5611.readPressure();
51.
52.     // Calculate altitude
53.     float absoluteAltitude = ms5611.getAltitude(realPressure);
54.     float relativeAltitude = ms5611.getAltitude(realPressure, reference-
Pressure);
55.
56.     Serial.println(-");
57.
58.     Serial.print(rawTemp = ");
59.     Serial.print(rawTemp);
60.     Serial.print(", realTemp = ");
61.     Serial.print(realTemperature);
62.     Serial.println("*C");
63.
64.     Serial.print(rawPressure = ");
65.     Serial.print(rawPressure);
66.     Serial.print(", realPressure = ");
67.     Serial.print(realPressure);
68.     Serial.println("Pa");
69.
70.     Serial.print(.absoluteAltitude = ");
71.     Serial.print(absoluteAltitude);
72.     Serial.print("m, relativeAltitude = ");
```

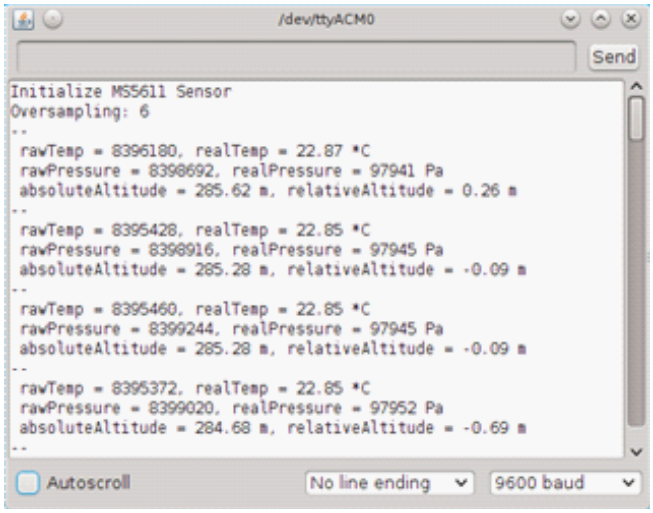

Figura 3.21: Datos seriales. (Fuente: Elaboración propia)

```
73.     Serial.print(relativeAltitude);
74.     Serial.println("m");
75.
76.     delay(1000);
77.     }
```

El código muestra la información en el terminal serial (ver figura 3.21) y de esa forma sabemos que está trabajando correctamente

3.7. OLED LCD

Para la visualización de datos en la estación tierra se investigó un dispositivo por el cual se pudiera mostrar la información de manera clara, la forma de conexión es mediante I2C, puede trabajar con voltajes entre 3V a 5V. En ella se mostrará información relevante que se necesita, como el modo de operación.

3.7.1. Conexiones

La pantalla va a alimentación de 5V y tierra, y sus conexiones van al puerto I2C de la tarjeta como en la figura, se puede conectar en paralelo con otros dispositivos I2C (ver figura 3.22).

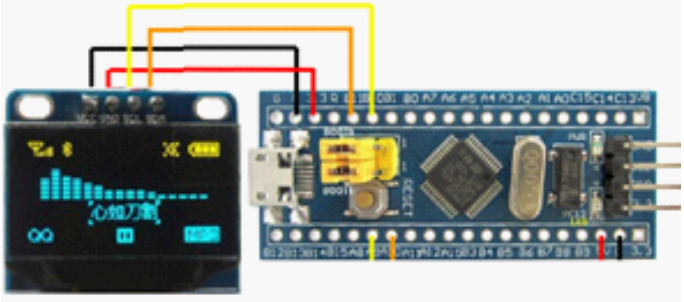

Figura 3.22: Conexión pantalla LCD. (Fuente: Elaboración propia)

3.7.2. Librería

Se investigó una librería que pudiera trabajar con la tarjeta STM32F103, debido a que este dispositivo requiere de un código especial para cada tarjeta dada su frecuencia de trabajo, interrupciones y buffer de almacenamiento de los caracteres a escribir. El repositorio se encuentra en el siguiente link: https://github.com/adafruit/Adafruit_SSD1306

La biblioteca puede imprimir texto, mapas de bits, píxeles, rectángulos, círculos y líneas. Utiliza 1K de RAM, ya que necesita almacenar toda la pantalla.

Ejemplo:

```
1.  /*********************************************************************
2.  This is an example for our Monochrome OLEDs based on SSD1306
    drivers
3.
4.  Pick one up today in the adafruit shop!
5.  ------> http://www.adafruit.com/category/63_98
6.
7.  This example is for a 128x64 size display using I2C to communicate
8.  3 pins are required to interface (2 I2C and one reset)
9.
10.  Adafruit invests time and resources providing this open source code,
11.  please support Adafruit and open-source hardware by purchasing
12.  products from Adafruit!
13.
14.  Written by Limor Fried/Ladyada for Adafruit Industries.
15.  BSD license, check license.txt for more information
16.  All text above, and the splash screen must be included in any redis-
    tribution
```

```
17.  **********************************************************************/
18.
19.  #include <SPI.h>
20.  #include <Wire.h>
21.  #include <Adafruit_GFX.h>
22.  #include <Adafruit_SSD1306.h>
23.
24.  #define OLED_RESET 4
25.  Adafruit_SSD1306 display(OLED_RESET);
26.
27.  void setup() {
28.  lcd.begin(SSD1306_SWITCHCAPVCC, 0x3C);
29.  lcd.setTextSize(0);
30.  lcd.setTextColor(WHITE,BLACK);
31.  }
32.
33.  void loop() {
34.  lcd.clearDisplay();
35.  lcd.setCursor(0,0);
36.  lcd.print(-FOLLOWME V2.0–"); lcd.println();
37.  lcd.print(Çodigo de ejemplo"); lcd.println();
38.  lcd.print("12345678912345678"); lcd.println();
39.  lcd.print(.ABCDEFGHIJKLMNOPQ"); lcd.println();
40.  lcd.display();
41.  delay(1000);
42.  }
```

3.8. JOYSTICK

Se utilizaron en el prototipo dos Joystick para realizar la función que realizaría un radio control convencional ya que el funcionamiento está basado en el movimiento en dos dimensiones de una palanca, este movimiento es capturado por dos potenciómetros (uno para cada movimiento), de este modo se entiende que para cada movimiento en cada dirección será regulado un potenciómetro.

Con este módulo joystick se puede utilizar el cambio de valor resistivo para tomar lectura con dos entradas analógicas en el Arduino.

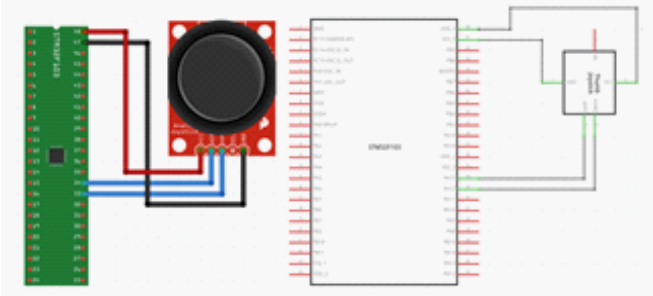

Figura 3.23: Conexiones joysticks. (Fuente: Elaboración propia)

3.8.1. Conexiones

Este módulo de joystick cuenta con cinco pines los cuales se enumeran tomando de referencia de izquierda a derecha (ver figura 3.23):

1. GND: Pin conectado a tierra.
2. +5V: pin de alimentación (5v).
3. VRx: pin de lectura de potenciómetro para el eje de las x's.
4. VRy: pin de lectura de potenciómetro para el eje de las y's.
5. SW: es un pin adicional que se utiliza para un push button en la parte inferior.

3.8.2. Librería

Para utilizar este dispositivo solo se necesitó realizar lecturas analógicas y efectuar un mapeo a valores deseados de trabajo.

```
1.  /*GND - GND
2.  Vcc - 5v
3.  VRx - A0
4.  VRy - A1
5.  SW - D9
6.  */
7.
8.  const int pinLED = 13;
9.  const int pinJoyX = A0;
10.   const int pinJoyY = A1;
11.   const int pinJoyButton = 9;
```

```
12.
13.   void setup() {
14.   pinMode(pinJoyButton , INPUT_PULLUP); //activar resistencia
pull up
15.   Serial.begin(9600);
16.   }
17.
18.   void loop() {
19.   int Xvalue = 0;
20.   int Yvalue = 0;
21.   bool buttonValue = false;
22.
23.   //leer valores
24.   Xvalue = analogRead(pinJoyX);
25.   delay(100); //es necesaria una pequeña pausa entre lecturas analóg-
icas
26.   Yvalue = analogRead(pinJoyY);
27.   buttonValue = digitalRead(pinJoyButton);
28.
29.   //mostrar valores por serial
30.   Serial.print("X:");
31.   Serial.print(Xvalue);
32.   Serial.print("| Y: ");
33.   Serial.print(Yvalue);
34.   Serial.print("| Pulsador: ");
35.   Serial.println(buttonValue);
36.   delay(1000);
37.   }
```

3.9. RADIO CONTROL PPM

Gracias al uso de los joystick en dicha investigación, se logró sustituir un radio control convencional del drone y mandarla a la controladora naza para controlar el drone, hoy en día nuestras emisoras y receptores, ofrecen varios sistemas de conexionado con nuestros modelos. En el “aeromodelismo tradicional”, se utiliza normalmente PWM. Servos motores y variadores son controlados de manera independiente y cada uno utiliza un canal. Con

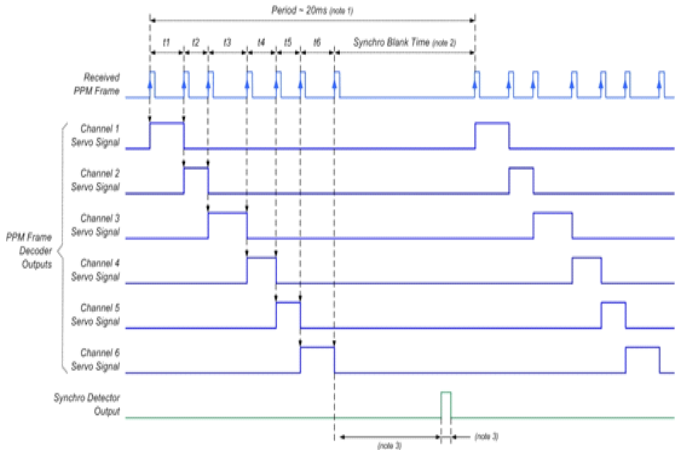

Figura 3.24: Conexiones PPM. (Fuente: http:// fpvmax.com/ 2017/07/28 /protocolos-comunicacion-drones/)

la emisora y sus configuraciones, mediante finales de carrera, mezclas, expos, rates, etc. conseguimos configurar nuestro aparato para que vuele según nuestro gusto.

Al aparecer la controladoras de vuelo, se vio que era esta la que gestionaba todos ellos y se podía eliminar cableado con otras formas de modulación. Con PPM y S-Bus se pudo desarrollar los micro-receptores.

Es por ello que se creó una librería que genera la señal PPM hacia la receptora (ver figura 3.24).

Solo es necesario un pin de señal y uno de tierra para poder mandar la señal PPM a la controladora de vuelo NAZA (ver figura 3.25).

3.9.1. Librería

En Arduino, los timers se usan para el manejo de las funciones de tiempo, para el manejo de servos y para generar señales periódicas (PWM). Por lo tanto tenemos esta restricción si queremos usar timers. Si se usa el Timer0, estaríamos alterando a las funciones: micros(), milis() y delay().

En orden de importancia tendríamos al Timer-0, Timer-1 y Timer-2.

Para la librería utilizamos el Timer-2 este se configura con el siguiente código:

```
1.  #define PinPPM PB13 //Pin de salida
2.  #define PPMCompleto 20000 //Duracion completa de la señal
3.  #define PPMpause 400 //Pausa entre señales
4.
5.  // Setup PPM Timer
```

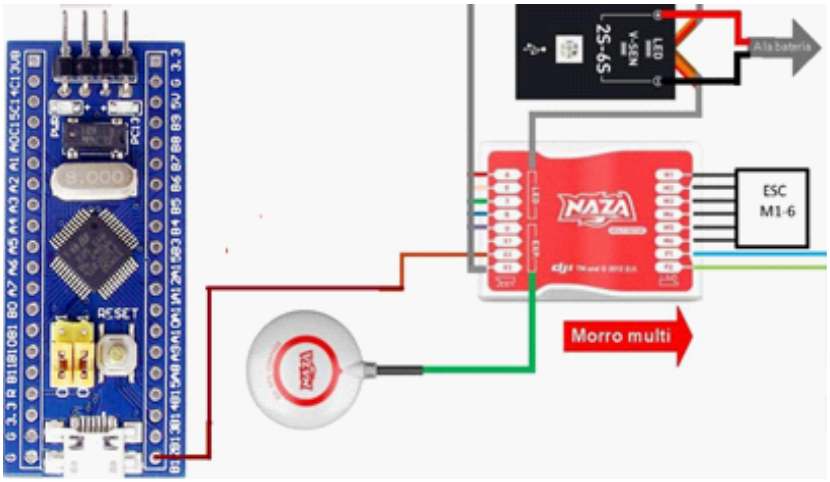

Figura 3.25: Conexión NAZA. (Fuente: Elaboración propia)

```
6.  Timer2.setChannel1Mode(TIMER_OUTPUTCOMPARE);
7.  Timer2.setPeriod(PPMpause); // in microseconds
8.  Timer2.setCompare1(1); // overflow might be small
9.  Timer2.attachCompare1Interrupt(PPM);
10.  Timer2.refresh();
11.  pinMode(PinPPM, OUTPUT);
```

Se irán montando las señales en el Pin de PPM esto se realiza con una frecuencia de 50Hz dentro se integraran 8 señales, entre cada señal según la configuración de la pausa entre ellas habrá un tiempo de 400uS y al final se tendrá que completar el ciclo de 20mS.

```
1.  ////////////////////////////////////////////////Salida PPM
2.  void PPM(void)
3.  {
4.  static int EstadoPinPPM = HIGH;
5.  static int p=0;//para escritura de pulso ppm
6.  static int PPMsobrante=0;
7.
8.  EstadoPinPPM ^=1;//cambia el estado para el pin
9.
10.
11.
12.
13.
```

```
14.  if(EstadoPinPPM==HIGH){
15.  Timer2.setPeriod(RC[p]-PPMpause); //cambia el tiempo del ciclo a
la variable del canal PPM i
16.  p++;}
17.  else if(p>chanel_number){p=0;
18.  int sumaPPM=0;
19.  for(int x=0;x<chanel_number;x++)
20.  {sumaPPM=sumaPPM+(RC[x]);} //creamos una sumas de la du-
racion del tiempo de cada canal
21.  PPMsobrante = PPMCompleto - (PPMpause * chanel_number) -
PPMpause - sumaPPM; //Determinamos el tiempo que dura la pausa para que
se complete el ciclo de 20mS.
22.  Timer2.setPeriod(PPMsobrante-PPMpause);}
23.  else Timer2.setPeriod(PPMpause);
24.  Timer2.refresh();
25.  digitalWrite(PinPPM, !EstadoPinPPM);
26.  }
```

3.10. HC-12

3.10.1. Selección

Para la comunicación entre el Multicoptero y la estación tierra se realizaron pruebas para obtener el mejor dispositivo de comunicación entre los que destacan el modulo Bluetooth HC-06, los módulos HC-12 y los módulos NRF24L01, se realizaron pruebas y los resultados se describe en el cuadro 3.5.

Después de hacer pruebas de comunicaciones y registrado los resultados, se observó que el módulo HC-12 tiene mejores resultados y practicidad de uso, esta conexión al igual que el bluetooth solo necesita de 4 pines en cambio el módulo NRF24L01 requiere de 6 pines, el emparejamiento del módulo HC-12 y del NRF24L01 son rápidos mientras que el bluetooth tiene un tiempo de conexión largo, y la distancia que pueden tener entre ellos sin desconexiones es mayor a 100 metros. Por lo se eligieron los módulos HC-12 (ver figura 3.26).

3.10.2. Conexiones

En la figura 3.27se muestran los esquemas del circuito. La tensión de funcionamiento del módulo es de 3,2 V a 5,5 V y para un trabajo más estable, se recomienda utilizar un

	Bluetooth HC-06	Módulos HC-12	Módulos HC-12
Distancia	6 a 10 metros	1000 metros	1000 metros
Conexión simultánea	2 dispositivos	2 dispositivos	2 dispositivos
Frecuencia	2.4 GHz	433.4 MHz a 473.0 MHz	433.4 MHz a 473.0 MHz
Protocolo	Serial	Serial	Serial
Voltaje de trabajo	3.6 V a 6 V	5V	5V
Precio	150 pesos c/u	250 pesos el par	250 pesos el par
Emparejamiento	Hay que ingresar contraseña y emparejar antes	Hay que poner ambos en el mismo canal	Hay que poner ambos en el mismo canal

Cuadro 3.5: Comparación de dispositivos de comunicaciones

Figura 3.26: Módulos HC-12. (Fuente: https://howtomechatronics.com/ tutorials/arduino/ arduino-and-hc-12-long-range- wireless-communication-module/)

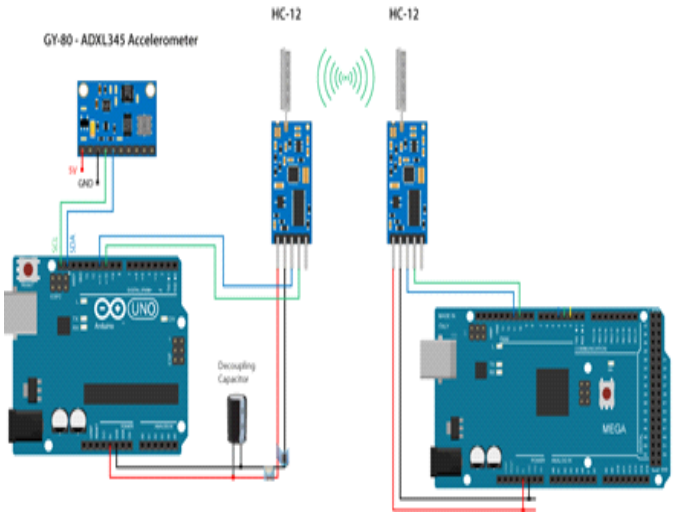

Figura 3.27: Esquema de Conexiones HC-12. (Fuente: https://howtomechatronics.com/tutorials/arduino/arduino-and-hc- 12-long-range-wireless-communication- module/)

condensador de desacoplamiento y una fuente de alimentación externa.

3.10.3. Librería

Estos módulos no necesitan una librería específica ya que únicamente necesitan un puerto UART y el código que se utiliza normalmente cuando se envían y reciben datos seriales.

A continuación se muestra el código de Arduino para un puerto UART, es decir, una comunicación básica entre los dos módulos que usa el Monitor de serie.

```
1.  /* Arduino Long Range Wireless Communication using HC-12
2.  Example 01
3.  by Dejan Nedelkovski, www.HowToMechatronics.com
4.  */
5.  #include <SoftwareSerial.h>
6.  SoftwareSerial HC12(10, 11); // HC-12 TX Pin, HC-12 RX Pin
7.  void setup() {
8.  Serial.begin(9600); // Serial port to computer
9.  HC12.begin(9600); // Serial port to HC12
10.  }
11.  void loop() {
12.  while (HC12.available()) { // If HC-12 has data
```

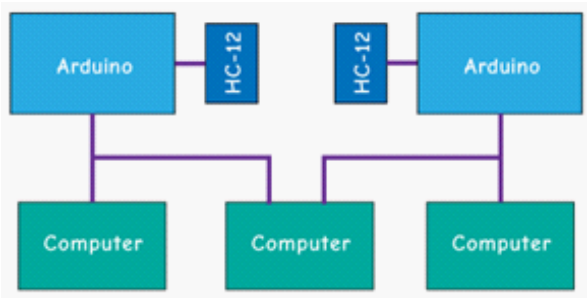

Figura 3.28: Conexión Arduino-computador. (Fuente: Elaboración propia)

```
13.     Serial.write(HC12.read()); // Send the data to Serial monitor
14.     }
15.     while (Serial.available()) { // If Serial monitor has data
16.     HC12.write(Serial.read()); // Send that data to HC-12
17.     }
18.     }
```

El mismo código se usa para ambos Arduinos. Podemos conectar los dos Arduinos en dos computadoras separadas pero también podemos usar una sola computadora.

En este caso, una vez que se conecta el primer Arduino a la computadora, se debe seleccionar el modelo, el puerto COM y cargar el código en el Arduino. Luego conectamos el segundo Arduino y tenemos que volver a arrancar el Arduino IDE para poder seleccionar el otro puerto COM al que está conectado nuestro segundo Arduino, y en seguida subir el mismo código.

Una vez que se tienen los dos IDE de Arduino en ejecución, podemos iniciar los monitores en serie y probar si la comunicación funciona correctamente. Todo lo que escribamos en el monitor serial se enviará de uno a otro Arduino (ver figura 3.28).

3.11. ALIMENTACIÓN

Se deben alimentar dos dispositivos, la estación tierra y el UAV, para el caso del UAV se utiliza como fuente principal una batería de Polímero de Litio a 3s de entre 2200mHA a 5500mHa (ver Figura 3.30), esta batería suministra energía al UAV para su completo funcionamiento y un regulador de 5V a 3A para suministrar energía a los componentes de la controladora de vuelo.

Se utilizaron baterías de LiPo como sistema de aporte energético porque son fáciles de recargar, y mediante su corriente es posible alimentar todos los circuitos de drones

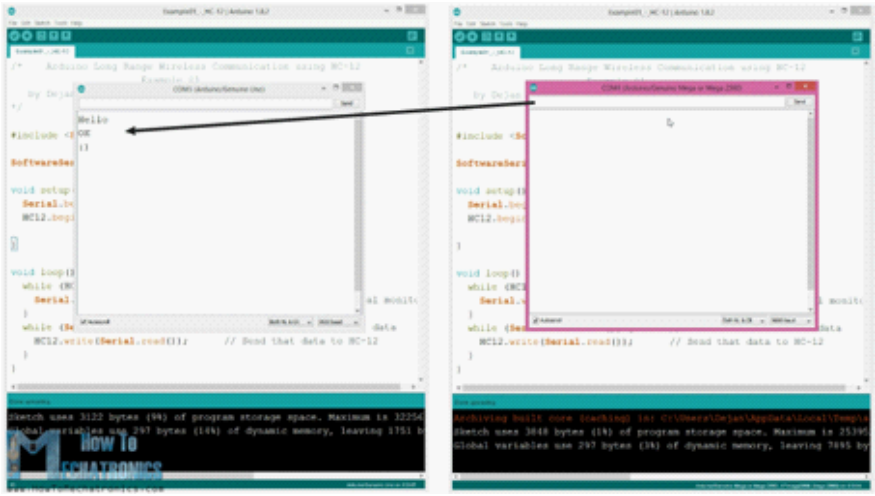

Figura 3.29: Monitores seriales. (Fuente: Elaboración propia)

Figura 3.30: Bateria Lipo 3S. (Fuente: Elaboración propia)

Figura 3.31: Bateria 1S. (Fuente: https://laniakea.mx/bateria -lipo-3.7v-260mah-jjrc-h8)

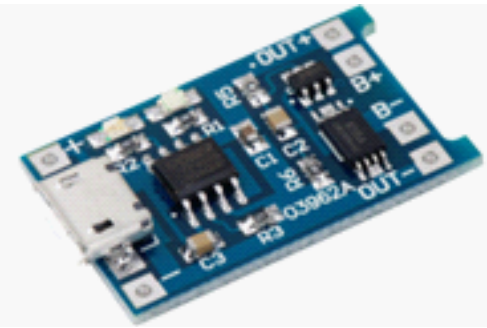

Figura 3.32: Chip de protección de batería DW01. (Fuente: Elaboración propia)

de mediano y pequeño tamaño. El mercado está plagado de baterías LiPo, que son las que ofrecen una mejor relación entre la energía que son capaces de almacenar y el peso y volumen que ocupan, dentro de unos costes accesibles para el gran público.

Para la estación tierra se optó por una batería que fuera pequeña capaz de suministrar mas de 3V por lo cual se encontró una batería de lipo muy compacta la cual ofrece un voltaje nominal de 3.7V (ver figura 3.31).

Para recargar esta batería se investigaron varios tipos de cargadores y se decidió utilizar el cargador de baterías USB basado en el chip TP4056 y el chip de protección de batería DW01 (ver figura3.32), el módulo ofrece una corriente de carga de 1A y corte automático cuando se encuentre cargada la batería.

3.11.1. Instalación

Este dispositivo va dentro de la estación tierra, sus conexiones son en paralelo con la batería y el circuito a alimentar (ver figura 3.33), ya que el microcontrolador cuenta con entradas analógicas se realizó una lectura de la batería para saber cuándo hay que cargarla,

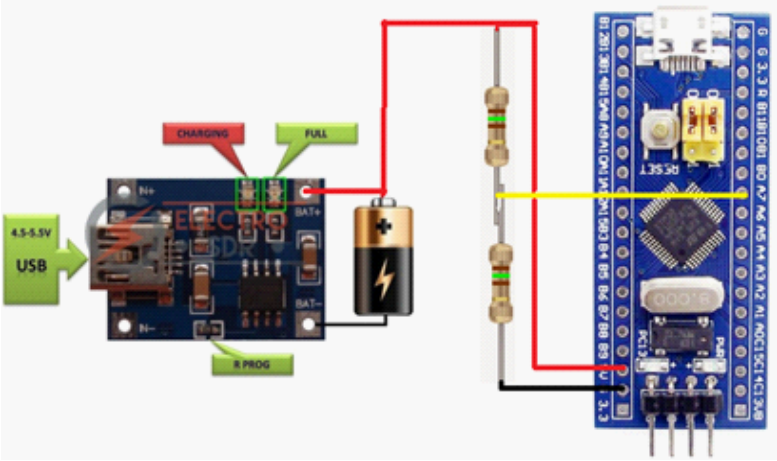

Figura 3.33: Diagrama de conexión LiPo. (Fuente: Elaboración propia)

utilizamos un circuito resistivo en serie para reducir el voltaje ya que el máximo voltaje de la tarjeta son 3.3V y la batería entrega un máximo de 4.2V.

3.11.2. Código de alimentación

El código es el siguiente, en él se calcula el voltaje y el porcentaje de la batería:

```
1.  /* Sensor de voltaje para baterías Li-Po de una celda (1S).
2.  * Se toma el voltaje de la batería a través de la entrada A0.
3.  * Se muestra el voltaje y el porcentaje de carga
4.  * 4.2 = 100 % (máximo voltaje, máxima carga)
5.  * 3.2 = 0 % (mínimo voltaje de seguridad)
6.  *
7.  * IMPORTANTE: La tensión que entra por USB o por el pin de 5V,
8.  * afecta a la precisión, ya que ésta puede ser difente de 5.0V en algunos
mV.
9.  * El ADC utiliza el voltaje de alimentación como voltaje de referencia.
10.  * Utilizamos un potenciómetro para regular el voltaje de offset, y
regular
11.  * la precisión, con la ayuda de un multímetro.
12.  *
13.  */
14.
```

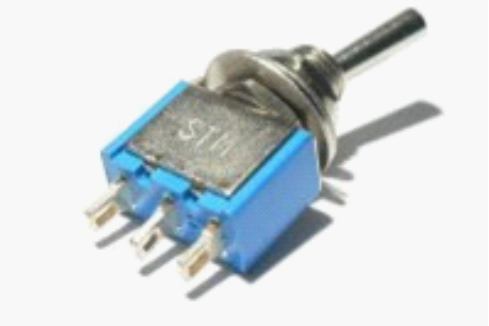

Figura 3.34: Interruptor de 3 posiciones. (Fuente: Elaboración propia)

15. float voltaje= ((((analogRead(Pin_Batery)*6.7)/4096.0)));//Como se calculo este dato

16. float porciento= map(voltaje*100,3.4*100,4.2*100,0,100*100);

3.12. MODOS DE VUELO

En general, el propósito del multirotor es que siga a una persona, pero para ellos se necesitan diferentes modos de vuelo que realizan acciones como control de vuelo, vuelo autónomo y Return Home.

Para cambiar entre modos se cuenta con un interruptor de 3 posiciones en la estación tierra (ver figura 3.34), se puede visualizar el modo de vuelo en la LCD de la estación tierra.

3.12.1. Modo Radio

Este modo se utiliza para tener completo control del Multirotor, lo que permité para posicionarlo en el punto en el que se desee realizar el seguimiento, regresarlo de forma segura a tierra o controlarlo para no sufrir algún accidente (ver figura3.35).

En este modo no actúan los sensores como el GPS o barómetro, solo los Joystick los cuales controlan el PITCH, YAW, ROLL y THROTLE, trabaja como modo estable si no se tocan los joystick ya que nivela el drone y lo mantiene en un punto fijo.

3.12.2. Modo Return Home

El modo RTL (modo Return Home) hace que Multicoptero salga de su posición actual para pasar el cursor sobre la posición de inicio. El comportamiento del modo RTL puede controlarse mediante varios parámetros ajustables.

Figura 3.35: Control drone. (Fuente: https://realcooltech.sg/products/ parrot-bebop-drone-with-skycontroller-red)

Cuando se selecciona el modo Return Home, el drone regresará a la ubicación de inicio. El helicóptero subirá primero a una altura de 15 metros antes de regresar a el punto despegue o mantendrá la altitud actual si la altitud actual es mayor que 15 metros (ver figura 3.36).

En el modo Return Home ordenará al robot que regrese a la posición inicial, lo que significa que volverá a la ubicación donde estaba armado. Por lo tanto, siempre se supone que la posición inicial es la ubicación real de despegue del GPS de su helicóptero, sin obstrucciones y lejos de las personas.

3.12.3. Modo Follow-Me

El modo follow hace posible que el multicoptero siga a la estación tierra a donde se mueva, este modo se selecciona después de que el piloto dirigió al drone al lugar donde se a partir de ese punto después de seleccionar modo follow-me el drone mantendrá la misma distancia con la estación (ver figura 3.37). Para esto es necesario el uso del GPS, barómetro, y todo el algoritmo desarrollado en este trabajo.

3.13. INSTRUCCIONES

1. El multicoptero en el suelo, establece una conexión con la estación tierra.
2. Establece uno de tus modos de vuelo a Modo Radio.
3. Se debe tener en cuenta que los dispositivos GPS han establecido la conexión.

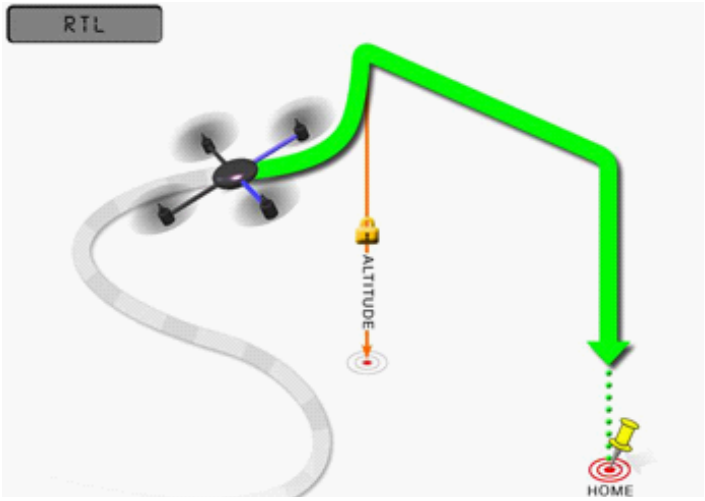

Figura 3.36: Return Home. (Fuente: http://ardupilot.org/copter/docs/ rtl-mode.html)

Figura 3.37: Modo Follow-me. (Fuente: http://blog.parrot.com/2016/11/03/ con-follow-tu-drone-se-convertira- en-tu-fiel-companero-de-aventuras/)

4. Despega, y una vez en el aire, cambia a modo Follow-me. (A la altitud suficiente para tener seguro que mientras te está siguiendo).

5. Para descender el drone se debe regresar al modo Radio para aterrizar el drone manualmente o cambiar al modo Return Home para que el drone descienda al punto de partida automáticamente.

3.14. PID

Para cada una de las variables a controlar necesitamos un PID, los cuales fueron los siguientes:

- Altura
- Dirección de UAV
- Distancia entre dispositivos

Estos fueron sintonizados de forma empírica ya que se desarrolló el prototipo en forma real.

$$u(t) = K\,[e(t)+]\,\frac{1}{T_i}\int_0^I e(\tau)d\tau + T_d\frac{de(t)}{dt}) \tag{3.4}$$

3.14.1. Sintonización del Controlador PID

Dentro del código se consideran las constantes PID que sean óptimas para cada caso, todo esto dependerá de la plataforma donde se implementen de tal manera que de esta manera se elegirán o cambiaran las constantes según los intereses.

Para la sintonización de los valores para las constantes KP, KI, KD y Tiempo de muestreo se realizaron en cada una de las variables los siguientes pasos según el PID.

1. Control Proporcional: Este control es el más importante, ya que es el que mueve el drone la mayor parte del recorrido. El funcionamiento del control proporcional va en base a la actuación del valor KP , es decir, al aumentar el valor KP se genera un ligero descontrol ocasionando que se mueva un poco mas de lo deseado, pero nunca sobrepasara el límite. En caso de que el motor se encuentre libre de su efecto será que por la propia inercia del movimiento este sobrepasara de la posición designada y generará oscilaciones cada vez más lentas, ocasionando que en lugar de llegar a un punto específico este se posicione en otro lugar.

2. Control derivativo (diferencial): Se denomina derivada ya que la velocidad así es clasificada. La función de este control es restar la velocidad al motor en la medida en que se acerque al punto asignado. Cuando se apliquen valores demasiado altos en la constante derivativa (KD) se producen comportamientos ruidosos u oscilaciones arbitrarias. Es importante recordar que este control es muy sensible a los cambios debido a que trata de darle al motor un a velocidad óptima para obtener una llegada correcta.

3. Control integral. Este control actúa en el motor haciendo que en caso de que aun falte velocidad para llegar al punto designado se eleve el valor del PWM, hasta lograr que el motor se mueva lo suficiente para llegar a su destino final, es decir si se pone un valor muy pequeño a la constante integral (KI) hace un esfuerzo progresivo que va ir progresando, en cambio si colocamos un valor alto en KI el motor generará oscilaciones exageradas como sucede en KP y KD.

4. Tiempo de muestreo: Es la unidad de tiempo que le dicea l Arduino cada cuanto tiempo debe realizar los cálculos, actúa sobre el control integral y derivativo. En medida que se modifican los parámetros se generaran cambios y comportamientos en el motor según la variable modificada.

3.15. COMBINACIÓN DE CÓDIGO

Una vez que se tiene cada dispositivo trabajando individualmente se pasa cada librería a un programa final que sería la combinación de todos los códigos para tener la función final del prototipo. El diagrama de funcionamiento básico (ver figura 3.38), sería el siguiente:

En los cuadros 3.6 y 3.7 se resume el código desarrollado para el funcionamiento del prototipo cuando se hace el conjunto del código.

3.16. DISEÑO DE LA INTERFAZ DE VISUALIZACIÓN LABVIEW

Se realizó una investigación de los lenguajes de programación existentes y llegamos a la conclusión de implementar LabView ya que, cumple con las expectativas del sistema.

Las funciones que debe cumplir el software es la siguiente:

- Debe recibir datos por medio del puerto USB

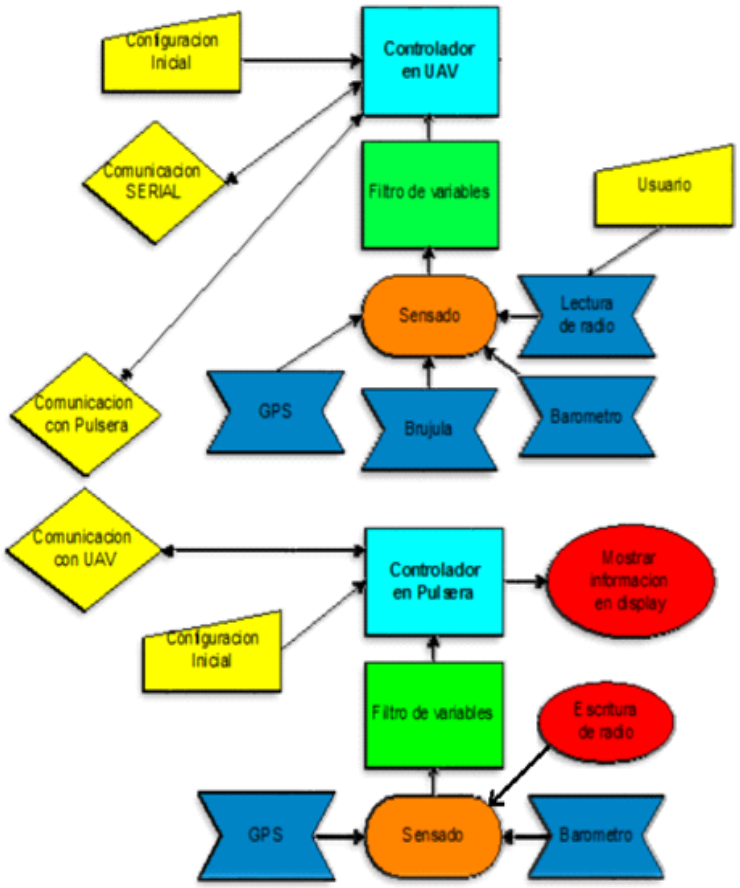

Figura 3.38: Diagrama de trabajo. (Fuente: Elaboración propia)

Nombre del archivo	Función
Pulsera V2	Inicializaciones y programa principal, instrucciones de modos de control, declaración de variables auxiliares, asociación de modos de control a funciones de los controladores.
GPS	Actualiza GPS, lee los datos obtenidos, los pone en las variables de trabajo.
HC12	Recibe y envía los datos de las variables que se necesitan entre los dispositivos.
Joysticks.cpp	Lee, actualiza y mapea los valores de los joystick, lee el voltaje de la batería.
Joysticks.h	Declara los pines, variables y funciones.
LCD	Actualiza LCD mostrando los datos de la pantalla principales
Barómetro	Lee y filtra la variable de altura.

Cuadro 3.6: Programa final Estación Tierra

Nombre del archivo	Función
Drone-Follow-V2	Inicializaciones y programa principal, instrucciones de modos de control, declaración de variables auxiliares, asociación de modos de control a funciones de los controladores.
GPS	Actualiza GPS, lee los datos obtenidos, los pone en las variables de trabajo.
HC12	Recibe y envía los datos de las variables que se necesitan entre los dispositivos.
KalmanFilterVA.cpp	Filtra la variable de posición mediante filtro de Kalman
KalmanFilterVA.h	Declaración de variables y funciones para el filtro de Kalman
MatrixMath.cpp	Parte de la librería Kalman para cálculos aritméticos
MatrixMath.h	Declaración de variables y funciones para cálculo de matrices
NazaDecoderLib.cpp	Librería para lectura de datos de GPS NAZA y su protocolo
NazaDecoder.Lib	Declaración de variables y funciones para librería NazaDecoder
PID	Calculo y declaración de variables de PID
Radio.h	Declaración de variables y funciones para librería Radio
Radio	Realiza el montaje para la señal de radio PPM
Barómetro	Realiza la lectura de la altura
Serialpc	Transmite y recibe los datos que se muestran por el puerto serial que va al PC

Cuadro 3.7: Programa final Drone

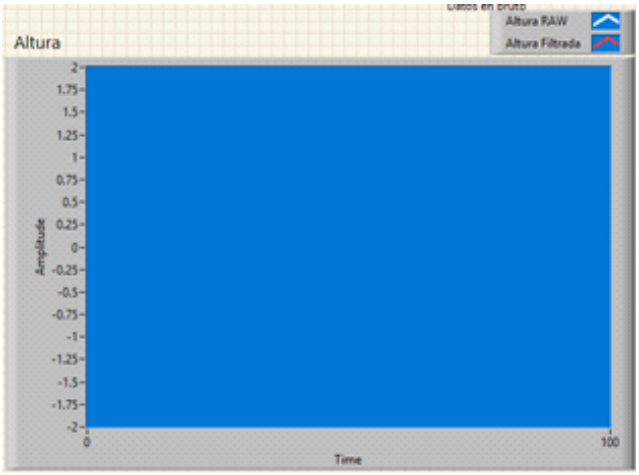

Figura 3.39: Gráficas virtuales. (Fuente: Elaboración propia)

- Estos datos deben ser mostrados o visualizados en indicadores y graficas virtuales en PC como:

  1. Altura real
  2. Filtro de Altura
  3. Joystick Pitch
  4. Joystick Yaw
  5. Joystick Roll
  6. Joystick Throtle
  7. Ch6
  8. Ch7
  9. Magnetómetro

- Debe realizar un historial de los datos en un archivo de Excel

3.17. GRAFICAS VIRTUALES

Las gráficas que se utilizaron para cada variable (ver figura 3.39), son de tipo wareform chart y se visualizan de igual forma en el panel frontal, de manera que se puede observar un historial de las variaciones de la altura filtrada y sin filtrar para poder comparar como está funcionando el filtro y poder realizar ajustes.

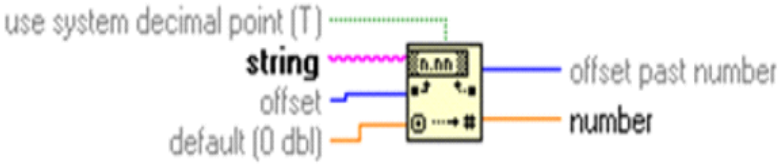

Figura 3.40: Frac/exp de cadena a número. (Fuente: Elaboración propia)

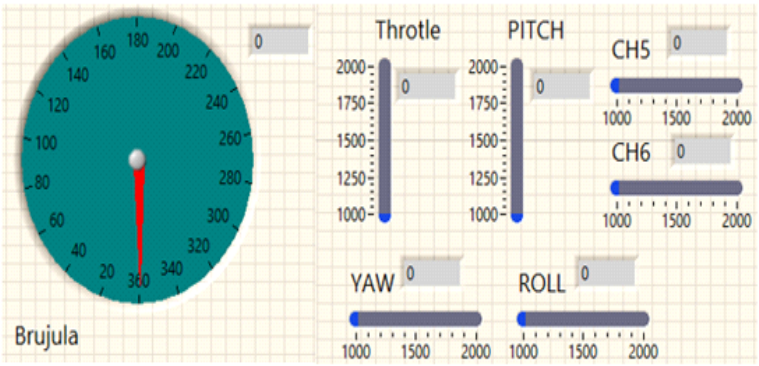

Figura 3.41: Indicadores virtuales. (Fuente: Elaboración propia)

3.17.1. String

El string es una cadena de datos que se desplaza en forma serial y secuenciada. Se utilizó para recibir los datos que transmite el microcontrolador y se visualizan en panel de bloques).

3.17.2. Convertidos de "string" carácter a un valor numérico

El Fract/Exp string to number (ver figura3.40), se utilizó para convertir un carácter numérico de la cadena seleccionada a un valor numérico entero o flotante para ser aplicado en operaciones, nodos formula o simplemente para visualizarse en los indicadores virtuales o en las graficas.

3.17.3. Indicadores virtuales

En la figura 3.41se muestra el panel frontal con los indicadores que se utilizaron para visualizarse cada una de las variables de la señal del radio, con un valor numérico y un deslizamiento del indicador.

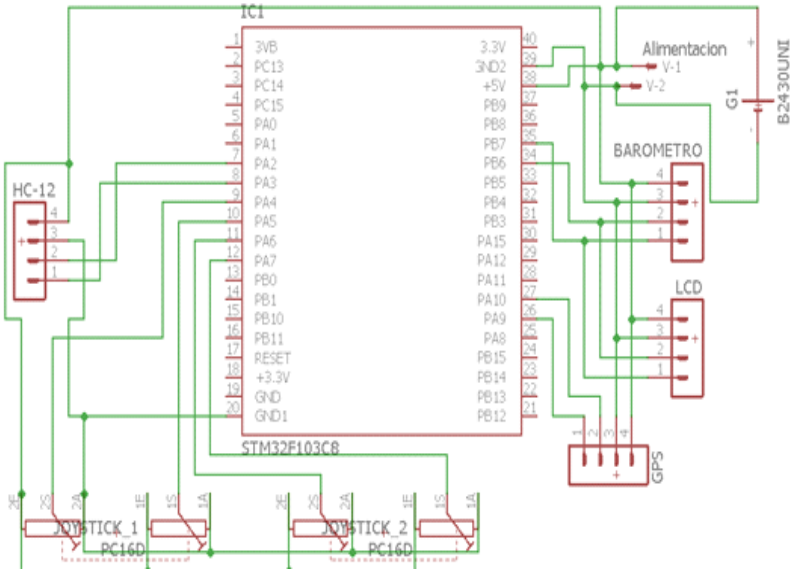

Figura 3.42: Circuito esquemático Estación Tierra. (Fuente: Elaboración propia)

3.18. DISEÑO DEL PROTOTIPO

Una vez que se realizaron las pruebas del funcionamiento de cada componente y del código completo compilado se realizó el siguiente diagrama de conexiones para la plataforma final (ver figura 3.42):

Los compontes que integra el dispositivo son los siguientes:

- GPS NEO M8N
- Barómetro MS5611
- 2 Joysticks
- OLED LCD
- Pines de Alimentación
- Batería LiPo 1S
- Cargador de Batería Lipo
- HC-12
- STM32F103C

El dispositivo que se desarrolló para el UAV es parecido al que va en la estación tierra solo que este cambia su en alguno componentes como el GPS.

Los componentes que compone el dispositivo son los siguientes:

- GPS NAZA V2
- Barómetro MS5611
- STM32F103C
- HC-12

3.19. CREACIÓN DE ROBOT CON TARJETA ROBOT-KIDS

El prototipo requirió de una etapa de pruebas ya que el dispositivo final era un UAV, se desarrolló una plataforma móvil terrestre, se tomaron en cuenta las necesidades básicas en la plataforma tales como:

- Push botón
- Potenciómetros
- Leds
- Motor con reductor
- Puente H L293D
- Arduino
- Puerto para alimentación
- Puerto de entrada PPM

Con los cuales se puedan desempeñar y experimentar para el prototipo final sin tantos riesgos programación, trabajar con actuadores, manejar entradas analógicas y digitales. Cuenta con la característica de que los puertos para conectar sensores no están limitados a los que tenemos en la lista y que la tarjeta posee protecciones en caso de conexiones de alimentación inversas.

Se desarrolló el circuito esquemático de la figura 3.43 en el software EAGLE® el cual integra puertos para conectar los sensores y componentes que contiene la tarjeta de manera fija.

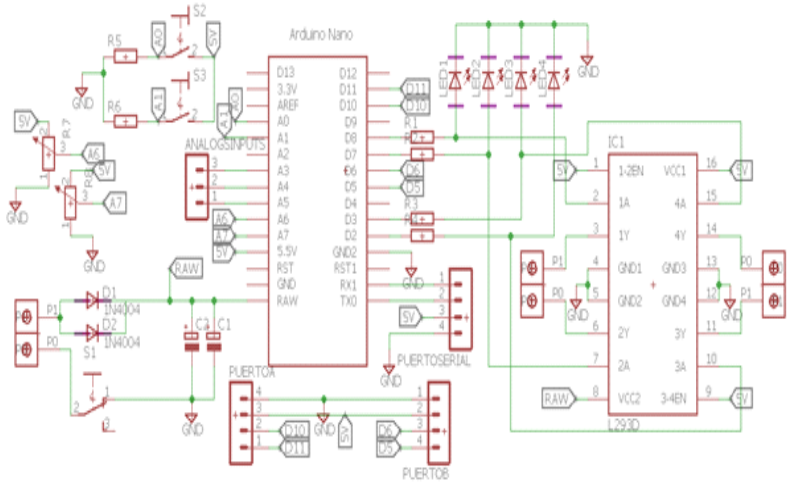

Figura 3.43: Circuito esquemático de la tarjeta RobotKids. (Fuente: Elaboración propia)

La PCB de la figura 3.44 se diseñó en el software PCB Wizard®, tratando de que el acomodo de los componentes fuera lo más práctico posible, colocando en el centro de la tarjeta a alimentación y el Arduino nano; se ubicaron las conexiones de los motores a cada costado para facilitar conectar un motor de lado izquierdo y otro de lado derecho, se igual manera se situaron los puertos para los sensores; se colocaron LEDs en orden de modo de que se puedan realizar prácticas básicas y sec0uencias; los interruptores y los potenciómetros se instalaron juntos en el área de entradas analógicas. Para las conexiones de los motores y de la alimentación se utilizaron Bloques Terminales, que facilitan conectar cableado a la tarjeta con un desarmador.

Cada tarjeta dispone de tiras de pines hembra para colocar el Arduino Nano y el puente H, pensado en que en caso de corto circuito pueda ser reemplazado o en caso de contar con una nueva versión o de que se quiera elaborar otra aplicación con los dispositivos. La tarjeta terminada ya ensamblada se puede apreciar en la figura 3.45.

Como fuente de poder se utilizó una batería de Polímero de Litio 2s 7.4V a 370mAh como la de la figura3.46; esta es una batería que proporciona una taza alta de descarga y un peso de solo 27g.

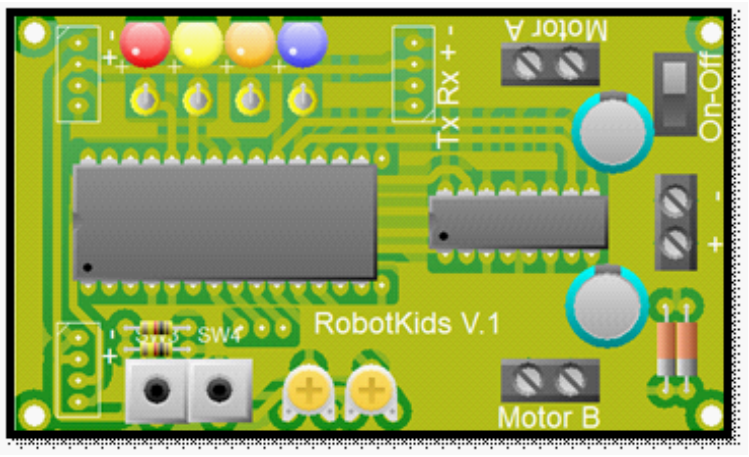

Figura 3.44: PDB RobotKids. (Fuente: Elaboración propia)

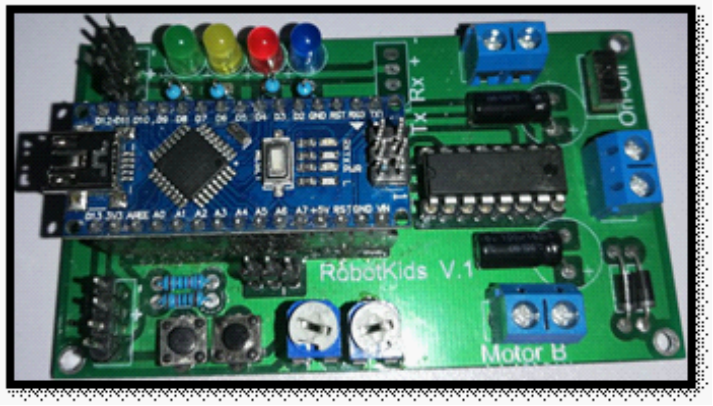

Figura 3.45: RobotKids. (Fuente: Elaboración propia)

Figura 3.46: Bateria liPo. (Fuente: Elaboración propia)

Figura 3.47: a) Motor Reductor y b) Rueda. (Fuente: http://www.dx.com/es/p/ dc-drive-gear-motor-tyre-for-smart -car-robot-black-yellow-433776)

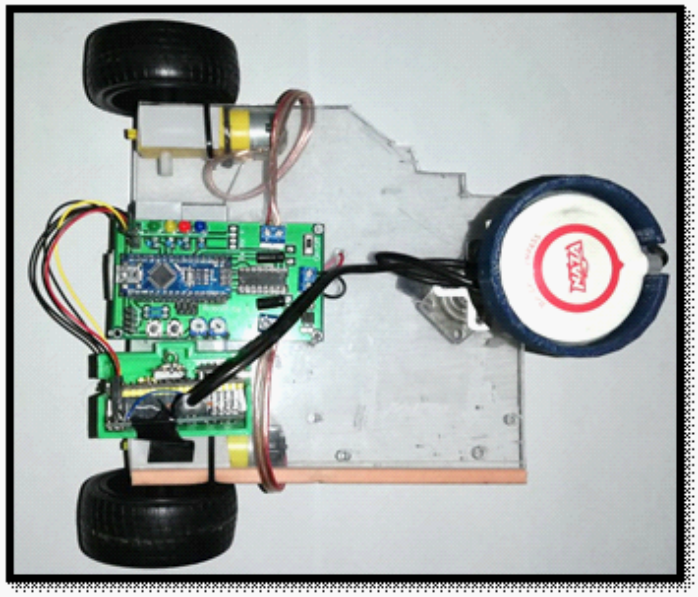

Figura 3.48: Prototipo armado. (Fuente: Elaboración propia)

3.20. SISTEMA MECATRÓNICO

Los motores con reductor y la rueda loca utilizados fueron los de la figura3.47, los motores son económicos y son capaces de dar el torque necesario para mover el prototipo y la rueda como punto de apoyo para reducir la fricción siendo el modelo para el movimiento de tipo diferencial.

Se planteó el diseño del uso en un chasis básico hecho de acrílico. Este contendrá los motores en posición diferencial, una rueda loca, batería, tarjeta y los sensores que se vallan a utilizar. El prototipo ensamblado se muestra en la figura 3.48.

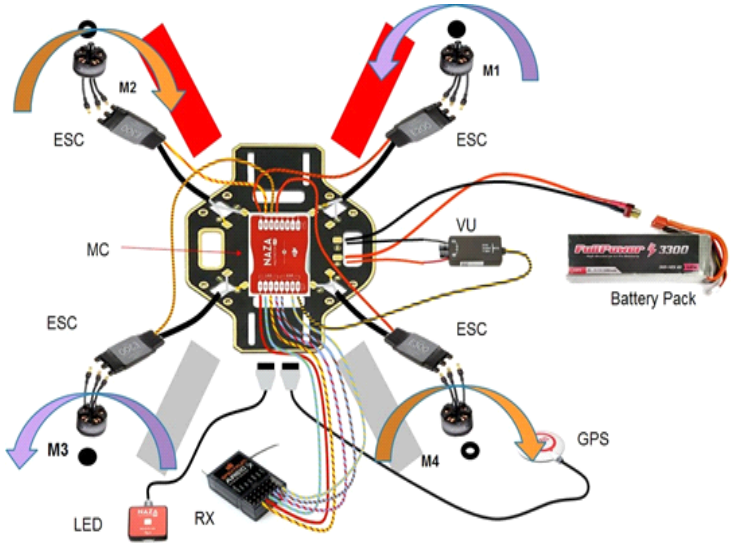

Figura 3.49: Conexión de componentes. (Fuente: https://sites.google.com/site/ making-drones/home/build)

3.21. ENSAMBLE DE UAV

Para el ensamblado del multirotor se necesitaron los siguientes materiales:

- Propelas 10”
- Frame DJI F450
- 4 ESC
- 4 Motores
- Bateria y medidor de LiPo 3S
- Tornilleria

Posteriormente se procede a la armadura del multirotor de la siguiente manera (ver figura 3.49):

1. Soldadura: Existen 12 puntos que requieren soldadura, se muestra en la fig 4-52.
2. Montaje de la estructuras:

   a) Colocar las hélices 1 y 3 se colocan en sentido anti-horrio y las hélices 2 y 4 se ponen en sentido a las manecillas del reloj. (ver figura 4-53)

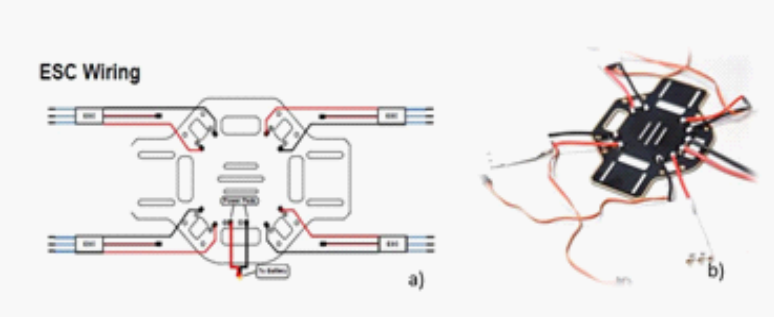

Figura 3.50: ESC: a) Puntos de soldadura y b) ESC's soldado. (Fuente: https:/ /sites.google.com/ site/ makingdrones/home/build)

b) Montar los brazos en el tablero inferior, recordando que los motores van en sentidos opuestos

c) Conectar el ESC a los motores y ajústelos en los brazos con cable de plástico

3. Montaje electrónico:

a) Montar tablero superior

b) Ensamblar el controlador principal (NAZA)

c) Conecte el controlador principal con el 4 ESC

d) Ensamble y conecte la unidad versátil

e) Ensamble y conecte el LED

f) Ensamblar y conectar el GPS

g) Ensamblar y conectar el receptor

4. Ensamble inmediatamente la tabla superior con tornillos para evitar dañar el marco

5. En este proyecto se utilizó cinta doble cara para colocar la controladora de vuelo NAZA en el tablero superior como en la figura 3.52.

   En la figura 3.49, se muestra las conexiones finales del multirotor, asi como el orden de las hélices demostrando la oposición que deben tener para generar la elevación del mismo sin tener dificultades de armado.

6. Instalación del programa, este requiere el uso de PC WIindows MAC OS x 10.9 o superior

a) Usar la versión software para NAZA M V2, que se encuentra en el siguiente link http://www.dji.com/product/naza-m-v2/download

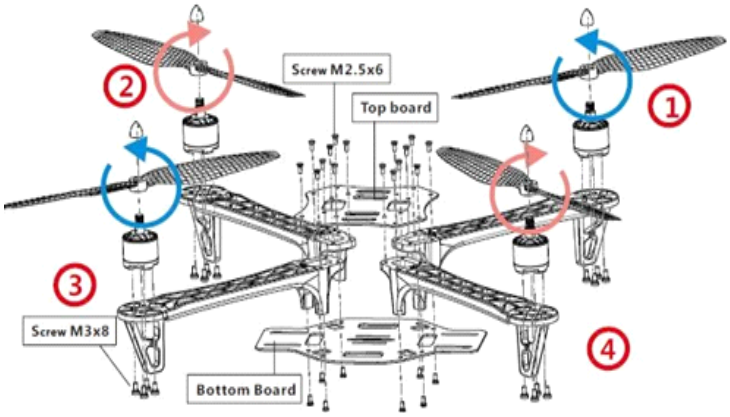

Figura 3.51: Montaje de la estructura. (Fuente: https://sites.google.com/site/ making-drones/home/build)

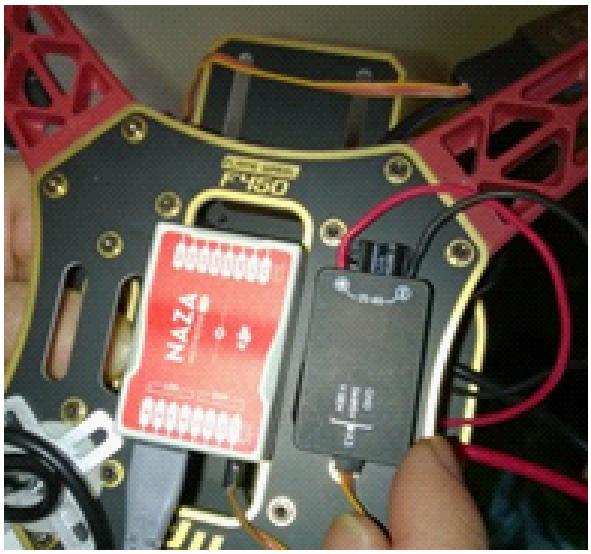

Figura 3.52: Posición tarjeta NAZA. (Fuente: Elaboración propia)

Figura 3.53: Icono de Software Naza Lite V2. (Fuente: Elaboración propia)

b) Descargar NAZA-M V2 Asistant Software x.xx de acuerdo a tu plataforma (MAC o Windows)

c) En caso de utilizar Windows se realizaran los siguientes pasos:

1) Encontrar el archivo DJI WIN Driver Installer (Windows)
2) Ejecute el instlador del control DJI WIN

d) Ejecute el programa descargado Naza-M- Lite Assistant Software (NAZA-M %20LITE_Installer_ 1.00.exe)

e) Prepárate con el dron cuando el software te pida que conectes el Naza LED con la PC a través de un cable USB

f) Ejecuta el icono +del programa (ver figura 3.53)

3.21.1. Configuración de Drone

1. Ejecuta el icono del programa (ver figura 3.54)
2. Revise los menús y pestañas y configure los parámetros PID (ver figura 3.55)

Una vez ensamblado el drone se realizan las siguientes conexiones, para el Gimbal, y la controladora de vuelo desarrollada (figura 3.55):

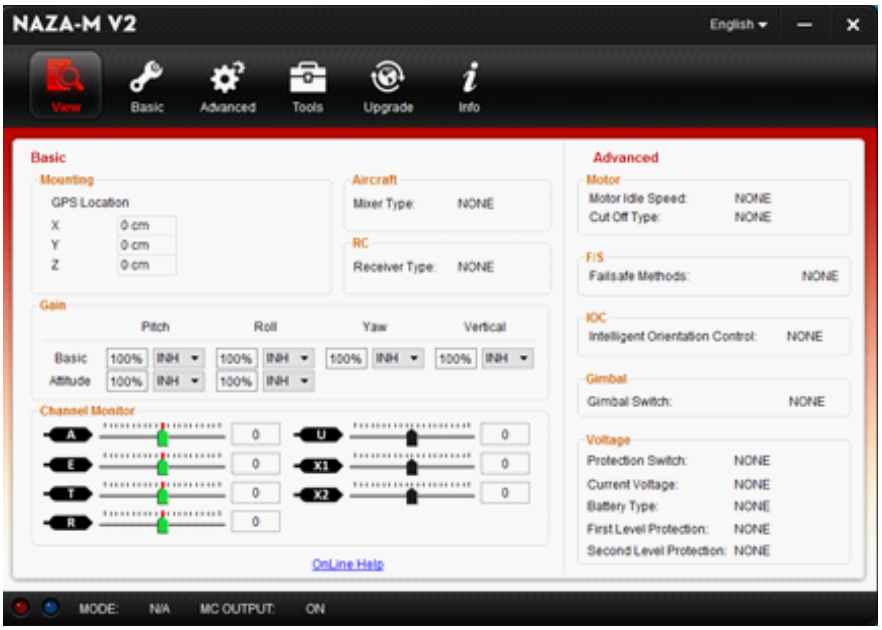

Figura 3.54: Ajuste de controladora de vuelo Naza-M V2. (Fuente: Elaboración propia)

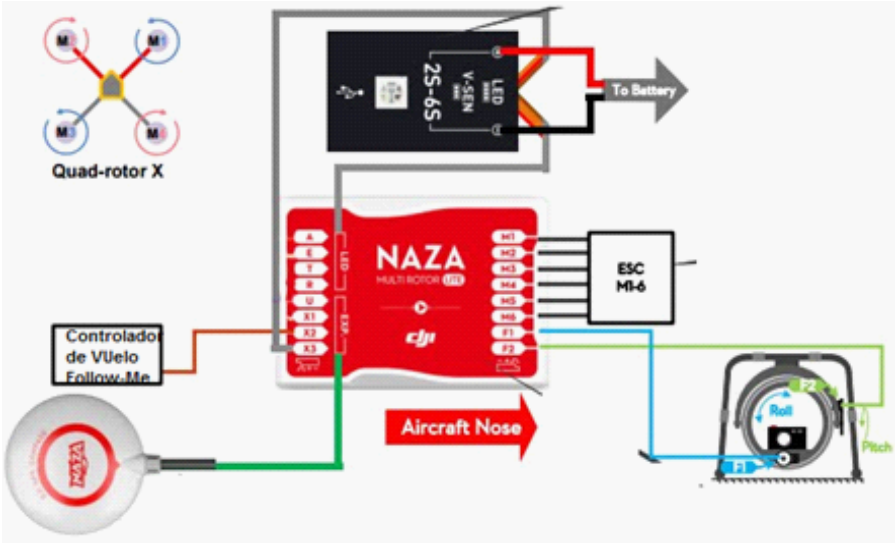

Figura 3.55: Conexión final. (Fuente: Elaboración propia)

Capítulo 4

RESULTADOS

En este capítulo se presentan los resultados del prototipo, las diferentes pruebas realizadas en conjunto con los resultados obtenidos en forma de gráficos y código.

4.1. DISPOSITIVO FINAL – ESTACIÓN TIERRA

En la figura 4.1, se muestra el dispositivo con el cual el usuario controla el UAV, la primer figura se muestra la estación tierra dentro de un case de acrílico de ella sobresalen los joysticks mismos que deben tener libertar de movimiento, el GPS sobresale de igual manera de forma que queda libre de interferencias, el interruptor de selección y el de encendido.

La figura 4.2, muestra el dispositivo internamente, como prototipo se realizó en un baquelita previamente perforada, ya que como trabajo futuro sería el de reducir las dimensiones del prototipo, en ella se pueden apreciar los componentes utilizados como lo son un barómetro, 2 Joystick, un GPS NEO M8N, una pantalla OLED LCD, un microcontrolador STM32F103C y un módulo de comunicación HC-12.

En la figura 4.3, se muestra la parte lateral, la cual cuenta con la batería y su conectar, el cargador de baterías de Baterías LiPo USB, el interruptor de toda la energía y el switch con el cual seleccionaremos el modo de vuelo.

En la figura 4.4, se muestra la estación tierra encendida, mostrando así los mensajes que visualizan en la LCD y los led que dan aviso del estado de los sensores.

4.1.1. DISPOSITIVO FINAL – CONTROLADORA FOLLOW-ME

En la figura 4.5, se muestra el dispositivo con el cual el usuario controla el UAV para después iniciar el modo Follow Me, con lo que el UAV utilizara los sensores como lo son el GPS, barómetro y magnetómetro.

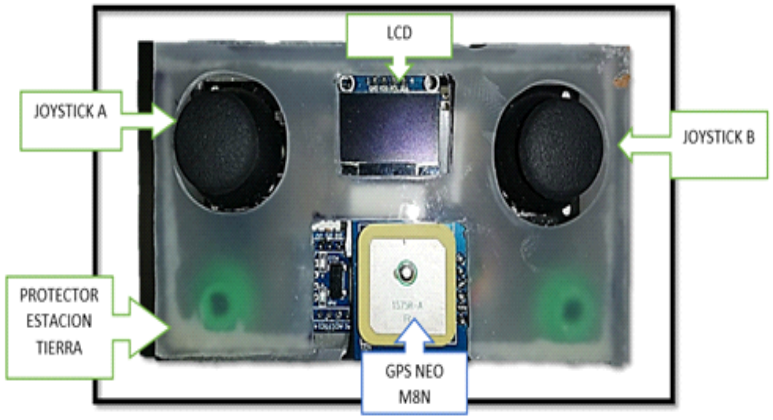

Figura 4.1: Estación tierra. (Fuente: Elaboración propia)

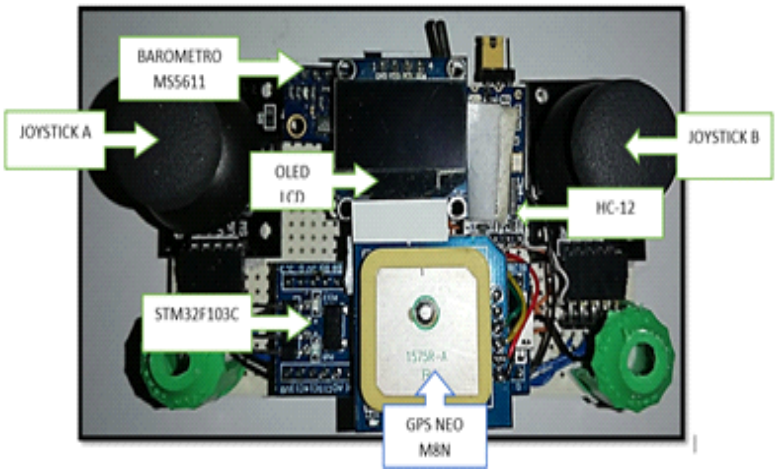

Figura 4.2: Estación tierra: Componentes frontal. (Fuente: Elaboración propia)

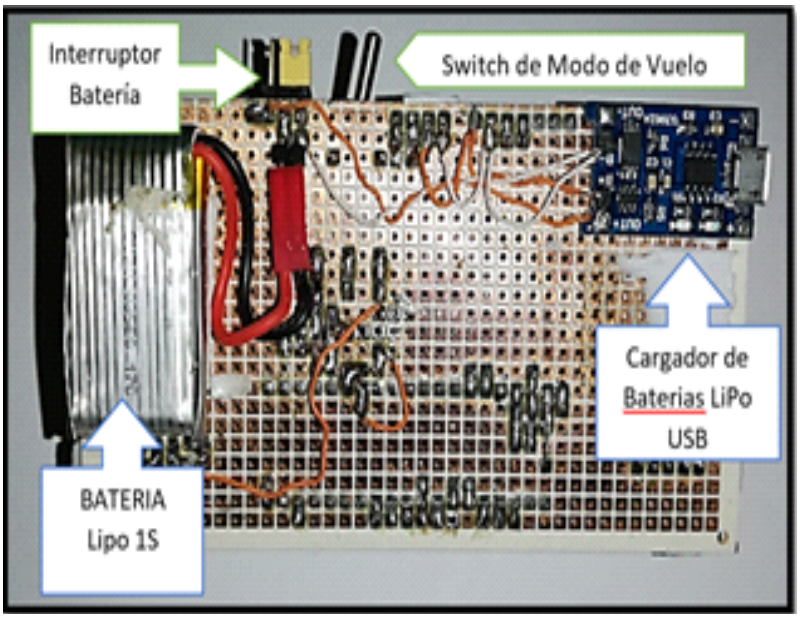

Figura 4.3: Estación tierra: Componentes lateral. (Fuente: Elaboración propia)

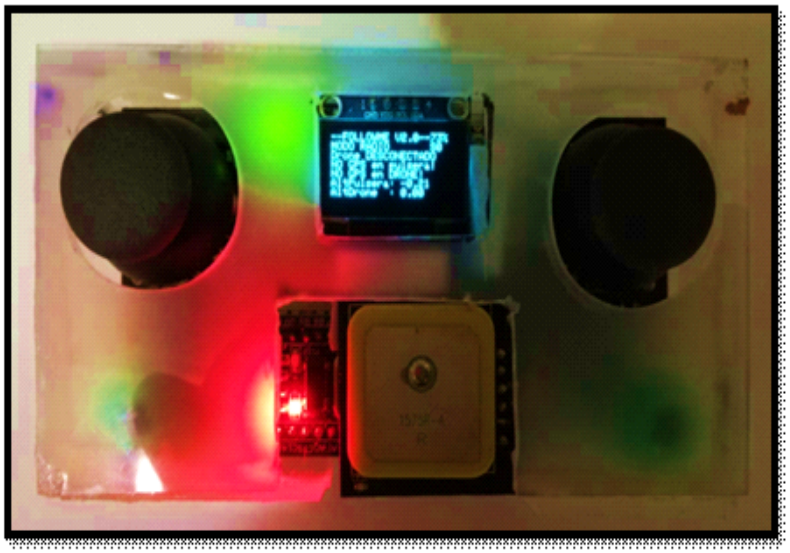

Figura 4.4: Estación tierra encendida. (Fuente: Elaboración propia)

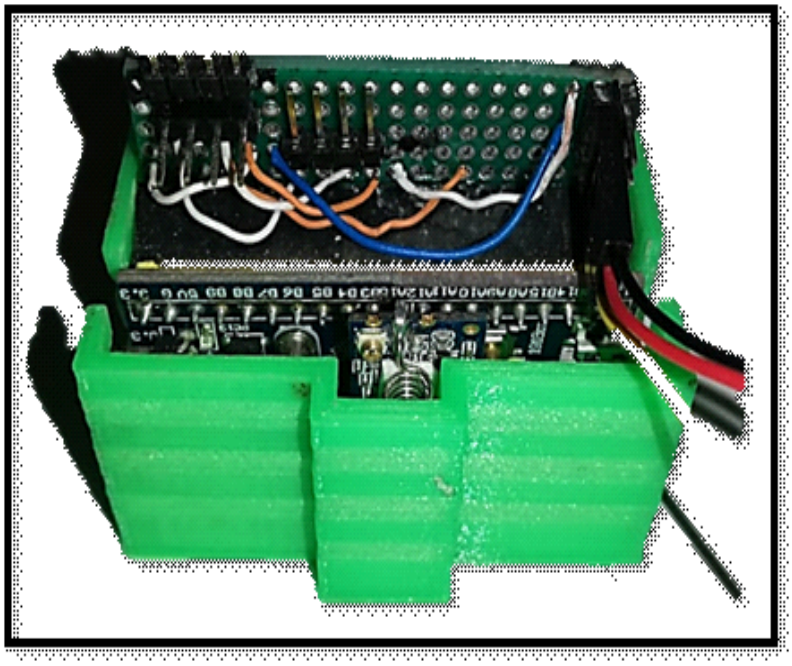

Figura 4.5: Controladora UAV con protector. (Fuente: Elaboración propia)

En la figura 4.6, se muestra los componentes internos de la controlara UAV como los son el módulo de comunicación HC-12, un conector para GPS NAZA, el barómetro, el conector para Radio PPM y el microcontrolador STM32F103C.

4.1.2. VISUALIZACION DE DATOS EN TERMINAL SERIAL

Barómetro

La altura es calculada por el microcontrolador con la siguiente ecuación:

$$altitude = 44330 * (1 - (\frac{p}{p_0})\hat{}(\frac{1}{5{,}255})) \tag{4.1}$$

Calcula la altitud

Los datos son filtrados por medio de un filtro de media ponderada:

$$x = \frac{\sum_{i=1}^{n} X_i}{n} \tag{4.2}$$

Filtro de media ponderada

```
1.  void Actualizar_Barometro(void)// Calculate altitude
2.  { static long Barometrotiempo;
```

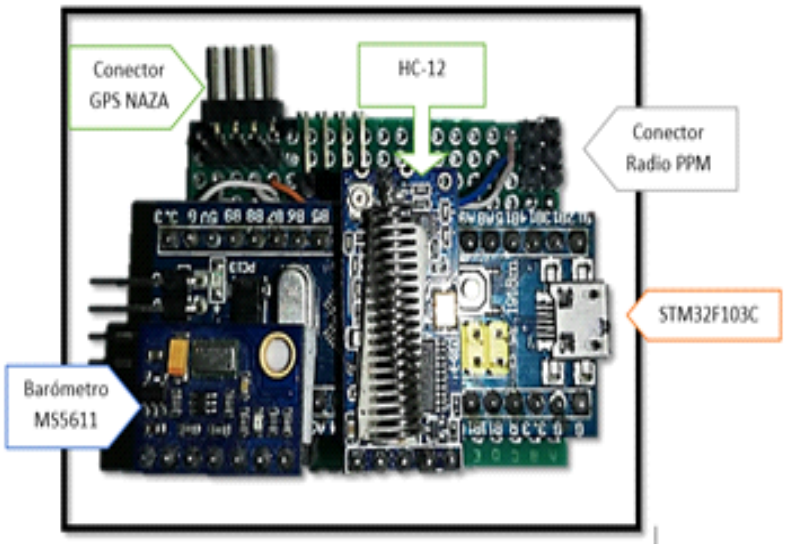

Figura 4.6: Controladora UAV: Componentes. (Fuente: Elaboración propia)

```
3.  if((millis() - Barometrotiempo) > 50){
4.  long realPressure = ms5611.readPressure();
5.  // Calculate altitude
6.  float absoluteAltitude = ms5611.getAltitude(realPressure);
7.  float relativeAltitude = ms5611.getAltitude(realPressure, referencePressure);
8.
9.  //Guardamos el dato en el buffer
10.  for(int b=puntos; b>0; b–){
11.  bufferbaro[b]=bufferbaro[b-1];
12.  }
13.  bufferbaro[0]=relativeAltitude;
14.
15.  mediaAltitudrelativa=0;
16.
17.  for(int l=0; l<puntos; l++){
18.  mediaAltitudrelativa=(bufferbaro[l])+mediaAltitudrelativa;
19.  }
20.  mediaAltitudrelativa = mediaAltitudrelativa/puntos;
21.  AlturaDrone=mediaAltitudrelativa;
22.  Barometrotiempo = millis();
```

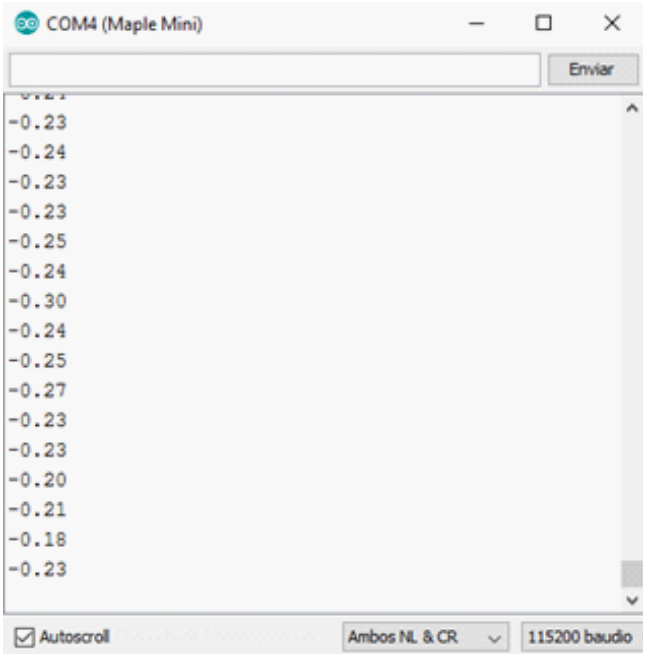

Figura 4.7: Datos seriales. (Fuente: Elaboración propia)

23. }
24. }

Graficamos la altura con filtro y sin filtro como se muestra en la figura 4.7 y 4.8.

La altura que es censada por el barómetro de la estación tierra y del controlador UAV es ingresada por medio de la siguiente ecuación, el resultado entre ellos es ingresado al controlador PID para mantener la distancia con la variable "distanciaFollowme", el setpoint será la altura en la cual el interruptor de modos cambia a modo Follow-me:

1.
AlturaFollowme = AlturaDrone - AlturaPulsera;
2. distanciaFollowme=distanciaentrepuntos(LatitudPulsera,LongitudPulsera,latitudfiltrada,longitu
3.

GPS

Los datos del GPS son obtenidos como los de la figura 4.9:

Estos datos por medio de la librería GPS fueron convertidos a variables como lo son las coordenadas de latitud, longitud, cantidad de satélites detectados (ver figura 4.9 y 4.10).

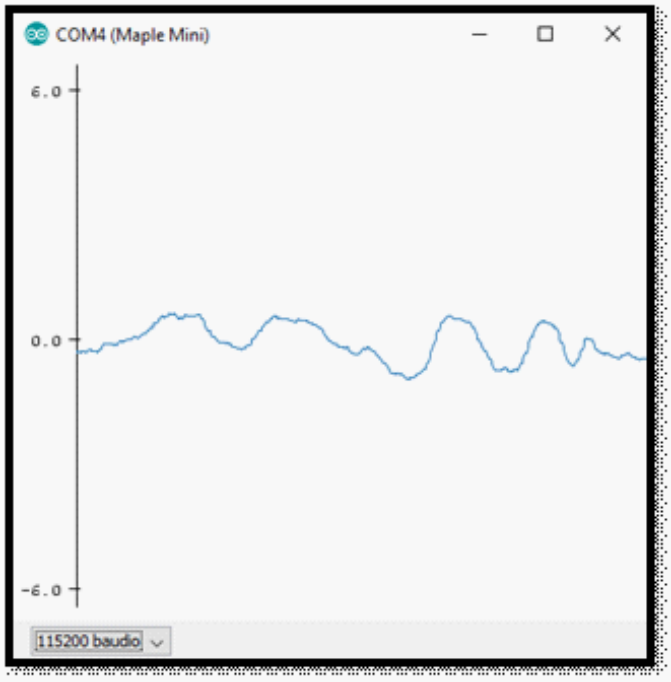

Figura 4.8: Gráfica de altura. (Fuente: Elaboración propia)

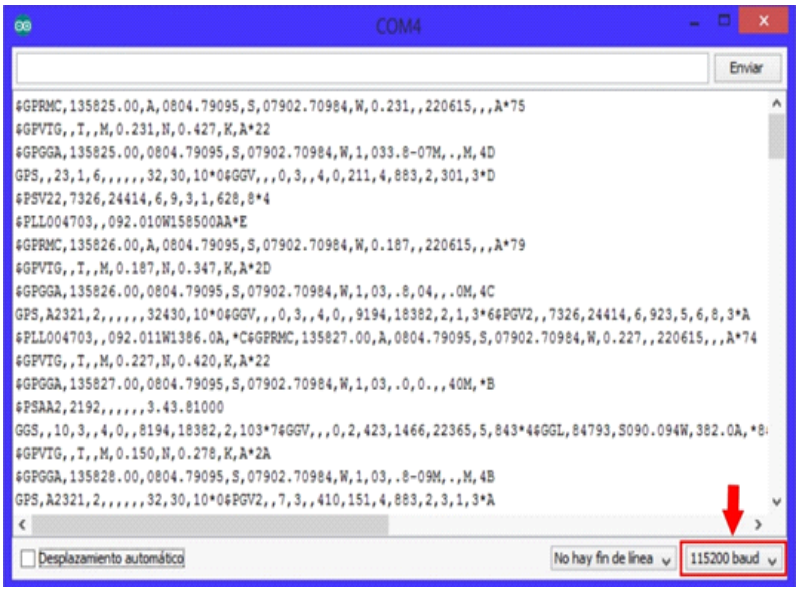

Figura 4.9: Datos GPS RAW. (Fuente: Elaboración propia)

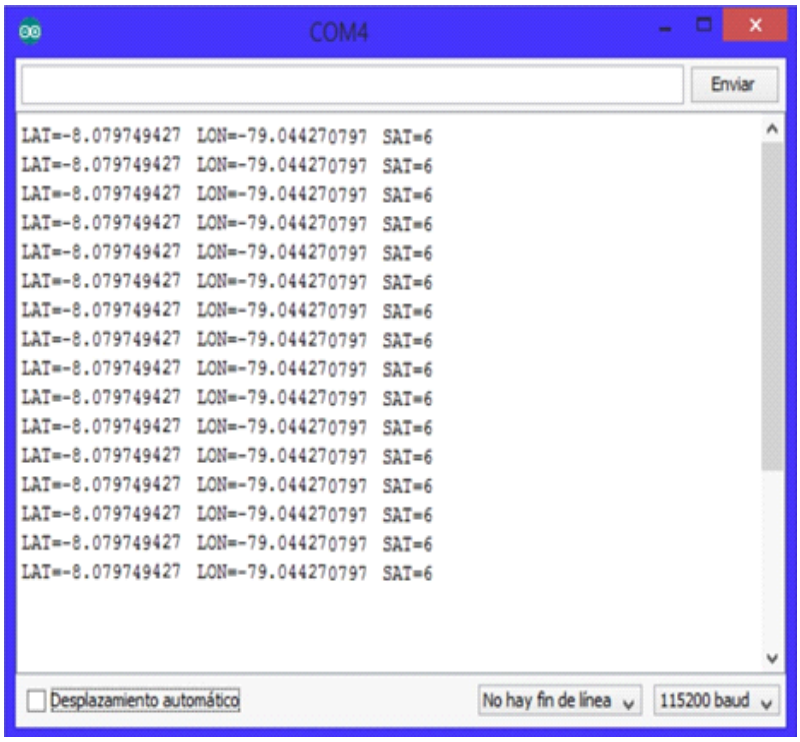

Figura 4.10: Coordenadas. (Fuente: Elaboración propia)

Calculo de distancia entre dos puntos En el cálculo de la distancia entre ambas posiciones debemos contemplar la curvatura terrestre. Es aquí donde entra en escena la Fórmula del Haversine.

$$R = radiodelaTierra \tag{4.3}$$

$$\Delta lat = lat2 - lat1$$

$$\Delta long = long2 - long1$$

$$a = sin^2(\Delta lat/2) + cos(lat1) \cdot cos(lat2) \cdot sin^2(\Delta long/2)$$

$$c = 2 \cdot atan2(\sqrt{a}, \sqrt{(1-a)})$$

$$d = R \cdot c$$

Ecuación 22. Fórmula de Haversine
El valor del radio ecuatorial es de *6372795* metros.

```
1.  double Radiotierra=6372795;
2.
3.  double Phi1=radians(lat1);
4.  double Phi2=radians(lat2);
5.
6.  double deltaPhi=radians(lat2-lat1);
7.  double deltalambda=radians(long2-long1);
8.
9.  double a=sin(deltaPhi/2)*sin(deltaPhi/2)+cos(Phi1)*cos(Phi2)*sin(deltalambda/2)*sin(deltalambda/2
10.
11.   double c=2*atan2(sqrt(a),sqrt(1-a));
12.
13.   double d=Radiotierra*c;
14.   return d;
```

En la figura 4.11 se muestran los datos del terminal serial obtenidos de las coordenadas GPS de la estación tierra y del controlador UAV necesarias para calcular la distancia entre los dispositivos.

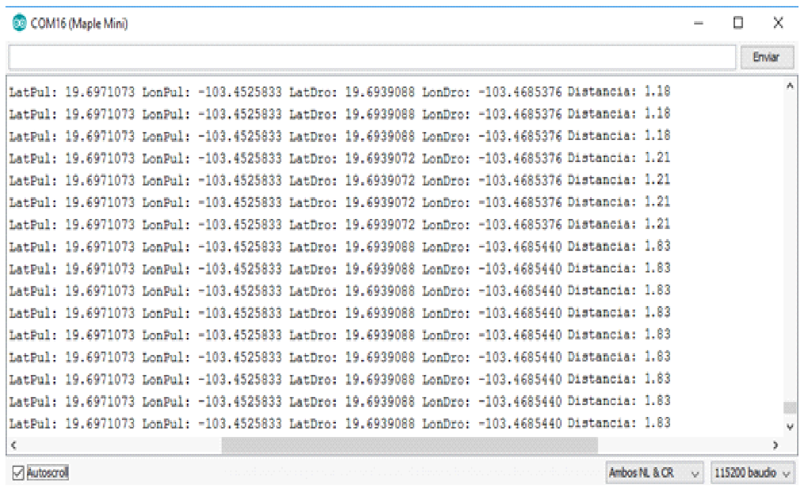

Figura 4.11: Distancia entre dispositivos. (Fuente: Elaboración propia)

Magnetómetro

Para realizar el seguimiento, la plataforma móvil siempre debe apuntar hacia la estación tierra, ajustando el YAW, para ello se calculó mediante las coordenadas entre dos puntos (ver figura 4.12) utilizando la ecuación 4.4.

$$(b1, b2) = (a1 + r \sin\theta, a2 + r \cos\theta) \tag{4.4}$$

Para después saber cuál es el error respecto a la orientación actual del UA, dando como resultado el siguiente código:

1. double Anguloentre(double lat1, double long1, double lat2, double long2)
2. {//https://math.stackexchange.com/questions/1596513/find-the-bearing-angle-between-two-points-in-a-2d-space
3. // returns course in degrees (North=0, West=270) from position 1 to position 2,
4. // both specified as signed decimal-degrees latitude and longitude.
5. // Because Earth is no exact sphere, calculated course may be off by a tiny fraction.
6. // Courtesy of Maarten Lamers
7. double dlon = radians(long23-long1);
8. lat1 = radians(lat1);
9. lat2 = radians(lat2);

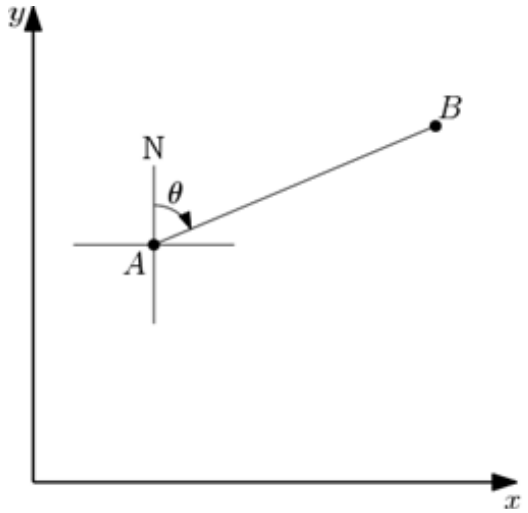

Figura 4.12: Representación de coordenadas. (Fuente: Elaboración propia)

```
10.  double a1 = sin(dlon) * cos(lat2);
11.  double a2 = sin(lat1) * cos(lat2) * cos(dlon);
12.  a2 = cos(lat1) * sin(lat2) - a2;
13.  a2 = atan2(a1, a2);
14.  if (a2 < 0.0)
15.  {
16.  a2 += TWO_PI;
17.  }
18.  return degrees(a2);
19.  }
20.  float errorangulo(float anguloactual,float angulodeseado){
21.  float error=anguloactual-angulodeseado;
22.  if(error>180)error=error-360;
23.  else if(error<-180)error=error+360;
24.  return error;
25.  }
```

Los resultados finalmente son obtenidos como una variable que ingresara al PID para su control, los datos obtenidos son los siguientes (ver figura 4.13):

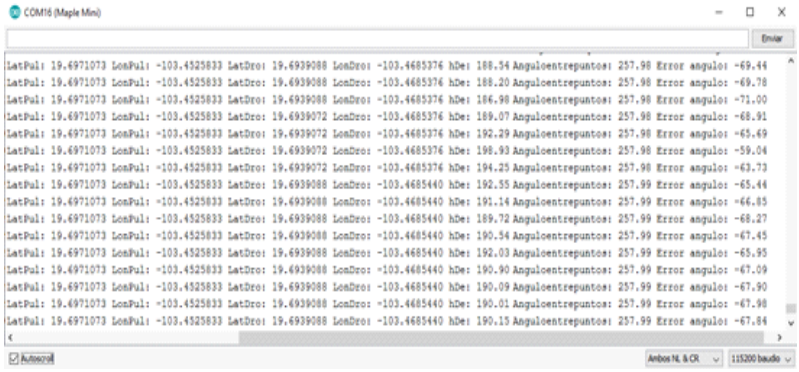

Figura 4.13: Ajustes de YAW. (Fuente: Elaboración propia)

4.2. VISUALIZACIÓN DE INTERFAZ LABVIEW FOLLOW-ME

Mediante el Software de LabVIEW el cual nos ofrece un entorno de programación grafico como lo hemos visto con anterioridad en los fundamentos teóricos y la forma en la que trabajan cada uno de ellos. Se desarrolló un programa cuya función fuera la de mostrar los valores en tiempo real de los sensores como lo son la altura, angulo del magnetómetro, valores del radio (ver figura 4.14). Cuya función fue la de comprobar que el sistema estuviera funcionando correctamente y así de forma sencilla realizar calibración y verificar conexiones.

El Diagrama a bloques fue el siguiente (ver figura 4.15):

4.3. PRUEBAS PLATAFORMA MÓVIL TERRESTRE ROBOTKIDS

Las siguientes figuras muestran una de las pruebas del prototipo RobotKids, el robot móvil terrestre fue utilizado hasta que se comprobó que el mismo lograba hacer un seguimiento de la estación tierra.

En la figura 4.16 se encuentra esperando que los GPS encontraran la posición actual de la estación tierra y del robot terrestre.

En la figura4.17, el robot terrestre se orienta en YAW hacia la estación tierra.

En la figura 4.18, el robot terrestre realiza el seguimiento de la estación tierra en movimiento.

En la figura 4.19 el robot terrestre realiza el seguimiento de la estación tierra desplazando y girando, en dirección a la estación tierra.

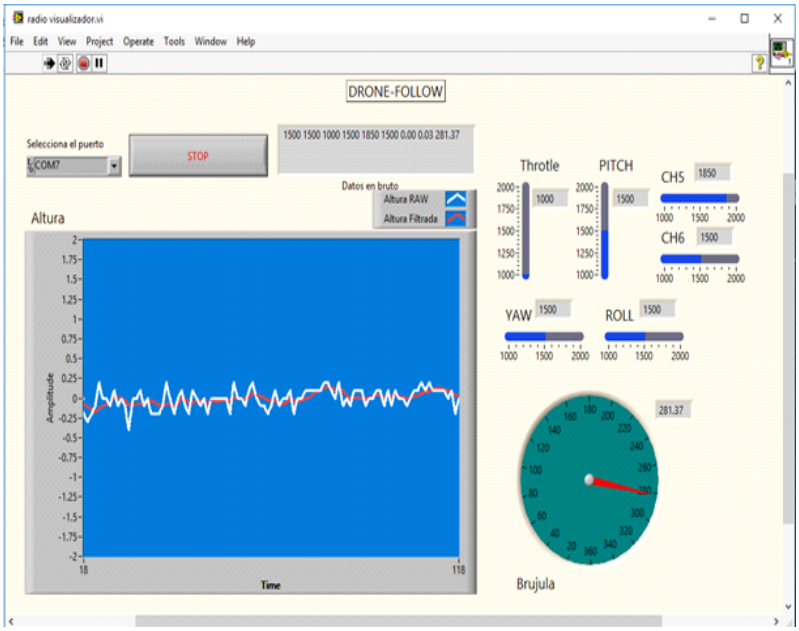

Figura 4.14: Programa DRONE-FOLLOW. (Fuente: Elaboración propia)

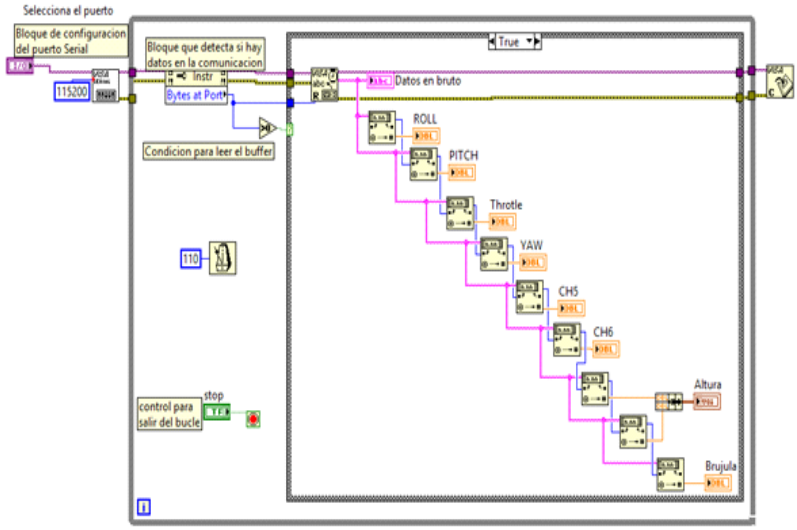

Figura 4.15: Diagrama de bloques de software DRONE-FOLLOWME. (Fuente: Elaboración propia)

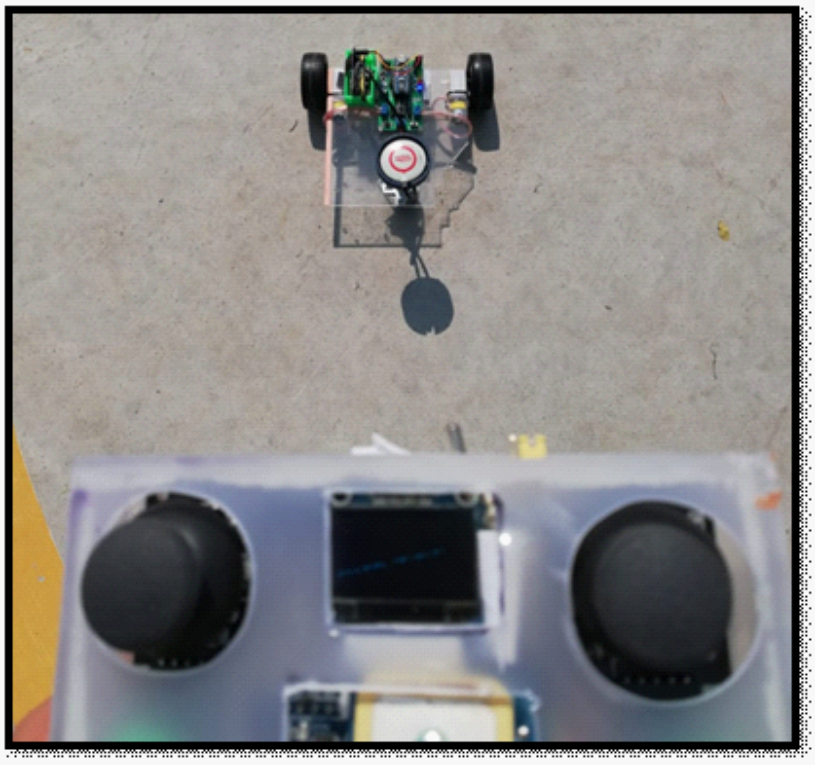

Figura 4.16: Prueba A. (Fuente: Elaboración propia)

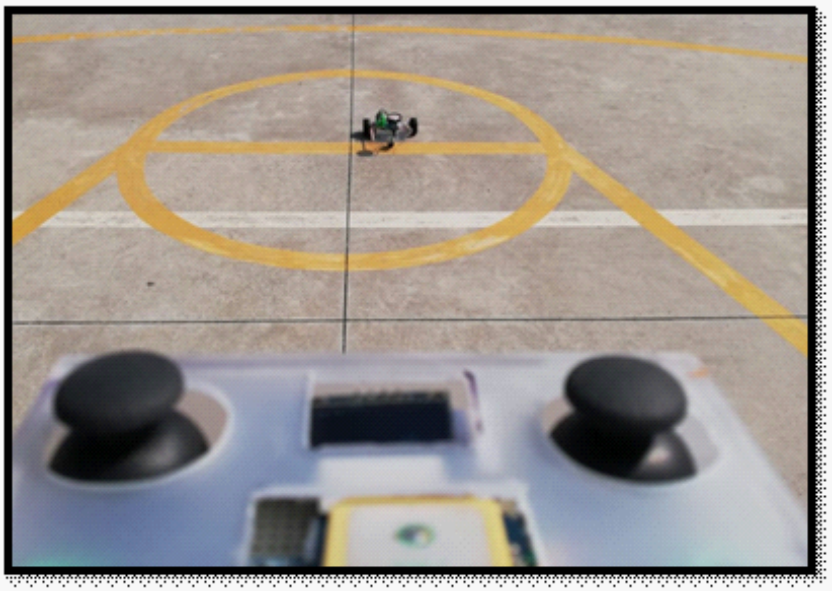

Figura 4.17: Prueba B. (Fuente: Elaboración propia)

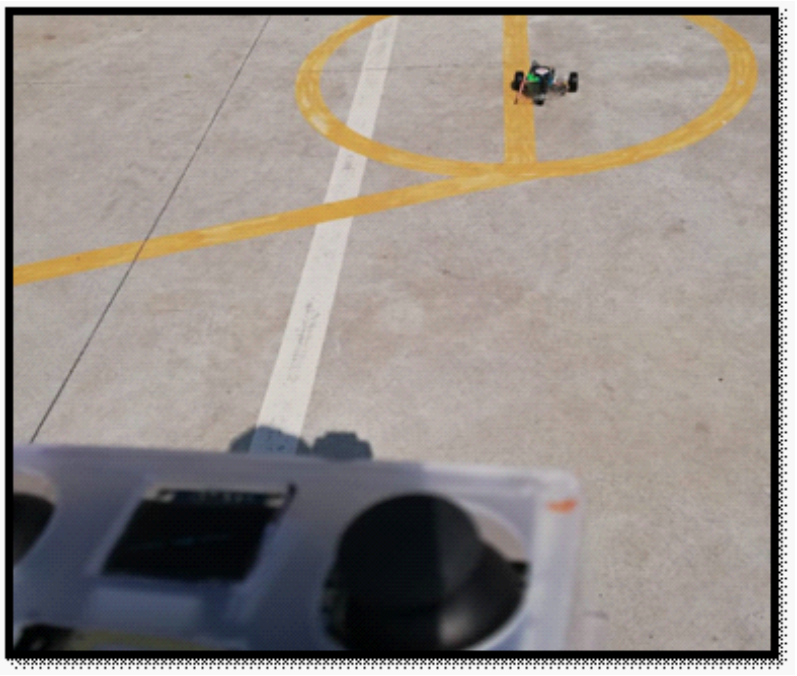

Figura 4.18: Prueba C. (Fuente: Elaboración propia)

Figura 4.19: Prueba D. (Fuente: Elaboración propia)

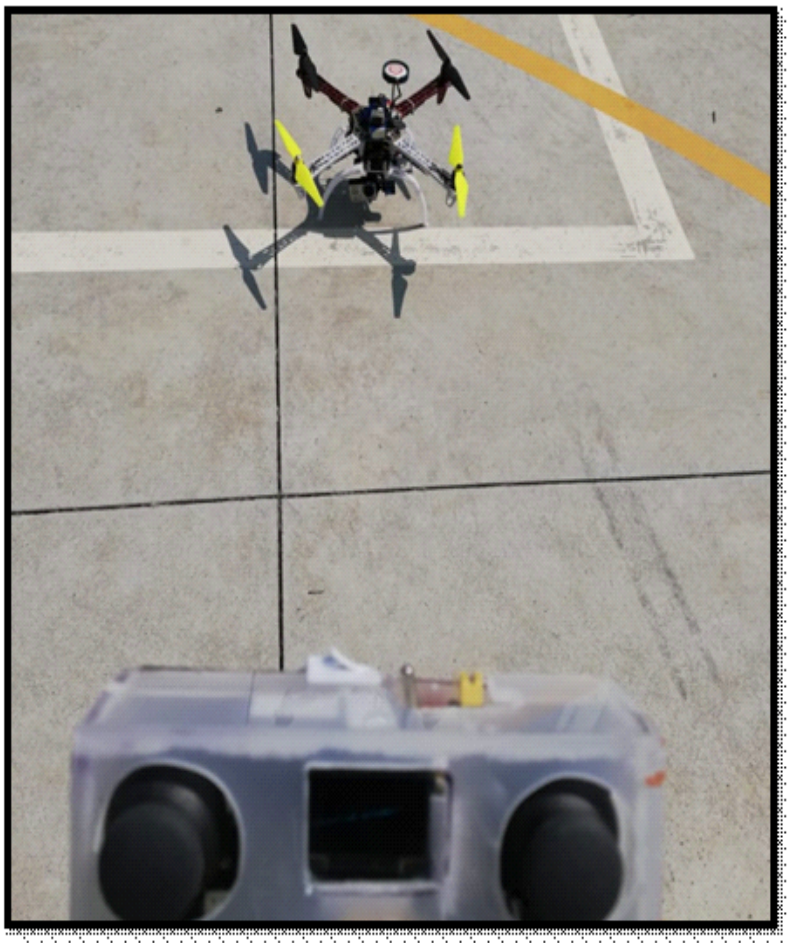

Figura 4.20: Prueba E. (Fuente: Elaboración propia)

4.4. PRUEBAS PLATAFORMA MÓVIL AÉREA UAV

Las siguiente etapas muestran una prueba que se realizó con el UAV ya con el sistema instalado, para el seguimiento, esta etapa se dejó al final ya que era la parte más delicada por el peligro de un descontrol del vehiculó.

En la figura 4.20 se encuentra en espera a que los dos GPS encontraran la ubicación, para después armar el UAV.

En la figura 4.21 se muestra ya el UAV volando se llevó a esa posición por medio de la estación tierra.

En la figura 4.22 se muestra el UAV realizando un seguimiento de la estación tierra solo girando en YAW.

En la figura 4.23 se muestra el UAV en seguimiento de la estación tierra.

Figura 4.21: Prueba F. (Fuente: Elaboración propia)

Figura 4.22: Prueba G. (Fuente: Elaboración propia)

Figura 4.23: Prueba H. (Fuente: Elaboración propia)

Capítulo 5

CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

En esta investigación se logró cumplir parte de los objetivos planteados inicialmente. Particularmente, se considera que:

- Se comprendió el funcionamiento general de UAVs y de manera específica tanto las leyes que rigen el comportamiento de los multicopteros, así como diferentes controladoras de vuelo comerciales que existen para desarrollar UAVs.
- Se encontraron las mejores alternativas en sensores como lo son magnetómetros, brújulas, GPS, giroscopio, acelerómetros y barómetros, así como probar el funcionamiento de diferente hardware y software de sistemas embebidos.
- Realizamos correctamente censado de altura, campo magnético, señales analógicas, latitud y longitud y al final filtrar esas señales, así como diferentes alternativas para comunicación inalámbrica de dispositivos, escritura de señales PWM.
- Logramos que la IDE de Arduino, trabajara con microcontroladores de 32bits de STM lo que muestra alternativas a los microcontroladores comunes de 8 bits.
- Implementamos un controlador para altura, dirección y posición geográfica de un UAV y se desarrolló un software para visualizar sus variables.

Esta investigación es enfocada al desarrollo de software y experimentación de diverso hardware y software que apoye a otros en la solución de sus problemas, gracias a la

investigación desarrollada al mercado actual encontrando muchas alternativas de componentes muy económicos y capaces de procesar gran cantidad de información, no recurriendo a lo más básico como lo son microcontroladores de Microchip o de Atmel. Si no utilizando aquellos que emplean grandes empresas como lo son DJI o PIXHAWK o sensores muy potentes y como trabajar con ellos.

Al final concluimos con las pruebas y resultados que se ha cumplido parte del objetivo principal de esta investigación, aunque se sigue necesitando muchas pruebas con diferentes dispositivos UAV, para crear código o controladores que puedan ser más universales, pero dado que los recursos para la investigación eran de procedencia personal solo estuvimos reducidos a un robot terrestre y un robot aéreo.

5.2. RECOMENDACIONES

Para continuar la mejora del proyecto de esta investigación una mejora sería la de implementar una librería de protocolo de comunicación que apoye en el envío y recepción de la variables de latitud y longitud ya que la transmisión serial común solo permite enviar variables de hasta seis decimales sin tener un error.

Desarrollar PCBs de los sistemas mínimos del hardware para reducir dimensiones y peso aunque ello pediría tener un sistema final, que ya no permitiera la modificación de hardware.

También la implementación de cámaras para realizar también el seguimiento mediante el procesamiento de imágenes, lo que vendría a mejorar mucho el sistema ya que el GPS tiene una frecuencia de actualización baja y requiere que de ciertas condiciones ambientales.

Algo que se llegó a necesitar mucho fue la de la optimización de código, abarcando gran espacio en la memoria del microcontrolador por la tanto seria continuar la optimización o cambiar por un microcontrolador que tenga más memoria para poder implementar más dispositivos y funciones.

Y por último la inclusión de controladores que sean capaces de adaptarse al dispositivo ya sea implementando un PID FUZZY, una red neuronal o algún controlador desarrollado para la aplicación.

Bibliografía

[1] A, M. F. (2004). El Bus I2C. (U. d. Cordoba, Ed.) España: Departamento de Computadores E. T Electronica.

[2] Ag Electronica. (Junio de 2017). Ag. Electronica. Recuperado el Enero de 2018, de Barometric Pressure Sensor-BMP085 Breakout SparkFun: http://www.agspecinfo.com/pdfs/S/SEN11282.PDF

[3] Ag Electronics. (junio de 2017). Ag Electronics. Recuperado el Enero de 2018, de Barometric Pressure Sensor-BMP0/85 Breakout-SparkFun Electronic: http://www.agspecinfo.com/pdfs/S/SEN11282.PDF

[4] AliExpress: Electronics Component Whosale. (2018). AliExpress: Electronics Component Whosale. Recuperado el Enero de 2018, de HC-12 wireless serial port module: https://es.aliexpress.com/item/433Mhz-HC-12-SI4463-Wireless-Serial-Port-RF-Module-HC-12-1000M-for-microcontroller/1903243296.html

[5] Arnéstegui, M. M. (2014). Apuntes de control PID. La Paz, Bolivia: Universidad Mayor de San Andres.

[6] Brown, G. (2006). Discovering the STM32 Microcontroller. Desconocido.

[7] Bueno, D. (2011). Motor Eléctrico Brushless: Funcionaientos y caracteristicas. México, D.F.

[8] Chuk, D. (2012). Los sistemas de primer orden y los Controladores PID.

[9] CIAA POSIX. (2009). Ingenieria Satelital. Recuperado el Enero de 2018, de Mision Sae: https://2mp.conae.gov.ar/attachments/article/1313/SAE-MAN-0008-B %20- %20Manual %20POSIX.pdf

[10] Desconocido. (2016). Control PID clásico. Recuperado el Enero de 2018, de Control PID clásico: http://www2.elo.utfsm.cl/~elo270/informacion/CLASES/12 %20CONTROL %20PID.pdf

[11] Desconocido. (2016). Wireless Serial Port Comunication Module. User Maual Version 2.3B. Recuperado el Enero de 2018/, de Wireless Serial Port Comunication Module. User Maual Version 2.3B: http://avrproject.ru/112/rf_hc12/2016-01-14_122335_HC-12_v2.3B.pdf

[12] Desconocido. (2017). u-Blox. Recuperado el Enero de 2018, de NEO-6: u-blox GPS Modules. Data Sheet: https://www.u-blox.com/sites/default/files/products/documents/NEO-6_DataSheet_(GPS.G6-HW-09005).pdf

[13] W., V. C. (Julio 2016). El Impacto de la tecnología dron en la seguridad provada en Colombia. . Especializacion en administraciónn de la seguridad. Universidad Militar Nueva Granada, 3-12.

[14] UAV DRONE. (2017). UAV DRONE. Recuperado el Enero de 2018, de UAV DRONE: https://uavdrone.es/follow-me-drone

[15] Desconocido. (s.f.). Interfase Gráfica para el medidor de nivel: La comunicación serial. Recuperado el Enero de 2018, de Interfase Gráfica para el medidor de nivel: La comunicación serial: http://catarina.udlap.mx/u_dl_a/tales/documentos/lem/morales_h_oe/capitulo3.pdf

[16] Desconocido. (s.f.). Naza para Multirrotores: Manual del usuario. Obtenido de Naza para Multirrotores: Manual del usuario.

[17] Desconocido. (s.f.). Tutorial de LabVIEW. Recuperado el Enero de 2018, de Tutorial de LabVIEW: http://www.esi2.us.es/~asun/LCPC06/TutorialLabview.pdf

[18] EPA: Electronica Practica Aplicada. (2017). Electronica Practica Aplicada. Recuperado el Enero de 2018, de Magnetómetro HMC5883L: https://www.diarioelectronicohoy.com/blog/magnetometro-hmc5883l

[19] FPVMax. (2016). FPVMax. Recuperado el Enero de 2018, de Variador electrónico (ESC): Qué es y cómo funciona: http://fpvmax.com/2016/12/21/variador-electronico-esc-funciona/

[20] Geek Factory. (2017). Geek Factory. Recuperado el Enero de 2018, de Geek Factory: https://www.geekfactory.mx/tienda/sensores/sensor-de-presion-atmosferica-bosh-bmp085/

[21] Geekbot Electronics. (2018). Geekbot Electronics. Recuperado el Enero de 2018, de HMC5883L Magnetómetro de 3 ejes: http://www.geekbotelectronics.com/producto/hmc5883l-magnetometro-de-3-ejes/

[22] Giménez, R. T., & Ros, B. M. (2010). Sistema de posicionamiento Global (GPS). Gravitación y Astrofísica .

[23] Heliboss. (2018). Heliboss. Recuperado el Enero de 2018, de DJI NAZA V2 + GPS: https://heliboss.com/dji-naza-v2-gps.html

[24] Hetpro. (2016). Hetpro. Recuperado el Enero de 2018, de Joystick analógico dos ejes con boton: https://hetpro-store.com/TUTORIALES/joystick-analogico-programado-con-arduino/

[25] Hetpro. (2017). Hetpro. Recuperado el Enero de 2018, de MPU 6050 Arduino, acelerómetro y giroscopio: https://hetpro-store.com/TUTORIALES/modulo-acelerometro-y-giroscopio-mpu6050-i2c-twi/

[26] Hetpro. (Junio de 2017). Hetpro. Recuperado el Enero de 2018, de Puerto serial-protocolo y su teoría: https://hetpro-store.com/TUTORIALES/puerto-serial/

[27] HobbyKing. (s.f.). HobbyKing. Recuperado el Enero de 2018, de Tarot T-2D GoPro 3 sin escobillas cámara del cardán y el controlador ZYX22: https://hobbyking.com/es_es/tarot-t-2d-v2-gopro-3-brushless-camera-gimbal-and-zyx22-controller.html

[28] HolyBro. (2018). HolyBro. Recuperado el Enero de 2018, de Ublox NEO-M8N GPS: http://www.holybro.com/product/21

[29] Llamas, L. (2015). Arduino. Recuperado el Enero de 2018, de STM32F103, El competidor ARM de Arduino de bajo coste: https://www.luisllamas.es/stm32f103-el-competidor-arm-de-arduino-de-bajo-coste/

[30] Master Ingenieros. (s.f.). Motor Brushless (sin escobillas) caracteristicas fundamentales.

[31] Mazzone, V. (2002). Controladores PID. Universidad NAcional de Quilmes.

[32] Montenegro, R. (2014). PANAMAHITEK. Recuperado el Enero de 2018, de Joystick en Arduino: http://panamahitek.com/joysticks-en-arduino/

[33] Motores Eléctricos. (s.f.). Motores Eléctricos. Recuperado el Enero de 2018, de Motores Brushless: http://motores.nichese.com/brushless.htm

[34] Moyano, J. E. (2008). Arduino PID- Guía de uso de la librería. Recuperado el Enero de 2018, de Arduino PID-Guía de uso de la libreria: http://brettbeauregard.com/blog/wp-content/uploads/2012/07/Gu%C3%ADa-de-uso-PID-para-Arduino.pdf

[35] N, P. L. (2006). Practicas del uso del Bus I2C para PIC 16F876 de Microchipo. Universit Rovira Virgili. Departament d' Engineryria Electronica Electrica i Automática , 9-49.

[36] National Instrument. (2006). National Instrument. Recuperado el Enero de 2018, de Comunicación Serial: Coneptos Generales: http://digital.ni.com/public.nsf/allkb/039001258CEF8FB686256E0F005888D1

[37] National Instrument. (2014). National Instrument. Recuperado el Enero de 2018, de Entorno NI LabVIEW: http://www.ni.com/academic/students/learnlabview/esa/environment.htm

[38] National Instruments. (2017). Los 4 principales beneficios de LabVIEW.

[39] Naylamp Mechatronics. (2016). Naylamp Mechatronics. Recuperado el Enero de 2018, de Tutorial Módulo GPS con Arduino: https://naylampmechatronics.com/blog/18_Tutorial-M%C3%B3dulo-GPS-con-Arduino.html

[40] Naylamp Mechatronics. (2017). Naylamp Mechatronics. Recuperado el Enero de 2018, de Tutorial MPU6050, acelerómetro y giroscopio: https://naylampmechatronics.com/blog/45_Tutorial-MPU6050-Acelerómetro-y-Giroscopio.html

[41] R, V. P. (2013). Sistema de detección de movimientos basados en sensores incerciales integrados. (C. d. tecnológica, Ed.) México, D.F.: Instituto Politecnico Nacional .

[42] S, G. G. (2013). Diseño y Construcción de magnetómetro triaxial para análisis y experimentación de aislamientos magnéticos. Departamento de Ingenieria Mecánica, 19-32.

[43] Talos Electronics. (2018). Talos Electronics. Recuperado el Enero de 2018, de Pantalla 128x64 LC OLED de 0.96.ªzul con L2C: https://www.taloselectronics.com/products/pantalla-128x64-lcd-oled-de-0-96-azul-con-i2c-4-pines.

[44] Techmake Electronics. (2018). Techmake Electronics. Recuperado el Enero de 2018, de Magnetómetro de tres ejes HMC5883L SparkFun: http://www.techmake.com/00262.html

[45] Torres, O. H., & Estrada, M. R. (2014). U-blox. Recuperado el Enero de 2018, de NEO-6M modulo GPS con MATLAB por USB: https://hetpro-store.com/TUTORIALES/gps-ublox-neo-6m-modulo-con-matlab/

[46] Trastejant. (2017). Trastenjant. Recuperado el Enero de 2018/, de Barómetro BMP085: CAtegoría Arduino: http://www.trastejant.com/tutoriales/barometro-bmp085/

[47] UAV DRONE. (2017). UAV DRONE. Recuperado el Enero de 2018, de UAV DRONE: https://uavdrone.es/follow-me-drone/

[48] U-blox. (2018). U-blox. Recuperado el Enero de 2018/, de NEO-M8 series: Versatile u-blox M8 GNSS modules: https://www.u-blox.com/en/product/neo-m8-series

[49] W, T. (2003). Sistemas de Comunciaciones Eléctronicas. México, D.F.: Pearson Eduación.

[50] W., V. C. (Julio 2016). El Impacto de la tecnología dron en la seguridad provada en Colombia. . Especializacion en administraciónn de la seguridad. Universidad Militar Nueva Granada, 3-12

[51] Yameli, A., Gijón, Y., Neftaly, G., & Herrera, J. (15 de 09 de 2017). ResearchGate. Recuperado el 01 de 2018, de Drones alternativa para la agricultura: https://www.researchgate.net/publication/319734521_DRONES_ALTERNATIVA_PARA_LA_AGRICULTURA_DRONS_AGRICULTURE_CHOICE

Apéndice A

ANEXOS

En la figura A.1 y A.2 se muestra con mas detalle el diagrama de pines de la tarjeta STM32F103.

En las figuras A.4, A.5 y A.6 se muestra el UAV con la tarjeta de control asistido durante las primeras pruebas de funcionamiento.

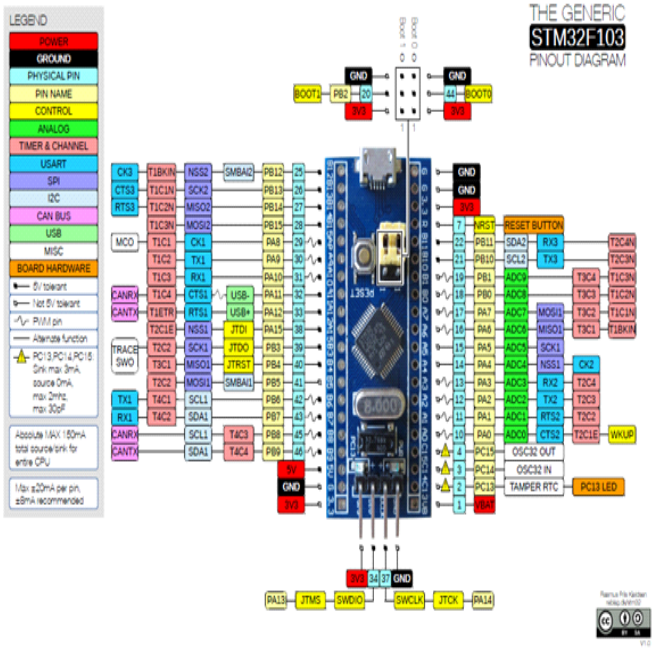

Figura A.1: PINOUT DIAGRAM STM32F103 A (Fuente: http://coytbarringer.com/programming-stm32f103-blue-pill- using-usb-bootloader-platformio/)

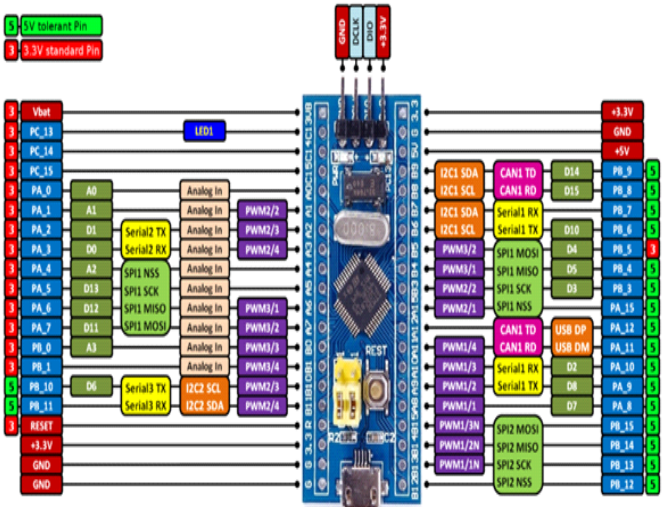

Figura A.2: PINOUT DIAGRAM STM32F103 B (Fuente: https://wiki.stm32duino.com/index.php?title=Blue_Pill)

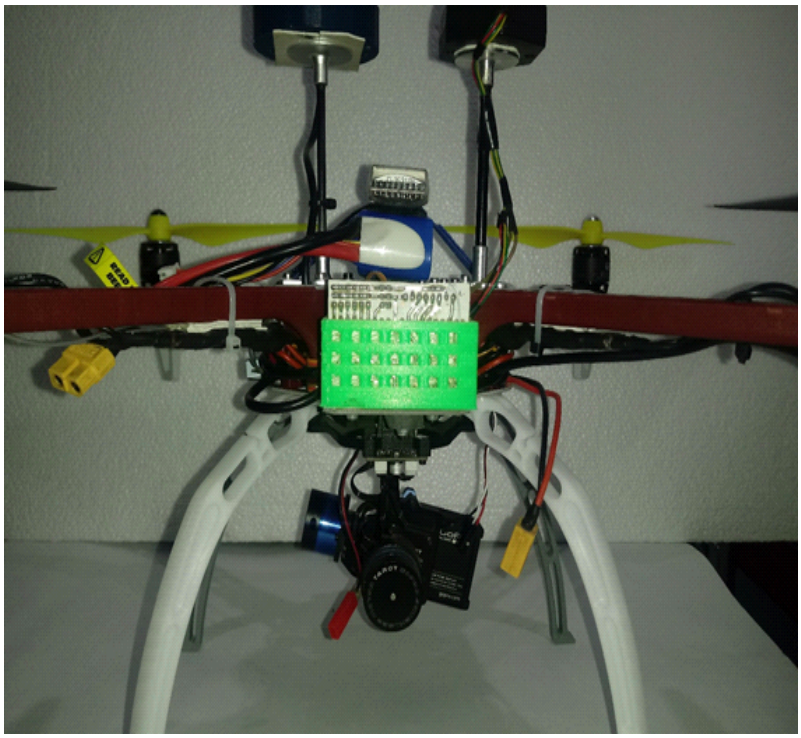

Figura A.3: UAV primer versión. (Fuente: Elaboración propia)

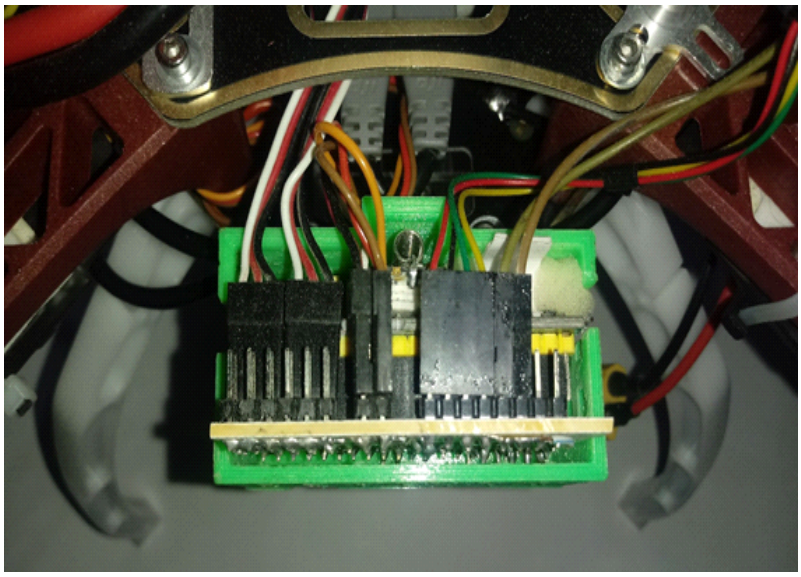

Figura A.4: Controladora UAV primer versión. (Fuente: Elaboración propia)

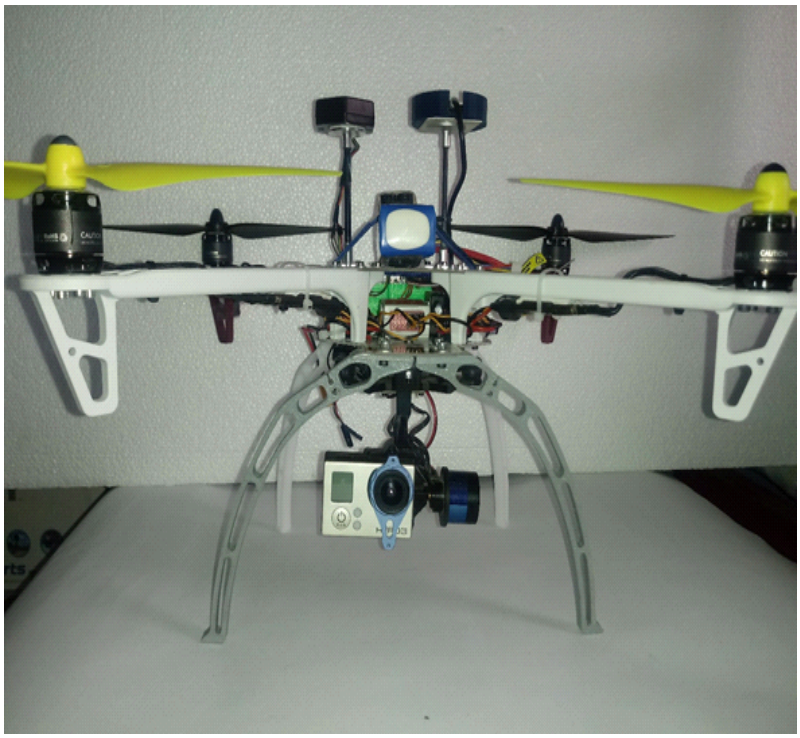

Figura A.5: UAV primeras pruebas. (Fuente: Elaboración propia)

Figura A.6: UAV en pruebas. (Fuente: Elaboración propia)