**FACULTAD DE ARQUITECTURA E INGENIERÍAS**

**Trabajo de fin de Carrera titulado:**

# EVALUACIÓN DEL DISEÑO DE UN DRON DE CUATRO EJES CONTROLADO POR GESTOS MANUALES PARA SU USO PEDAGÓGICO MEDIANTE LA APLICACIÓN DE METODOLOGÍAS CUALITATIVAS


**Realizado por:**

Diego Andrés Ponce Feijo

**Director del proyecto:**

MSc. Jaime Vinicio Molina Osejos


**Como requisito para la obtención del título de:**

**INGENIERO EN MECATRÓNICA**

**QUITO, 24 de septiembre del 2024**

**Declaración Juramentada**

Yo, Diego Andrés Ponce Feijo, con cédula de identidad 1725611212, declaro bajo juramento que el trabajo aquí desarrollado es de mi autoría, que no ha sido previamente presentado para ningún grado a calificación profesional y que se ha consultado las referencias bibliográficas que se incluyen en este documento.

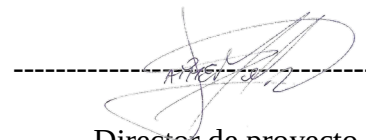
A través de la presente declaración, cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la UNIVERSIDAD INTERNACIONAL SEK, según lo establecido por la Ley de Propiedad Intelectual, por su reglamento y por la normativa institucional vigente

DIEGO ANDRÉS PONCE FEIJO

C.I.: 1725611212

## DECLARACIÓN DEL DIRECTOR DE TESIS

Declaro haber dirigido este trabajo a través de reuniones periódicas con el estudiante, orientando sus conocimientos y competencias para un eficiente desarrollo del tema escogido y dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación.



---

Director de proyecto

MSc. Jaime Vinicio Molina Osejos

LOS PROFESORES INFORMANTES:

MSc. Diana Belen Peralta Zurita

MSc. María Gabriela Mancheno Falconi

Después de revisar el trabajo presentado lo han calificado como apto para su defensa oral ante el tribunal examinador.

_____ _____

MSc. Diana Belen Peralta Zurita MSc. María Gabriela Mancheno Falconi

Quito, 24 de septiembre del 2024

# DECLARACIÓN DE AUTORÍA DEL ESTUDIANTE

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

DIEGO ANDRÉS PONCE FEIJO

C.I.: 1725611212

## Índice de contenidos

## Índice de figuras

## Índice de diagramas

## Índice de tablas

## Índice de anexos

## Dedicatoria

Dedico esta tesis en primer lugar, a mis padres Pedro Ponce y Mayra Feijo que me han brindado su amor y apoyo incondicional durante toda mi vida. Quiero mencionar también a mis abuelitos Hugo Feijo, Guadalupe Gallo, Segundo Ponce, Estela Delgado y hermanos Gabriel y Paula Ponce quienes han sido parte importante en el desarrollo de este trabajo. Además, quiero agradecer a mi futura esposa Sofía Ortiz por motivarme y darme palabras de aliento en los más momentos difíciles. A mis amigos Mateo, Danilo, Emilio, Melany que me han brindado su ayuda en todo momento además de las risas y buenos momentos durante la carrera.

**Agradecimientos**

Quiero expresar mi más profundo agradecimiento a mi tutor de tesis y profesores que en el trayecto de mi carrera me han brindado su guía, paciencia y sabiduría; su apoyo incondicional y sus valiosas recomendaciones fueron fundamentales para la realización de este trabajo. Mi gratitud a la Universidad Internacional SEK, por el apoyo recibido a través del uso de herramientas e instalaciones para llevar a cabo mi investigación.

Además, quiero agradecer a mi familia en especial a mis padres, por los valores inculcados a lo largo de mi vida, lo cual me ayudo a lograr los objetivos académicos actuales. A mi pareja por su amor, comprensión y constante apoyo emocional, su aliento y confianza en mí me motivaron a seguir adelante, incluso en los momentos más difíciles. Un sincero agradecimiento a mis amigos de la carrera, por su colaboración, apoyo y por compartir este viaje académico conmigo, su amistad y valiosas ideas fueron esenciales en muchos momentos de este proyecto. Adicionalmente, quiero agradecer a Don Edu por sus consejos y por ser un mentor en mi vida académica, su empatía me ayudo a resolver inconvenientes del día a día.

Gracias a todos los que, de una u otra manera, contribuyeron al logro de este proyecto.

## Abreviaturas

| | |
|---|---|
| Ah | Amperio hora |
| CA | Corriente Alterna |
| CC | Corriente Continua |
| DGAC | Dirección General de Aeronáutica Civil |
| DOE | Diseño de Experimento (siglas en inglés) |
| ESC | Control Electrónico de Velocidad (siglas en inglés) |
| GPS | Sistema de Posicionamiento Global (siglas en inglés) |
| I2C | Inter Integrated Circuit |
| IHD | Interfaz Humano Dron |
| IHR | Interfaz Humano Robot |
| IMU | Unidad de Medición Inercial (siglas en inglés) |
| LED | Diodo Emisor de Luz (siglas en inglés) |
| LIPO | Batería de Polímero de Litio |
| NUI | Interfaz Natural de Usuario (siglas en inglés) |
| PDB | Placa de Distribución de Energía (siglas en inglés) |
| PID | Control Proporcional Integral Derivativo |
| PLA | Ácido Poliláctico |
| PPM | Modulación por Pulso Posicional (siglas en inglés) |
| PWM | Modulación por Ancho de Pulso (siglas en inglés) |
| RPA | Aeronave Pilotada Remotamente (siglas en inglés) |
| RX | Receptor |

| | |
|---|---|
| STEM | Ciencia, Tecnología, Ingeniería y Matemáticas (siglas en inglés) |
| TX | Transmisor |
| UAV | Vehículo Aéreo no Tripulado (siglas en inglés) |
| VANT | Vehículo Aéreo No Tripulado (siglas en inglés) |
| WIFI | Wireless Fidelity (siglas en inglés) |

## Resumen

El presente trabajo tiene como objetivo principal la construcción y evaluación de un dron educativo controlado por gestos manuales. Este proyecto busca innovar en la enseñanza de la mecatrónica mediante la implementación de tecnologías actuales que faciliten el aprendizaje práctico y atractivo para los estudiantes. El estudio abarca desde la selección de componentes y la programación del dron, hasta su ensamblaje y prueba, enfocándose en su accesibilidad y utilidad en entornos educativos con recursos limitados.

El trabajo destaca la importancia de la interacción hombre-máquina y cómo el control por gestos manuales puede mejorar significativamente la experiencia de uso de los drones. Los resultados obtenidos demuestran la viabilidad del control por gestos como una interfaz intuitiva y eficiente para la interacción con drones. Las pruebas realizadas han evidenciado un buen desempeño del prototipo en términos de estabilidad, precisión y respuesta a los comandos del usuario.

En conclusión, este trabajo ha demostrado que el dron controlado por gestos es una herramienta educativa valiosa que puede contribuir a motivar a los estudiantes y fomentar su interés por las disciplinas STEM. Se recomienda continuar investigando en esta área para ampliar las funcionalidades del dron y explorar nuevas aplicaciones en diversos campos.

**Palabras clave**

Dron, control, gestos, manuales, educación

**Abstract**

The main objective of this work is the construction and evaluation of an educational drone controlled by manual gestures. This project seeks to innovate in the teaching of mechatronics through the implementation of current technologies that facilitate practical and attractive learning for students. The study ranges from the selection of components and programming of the drone, to its assembly and testing, focusing on its accessibility and usefulness in educational environments with limited resources.

The work highlights the importance of human-machine interaction and how control by hand gestures can significantly improve the user experience of drones. The results obtained demonstrate the viability of gesture control as an intuitive and efficient interface for interaction with drones. The tests carried out have shown good performance of the prototype in terms of stability, precision and response to user commands.

In conclusion, this work has shown that the gesture-controlled drone is a valuable educational tool that can help motivate students and foster their interest in STEM disciplines. It is recommended to continue research in this area to expand the functionalities of the drone and explore new applications in various fields.

**Keywords**

Drone, control, gestures, manuals, education

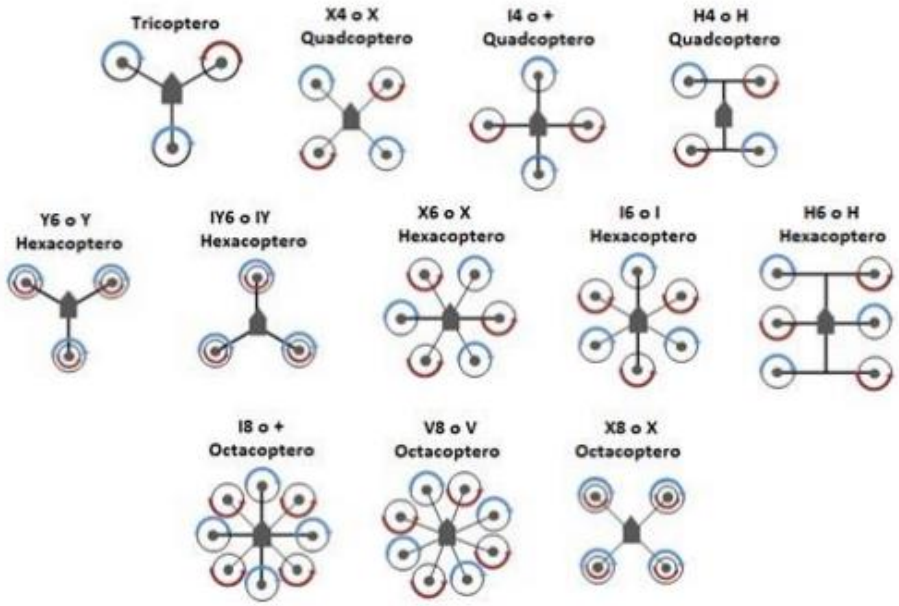
## Introducción

La interacción hombre-máquina se vuelve cada vez más avanzada y cómoda para el usuario. En este contexto, actualmente se encuentran dispositivos capaces de transmitir los movimientos que realiza un ser humano a un robot o máquina específica (Milán, 2017). Cabe destacar, que los drones son vehículos aéreos no tripulados los cuales pueden realizar diversas actividades como fotografía, cartografía, envío de paquetes o incluso educación. Sin embargo, para controlar un dron se requiere cierto grado de conocimiento o experiencia para realizar un vuelo adecuado. Por lo tanto, se hace necesaria una forma de facilitar el manejo de estos dispositivos con el uso de las tecnologías actuales. (Chico, 2022).

### Antecedentes

Un dron es un vehículo aéreo no tripulado que utiliza sensores, junto con comunicaciones de alta frecuencia, para elevarse en el aire y mantenerse estable mediante hélices. En un inicio, como plantea Lago (2017), los drones se utilizaron con fines militares en misiles y envío de proyectiles. Sin embargo, actualmente los drones tienen una amplia variedad de aplicaciones, por lo que se clasifican de acuerdo a ciertos factores (Sanz, 2019). Por ejemplo, se pueden clasificar según su estructura, como se muestra en la figura 1.

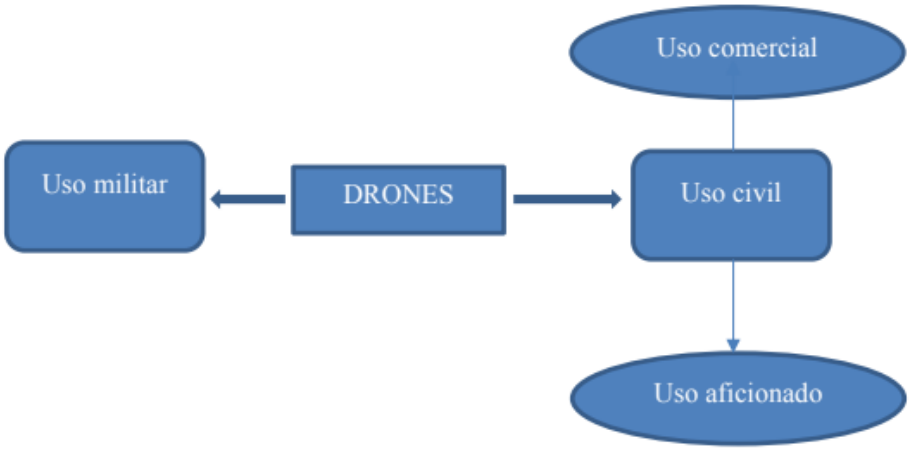
**Figura 1.** ***Clasificación según su estructura***




Fuente: (Sanz, 2019).

Además, los drones se pueden clasificar por su utilidad, como se muestra en el diagrama 1. Esta clasificación los divide en dos categorías principales: uso militar y uso civil. El uso civil se subdivide a su vez en uso comercial y uso de aficionados.

**Diagrama 1.** ***Clasificación según su utilidad***




Fuente: (Sanz, 2019).

Como explica Lago (2017), los drones que cuentan con dos y tres rotores no son usuales de observar, además de que presentan una mala maniobrabilidad. Por este motivo, se tiende a utilizar drones de cuatro ejes, ya que su manejo y construcción son más simples. En consecuencia, este tipo de aeronaves se vuelve ideal para estudiar el campo de aplicación de los drones (Llaguano, 2018).

Actualmente, en el mercado existe una gran diversidad de drones de cuatro ejes cuya principal diferencia radica en las características técnicas y los accesorios con los que cuentan. Por ejemplo, los drones que utilizan cámaras 4K, sensores de distancia para evasión de obstáculos, GPS o módulo Wi-Fi llegan a costar más de 1000$ como se observa en la tabla 1.

**Tabla 1.** ***Drones controlados por gestos***

| Dispositivo | Características principales | Precio |
|---|---|---|
| Dron DJI mini 3 Fly | • GPS<br>• Cámara 4K<br>• Tiempo de vuelo: 47m<br>• Plegable<br>• Auto retorno<br>• Evasión de obstáculos<br>• Control por gestos | 1049 $ |
| Dron QY66-K03 | • Cámara 0.3<br>• Tiempo de vuelo: 6m<br>• Plegable<br>• Evasión de obstáculos<br>• Control por gestos | 70 $ |
| Dron VAK 1841 | • Altura máxima: 30m<br>• Tiempo de vuelo: 10m<br>• Dimensión: 17x17cm<br>• Evasión de obstáculos<br>• Control por gestos | 16 $ |

Ahora bien, como plantea Chico (2022) la robótica y sistemas de reconocimiento de gestos son interfaces humano-máquina en donde se busca determinar

los movimientos del operario. Así pues, existen dos formas principalmente de obtener estos datos que son mediante cámaras y por medio de sensores o giroscopios como se muestra en la figura 2. Actualmente, dicho sistema se utiliza para aplicaciones en plataformas robóticas o manipulación de robots de exploración.

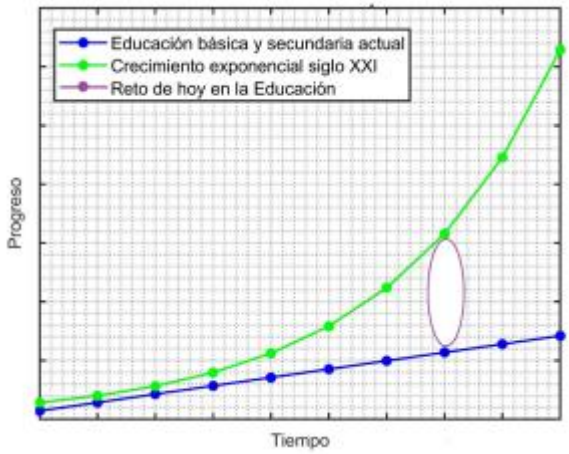
**Figura 2.** ***Detección de gestos***


Fuente: (Rodríguez, 2022).

**Planteamiento Del Problema**

Actualmente, la educación se ha convertido en un pilar fundamental para transformar y desarrollar las habilidades de los ciudadanos del siglo XXI. Sin embargo, la educación enfrenta un desafío importante: su crecimiento es lineal, mientras que la tecnología avanza de forma exponencial como se muestra en el diagrama 2. Esto significa que los métodos educativos tradicionales se están volviendo obsoletos, mientras que surgen nuevas formas de aprendizaje (Jiménez, 2020).

**Diagrama 2.** ***Visión lineal vs Visión exponencial de la educación***




Fuente: (Jiménez, 2020)

La educación en ingeniería, especialmente en las áreas de prácticas de mecatrónica, presenta desafíos considerables en la actualidad. Estos desafíos incluyen altos costos, requerimientos específicos de infraestructura y, en ocasiones, acceso limitado a los equipos necesarios. Como consecuencia, se restringe la cantidad de prácticas que pueden realizar los estudiantes, limitando así su experiencia práctica (Ríos, 2018).

Si bien el uso de drones en la educación genera interés en los estudiantes, la implementación de proyectos relacionados con esta tecnología aún es limitada. Uno de los factores que influyen en la elección de las prácticas es el costo y el conocimiento previo que requiere el alumno para aplicar adecuadamente la actividad. Por ello, se suele optar por utilizar otros componentes más accesibles y con mayor disponibilidad de información, aunque esto implique sacrificar la posibilidad de aplicar nuevos conceptos más atractivos y actuales (Bautista, 2020).

## Justificación

Por ello, el motivo del presente estudio es ensamblar un dron de cuatro ejes controlado por gestos manuales, dirigido a estudiantes de la carrera de mecatrónica de la Universidad Internacional SEK. De esta forma, se pretende que los alumnos cuenten con mayores recursos de aprendizaje y desarrollen habilidades de programación. Así mismo, se busca que el manejo del dron sea sencillo e intuitivo, facilitando el aprendizaje del vuelo de este tipo de aeronave. En este sentido, el proyecto pone de manifiesto la necesidad de este tipo de sistemas, donde se pueda aplicar el conocimiento en programación en conjunto con la facilidad de control de un dron.

Una de las pedagogías de enseñanza más recientes es el modelo STEM (Ciencias, Tecnología, Ingeniería y Matemática), que se enfoca en la combinación de estas cuatro disciplinas para la resolución de problemas. Este modelo ofrece una alternativa a los métodos tradicionales y permite a los estudiantes aprender de forma más interactiva y contextualizada (Bautista, 2020).

La integración de drones en la enseñanza STEM puede tener un impacto significativo en el aprendizaje de los estudiantes. Debido a que, los drones son sistemas mecatrónicos que integran elementos mecánicos, electrónicos y de programación, lo que los convierte en una herramienta ideal para el aprendizaje de estas áreas.

En este contexto, el desarrollo de un dron de cuatro ejes controlado por gestos manuales para la enseñanza de la mecatrónica, se presenta como una propuesta innovadora con un alto potencial para mejorar la calidad de la educación en la carrera.

Con base en estos antecedentes, el objetivo principal de este trabajo es evaluar un dron de cuatro ejes controlado por gestos manuales para su uso pedagógico mediante la aplicación de metodologías cuantitativas. Para lograr este objetivo principal, se plantean los siguientes objetivos específicos:

1. Determinar los requisitos técnicos y funcionales de un dron de cuatro ejes para su uso pedagógico mediante la definición de parámetros de diseño.
2. Diseñar el sistema electrónico y de comunicación del dron para su aplicación práctica mediante el uso de software open source.
3. Fabricar un prototipo funcional del sistema de manejo del dron por gestos manuales para el control del mismo, utilizando sensores de movimiento inercial.
4. Evaluar el funcionamiento del dron para la validación de su usabilidad pedagógica mediante pruebas de vuelo y control por gestos manuales.

**Hipótesis**

A partir de un dron de cuatro ejes controlado por gestos manuales, será posible su aplicación pedagógica dentro de la carrera de mecatrónica de la UISEK, de manera que facilite a los estudiantes desarrollar sus habilidades de programación y manejo de drones.

# Estado Del Arte

Los drones, o vehículos aéreos no tripulados (VANT), han irrumpido en el panorama actual como una tecnología disruptiva que está transformando diversos sectores. Estas aeronaves controladas a distancia, ofrecen un amplio abanico de aplicaciones y servicios, desde la vigilancia y topografía hasta el entretenimiento e incluso la educación.

Más allá de su uso tradicional, la incorporación de técnicas de control por gestos ha demostrado ser una herramienta valiosa para facilitar el manejo de los drones. Esta innovación no solo simplifica su manejo y control, sino que su aplicación educativa también contribuye a una mejor comprensión de los conceptos de electrónica y programación. De esta forma, permite a los estudiantes abordar el aprendizaje de manera más intuitiva y atractiva.

## Interacción Humano-Dron

La interacción humano-dron (IHD) se ha convertido en un campo de estudio relevante y en auge, impulsado por el creciente uso de drones o vehículos aéreos no tripulados (UAV) en diversos ámbitos. La IHD se enmarca dentro de la interacción humano-robot (IHR), lo que permite aprovechar los avances y conocimientos desarrollados en esta disciplina más amplia (Múnera, 2022).

Un aspecto clave a considerar es el nivel de autonomía en la interacción entre el humano y el dron, que puede variar desde la teleoperación completa hasta la colaboración de igual a igual (Gio, 2021). Esto implica diferentes grados de participación humana y

autonomía del sistema, los cuales deben ser cuidadosamente evaluados y diseñados en función de las tareas y contextos específicos.

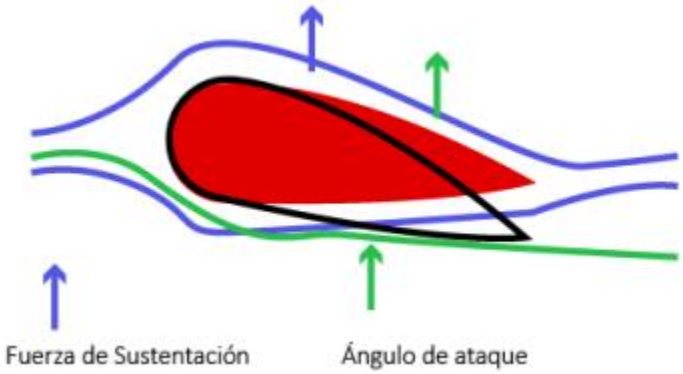
Como menciona Tezza (2019), el ser humano puede asumir el rol activo de controlador, operando directamente el dron a través de una interfaz para realizar tareas como la captura de imágenes o participar en carreras de drones. Así mismo, puede asumir el rol de destinatario, lo que implica que el usuario no controla el dron, pero se beneficia de la interacción con él, como recibir un paquete entregado por un dron.

**Principio De Un Dron**

El funcionamiento de un dron se basa en los mismos principios aerodinámicos que un avión. Las alas de los drones, al igual que las de los aviones, tienen un perfil alar curvo en la parte superior y plano en la inferior. Esta forma genera una diferencia en la velocidad del aire que fluye sobre el perfil, siendo mayor la velocidad en la parte superior que en la inferior. Esta diferencia de velocidades produce una fuerza de sustentación que permite al dron elevarse como se observa en la figura 3 (Cruz, 2021).

Para que un dron vuele, es crucial considerar el concepto de ángulo de ataque. Al inclinar ligeramente el perfil alar, se modifica el flujo de aire, generando una fuerza vertical ascendente que aumenta con el ángulo de ataque como se muestra en la figura 3. Sin embargo, para que esta fuerza sea suficiente para elevar el dron, se requiere una alta velocidad del aire sobre el perfil. Por esta razón, los drones necesitan alcanzar una cierta velocidad antes de poder despegar (Cruz, 2021).

**Figura 3.** ***Perfil alar***

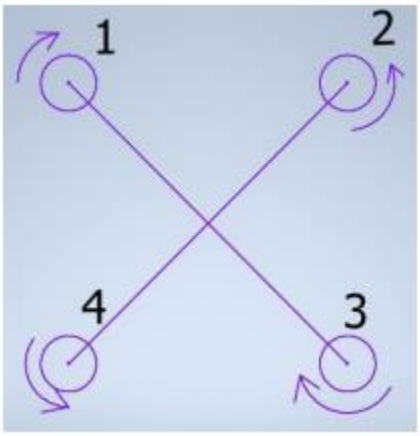



Fuente: (Cruz, 2021)

Por otro lado, existen diferentes tipos de drones, comúnmente clasificados en helicópteros de un solo rotor, de ala fija y multirrotor. Los drones de cuatro rotores, también conocidos como cuadricópteros, son los más comunes y pertenecen a la familia de los multirrotor. Estos drones generalmente tienen cuatro rotores dispuestos en una configuración cruzada, con dos pares de rotores opuestos girando en sentidos contrarios para generar equilibrio como se muestra en la figura 4 (Torres, 2021).

Al girar las hélices en una dirección, se genera un momento en el dron en sentido opuesto al giro, de acuerdo con la tercera ley de Newton (acción-reacción). Para contrarrestar la fuerza generada por la rotación de las hélices, se dispone de dos pares diagonales de motores (2 y 4) que giran en sentido antihorario. Mientras que el otro par de motores (1 y 3) gira en sentido horario como se muestra en la figura 4.

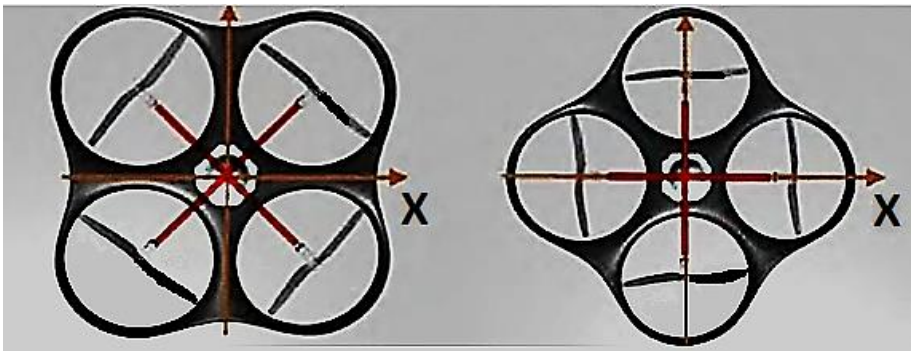
**Figura 4.** ***Configuración de motores dron***




Fuente: (Torres, 2021)

En este contexto, existen dos configuraciones básicas de cuadricópteros: plus y cruz como se muestra en la figura 5. La diferencia entre estas configuraciones radica en la ubicación de la parte frontal del dron. Además, para el mismo movimiento deseado, la configuración cruz proporciona un mayor impulso que puede mejorar el rendimiento de maniobrabilidad. (Kangunde, 2021).

**Figura 5.** ***Configuración cruz y plus de un dron***

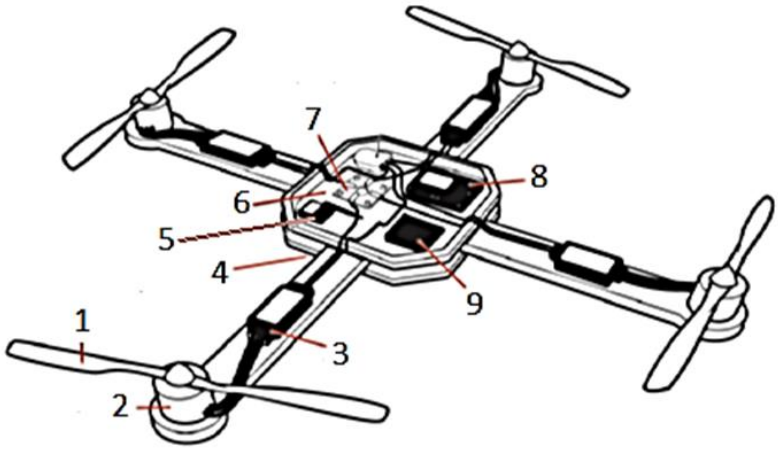



Fuente: (Kangunde, 2021)

**Componentes De Un Dron**

Los drones, también conocidos como vehículos aéreos no tripulados (VANT), están compuestos por diversos componentes esenciales para su correcto funcionamiento como se muestra en la figura 6.

**Figura 6.** ***Componentes de un dron***




Fuente: (Juárez, 2024)

Los principales componentes que se incluyen en un dron son:

1. **Hélices:** Las palas que giran impulsadas por los motores, generando el sustento aerodinámico que permite al dron mantenerse en el aire.
2. **Motores:** Los encargados de generar la potencia necesaria para que las hélices giren y producen el empuje para volar.
3. **ESC:** Los ESC actúan como intermediarios entre la batería y los motores, regulando la velocidad de giro de las hélices.

4. **Batería:** La fuente de energía que alimenta todos los componentes electrónicos del dron.
5. **Transmisor:** Permiten la comunicación entre el dron y el control remoto utilizado por el piloto para controlar.
6. **Chasis:** La estructura principal del dron que aloja y soporta todos los demás componentes.
7. **PDB:** La PDB distribuye de manera eficiente la energía de la batería a todos los componentes del dron.
8. **Controlador de vuelo:** El "cerebro" del dron, responsable de procesar los datos de los sensores, interpretar las instrucciones del piloto y controlar los motores para mantener el dron estable y en movimiento.
9. **Sensores:** Dispositivos electrónicos que recopilan información sobre el entorno del dron, como su posición, orientación, altitud y velocidad.

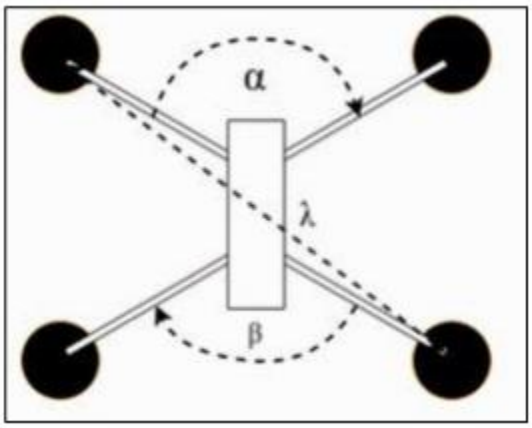
### *Chasis*

Como explica Fiesco (2022), el "airframe" o chasis del dron es la parte principal ya que ahí se colocarán los demás componentes electrónicos. Se fabrica con materiales como fibra de carbono, nylon o PLA y se caracteriza por tres elementos como se observa en la figura 7:

- Longitud: (λ) Distancia entre el eje de un motor y el del motor opuesto (wheelbase).
- Ángulos: (α) Grados de separación entre los brazos y (β) las patas del airframe.

- Geometría: Simétrica, no simétrica o híbrida.

**Figura 7.** ***Geometría de un dron X***




Fuente: (Fiesco, 2022)

El frame DJI-F450 es una estructura de dron versátil diseñada para diversas aplicaciones, incluyendo fotografía y videografía aérea, mapeo y vigilancia (Manrique, 2022). Está fabricado con la aleación PA66+30GF, una combinación de poliamida 66 (nylon 66) y fibra de vidrio al 30% en peso, que le otorga alta resistencia mecánica, rigidez y resistencia al calor (Manrique, 2022).

Las principales ventajas del frame DJI-F450 son su versatilidad, facilidad de uso, tamaño compacto, compatibilidad, capacidad de personalización, espacio adicional y diseño PDB integrado. Estas características lo convierten en una opción atractiva para usuarios que buscan un frame versátil y adaptable a diversas aplicaciones como se observa en la figura 8 (Prieto, 2023).

**Figura 8.** ***DJI Frame 450***

Fuente: (Prieto, 2023).

***Motores Brushless***

Los motores sin escobillas o brushless son una opción común en los drones debido a sus ventajas como menor ruido, menor sobrecalentamiento y mayor vida útil. Sin embargo, también tienen desventajas como precio más alto y necesidad de un ESC para su control.

El funcionamiento de estos motores se basa en la atracción entre imanes permanentes en el rotor y electroimanes en el estator como se muestra en la figura 9. Al cambiar la polaridad de los electroimanes rápidamente, se controla la rotación del motor (De las Heras, 2019). Se utilizan cuatro motores sin escobillas, los cuales requieren un controlador electrónico de velocidad (ESC) para el control del dron.

**Figura 9.** ***Motor Brushless***


Fuente: (Juárez, 2024).

***Controlador Electrónico De Velocidad (ESC)***

Los controladores de motores, conocidos como ESC (Electronic Speed Control), son componentes especializados que regulan la velocidad de los motores en los drones. Estos controladores convierten la alimentación de CC a CA, ajustan la frecuencia para controlar la velocidad de rotación y permiten la comunicación con el controlador principal como se observa en la figura 10 (Juárez, 2024).

La conexión entre el microcontrolador y el ESC se realiza mediante modulación de ancho de pulso (PWM). Un ciclo de trabajo mayor implica una velocidad de motor más alta. (Masoud, 2016). Para la elección de este componente es necesario revisar la corriente máxima que puede consumir el motor brushless.

**Figura 10.** ***Controlador de velocidad***


Fuente: (De las Heras, 2019)

***Batería***

El auge de los drones se debe en gran medida a los avances tecnológicos en áreas como las baterías de litio, la tecnología de microprocesadores y los sensores de medición inercial para la estabilización (Fouché & Malekian, 2018). Las baterías de Litio-Polímero (LiPo) ofrecen una alta densidad de energía, son ligeras y permiten una rápida recarga, lo que las convierte en la mejor opción para la alimentación de los drones. Este tipo de batería suele estar caracterizada por parámetros como el amperaje, el voltaje, el número de celdas y la capacidad de descarga (Montejo, 2023).

En los drones, la capacidad de la batería, medida en amperios-hora (Ah), determina la autonomía de vuelo. Sin embargo, es importante comprender que la capacidad de la batería no es el único factor que influye en la autonomía de vuelo.

Es fundamental que el voltaje de cada celda sea superior a 3.3 V ya que, si desciende por debajo de este nivel, la celda podría dañarse o incluso existir riesgo de ignición. Este riesgo aumenta si las celdas se descargan por completo o se almacenan durante largos periodos de tiempo con un voltaje bajo.

### *Hélices*

Las hélices son componentes esenciales en los vehículos aéreos no tripulados (UAV), ya que generan la sustentación necesaria para mantener el artefacto en el aire. Como menciona Pueyo (2018), el funcionamiento de una hélice se asemeja al de un ala, en donde el giro de las aspas acelera el aire hacia la punta, generando una deflexión de una gran masa de aire. Debido a la variación de velocidad a lo largo del aspa (mayor en la punta), el ángulo de ataque se modifica desde la raíz hasta la punta para compensar esta diferencia.

Para la selección adecuada de hélices en un UAV, se deben considerar diversos parámetros, entre los que destaca un código de cuatro números presente en cada hélice. Los dos primeros dígitos indican el tamaño de la hélice desde el eje de rotación hasta la punta, mientras que los dos últimos dígitos representan el ángulo de ataque de la misma (Pueyo, 2018).

La elección de las hélices está estrechamente relacionada con la selección previa del motor. Debido a que, el tamaño de la hélice se determina en función de las revoluciones del motor, siendo más grandes cuando el motor tenga menos revoluciones y más pequeñas cuando el motor gire más rápido

### *Controlador De Vuelo*

Actualmente, existen diversos controladores de vuelo que pueden ir desde microcontroladores hasta placas especializadas con procesadores de imágenes integrados. La principal diferencia entre los controladores de vuelo es la capacidad que tienen para maniobrar al dron. En este caso, se puede utilizar al microcontrolador Arduino como

controlador de vuelo ya que cuenta con una amplia disponibilidad de información sobre sus bibliotecas y el apoyo constante de la comunidad en línea.

Además, Arduino es una plataforma de código abierto basada en lenguajes C y Processing, y ofrece software gratuito como se observa en la figura 11. Como menciona Méndez (2015), los modelos Arduino UNO y Arduino Nano, pueden cumplir con los requisitos de control de un dron ya que cuenta con las siguientes características:

- Comunicación serie (TX y RX) para el intercambio de datos con un módulo Bluetooth.
- Comunicación I2C para los sensores, como el giroscopio MPU 6050.
- 4 salidas digitales para los actuadores.
- 1 entrada analógica para medir el nivel de batería.
- Pin de alimentación para periféricos a 5V y 3.3V

**Figura 11.** ***Placa Arduino Uno***

Fuente: (De las Heras, 2019).

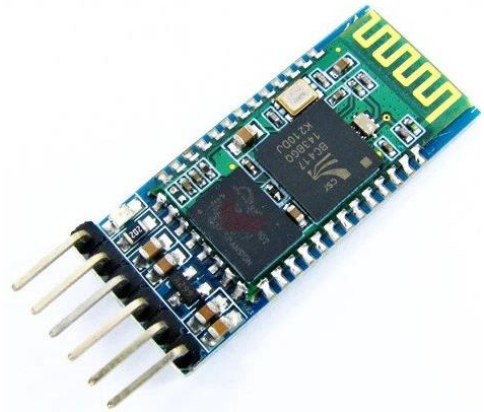
***Comunicación (Transmisor Y Receptor)***

Para la comunicación entre el dron y el controlador se utilizan transmisores y receptores como módulos de radiofrecuencia o bluetooth. Sin embargo, los módulos

bluetooth por su fácil implementación y reducción de pines necesarios para la conexión al microcontrolador resultan útiles de implementar. Para ello, la comunicación debe hacerse mediante un Arduino Maestro que será el que envié los datos para el control del dron mientras que el Arduino Esclavo será el que recepté y ejecuté órdenes.

Como explica Saltos (2018), el módulo HC-05 permite la comunicación por bluetooth actuando tanto como maestro como esclavo como se observa en la figura 12. Para ello, se debe utilizar los comandos AT para la configuración respectiva de cada módulo.

**Figura 12.** ***Módulo Bluetooth HC-05***



Fuente: (Saltos, 2018).

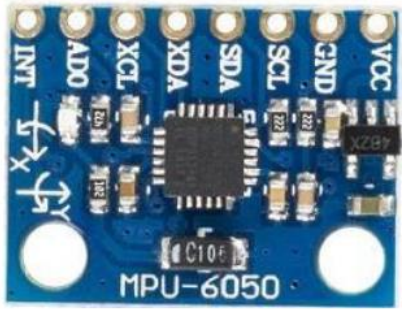
***Sensores***

Actualmente, se pueden utilizar diversos sensores para ser usados como sistemas de navegación. Entre los sistemas de navegación alternativos a los sistemas radioelectrónicos (incluidos los satelitales), la navegación inercial se destaca como una opción particularmente efectiva para los drones. Esta tecnología basa su funcionamiento en la medición del movimiento del dron a través de acelerómetros y giroscopios,

permitiendo una navegación precisa y confiable incluso en entornos donde las señales GPS o de radiocontrol son débiles o inexistentes (Alizada, 2023).

El auge de la navegación inercial en drones se ha visto impulsado por la reciente aparición en el mercado global de una amplia gama de sensores inerciales miniaturizados, relativamente económicos y con un alto nivel de precisión como se muestra en la figura 13. Estos sensores compactos y asequibles han permitido la integración de la navegación inercial en drones de diversos tamaños y presupuestos. De esta forma, se abre un abanico de posibilidades para su aplicación en diferentes campos.

**Figura 13.** ***Sensor MPU-6050***




Fuente: (Alizada, 2023).

***Control PID***

El control PID (proporcional, integral y derivativo) es una herramienta clásica de control ampliamente utilizada en diversos sistemas, incluyendo drones. Principalmente se utiliza para controlar una variable para que se mantenga constante que en este caso será el ángulo de giro del dron. Se utilizan dos ángulos de referencia: Pitch y Roll, omitiendo Yaw. Se calculan dos errores y se emplean dos controladores PID independientes para

cada uno. Las señales de control obtenidas se utilizan para modificar la velocidad de los cuatro motores del dron.

El ajuste PID se realiza utilizando el DMP (Digital Motion Processor) dentro del sensor MPU 6050 para obtener datos IMU (Inertial Measurement Unit) precisos y confiables. La biblioteca 'PID' se emplea como base para el ajuste de los parámetros PID, asegurando un control estable y preciso del dron (Jeong, 2023).

**Legislación en Ecuador**

Ahora bien, la Dirección General de Aviación Civil (DGAC) emitió el "Reglamento de Operación de Aeronaves Pilotadas a Distancia (RPAs)" el 4 de noviembre de 2020, para regular el vuelo de drones (Manrique, 2022). El reglamento establece que los drones de uso recreativo con un peso inferior a 25 kg no requieren seguro y pueden ser manipulados sin necesidad de haber tenido capacitaciones. Sin embargo, el rango de alcance del dron debe ser menor a 122 metros, excluyendo su vuelo de zonas de seguridad del estado y áreas restringidas.

**Métodos de control de drones**

Los drones han experimentado un auge significativo en los últimos años, impulsados por avances tecnológicos y una amplia gama de aplicaciones potenciales. Su versatilidad los convierte en herramientas valiosas en diversos sectores, desde la agricultura y la transmisión de eventos en vivo hasta la respuesta a emergencias y la educación (Cetin, 2023). Un factor clave en el éxito de los drones es el desarrollo de métodos de control efectivos que permitan a los usuarios interactuar con ellos de manera intuitiva y precisa.

Ahora bien, el lenguaje humano posee una doble función fundamental en la interacción humano-dron IHD, como herramienta para expresar simbólicamente el pensamiento propio y como herramienta para describir el estado del entorno (Bermejo, 2020). Esta doble faceta convierte al lenguaje simbólico en una característica única de la especie humana, diferenciándose del resto de los animales. La capacidad de utilizar el lenguaje para comunicar ideas, instrucciones y descripciones del entorno es esencial para una interacción efectiva entre humanos y drones.

Por eso, en los últimos años la mayoría de los drones comerciales han venido con controladores diseñados específicamente para el usuario, ya sean transmisores de señal dedicados o aplicaciones de software (Natarajan, 2018). Estos controladores envían comandos detallados de movimiento a través de canales inalámbricos, proporcionando un control preciso y confiable.

Recientemente, han surgido métodos de control alternativos que aprovechan tecnologías como los gestos con las manos y la visión por computadora. El uso de guantes especiales con sensores o la cámara a bordo del dron permiten el control mediante gestos intuitivos (Dong, 2020). En el ámbito académico, se han explorado interfaces naturales de usuario (NUI) con dispositivos como Leap Motion Controller o Microsoft Kinect para el control de drones mediante gestos corporales o movimientos de manos.

Entre los métodos emergentes, los sistemas RF ofrecen una solución asequible y sin contacto ni restricciones de luz. Debido a su precisión en la detección del movimiento

de la mano, los sistemas RF han ganado popularidad como un método eficaz de interacción humano-dron (IHD) en los últimos años (Budiyanto, 2021).

**Sistemas de control por gestos**

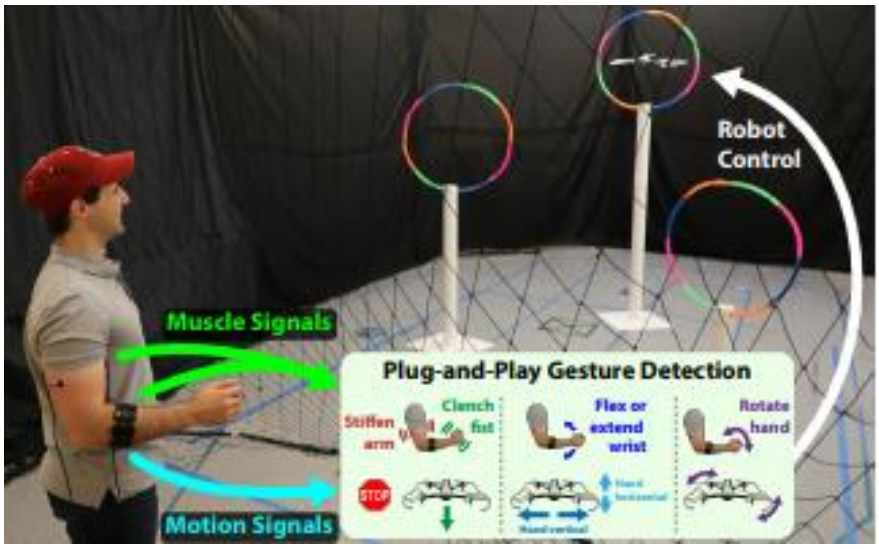
Dentro de la robótica, el control de movimiento a través de gestos es fundamental para planificar trayectorias, programar movimientos precisos y fluidos para el robot. Esto se logra mediante algoritmos de control de retroalimentación que corrigen errores y ajustan los movimientos en tiempo real (Martínez, 2023). Sin embargo, el control clásico presenta limitaciones al abordar sistemas no lineales, tiene rangos de operación restringidos y resulta complejo para manejar perturbaciones (Molina, 2020).

En la actualidad, se está explorando la aplicación de métodos de aprendizaje no supervisado en tareas relacionadas con drones. Estas técnicas ofrecen un enfoque prometedor para abordar desafíos como el etiquetado de datos para el entrenamiento de modelos supervisados en aplicaciones agrícolas. La estimación de mapas de profundidad a partir de imágenes y la detección de anomalías tomadas por drones (Shin, 2019).

Sin embargo, las unidades de medición inercial (IMU) han desempeñado un papel fundamental en el desarrollo de la robótica, especialmente en la robótica móvil y en sistemas que requieren análisis cinemático, calibración y teleoperación de robots o drones (Rodríguez, 2022).

El reconocimiento de gestos con las manos es un área de investigación en la interfaz persona-computadora. En donde, se busca desarrollar sistemas capaces de interpretar el movimiento de las manos como comandos para controlar dispositivos sin necesidad de tocar botones o pantallas como se observa en la figura 14 (Berarache, 2023).

**Figura 14.** ***Reconocimiento de gestos***




*Nota:* La imagen muestra un sistema de control para un dron por gestos, mediante una banda que detecta las contracciones del músculo del brazo. Tomado de *Plug-and-play Gesture Control Using Muscle and Motion Sensors* (p. 1), por J. DelPreto, 2020, International Conference on Human-Robot Interaction.

La naturaleza estadística del problema de reconocimiento de gestos debe tenerse en cuenta al utilizar este método de control. La variabilidad natural en los movimientos de las manos dificulta su control preciso. Por lo tanto, los gestos deben estar bien diferenciados para acomodar esta aleatoriedad y mantener resultados correctos. Se busca un equilibrio entre la intuición y simplicidad de los gestos, junto con su capacidad para distinguirse de otros gestos. Se destaca la importancia del uso de metodologías estadísticas sobre métodos no interpretables (como redes neuronales) (Chodhury, 2020).

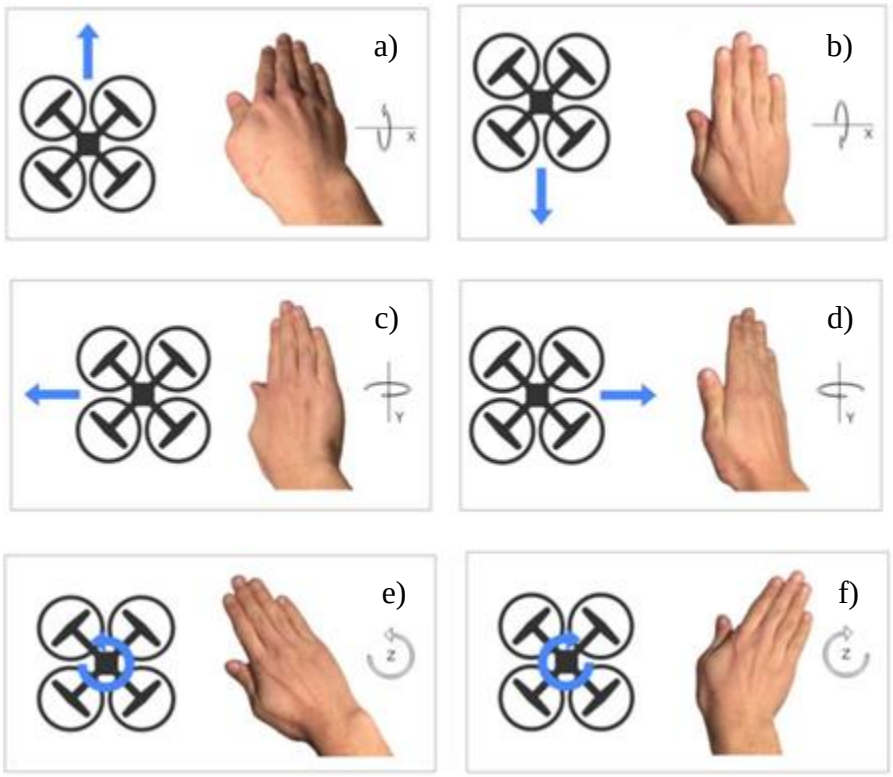
Como menciona Alizada (2023), el objetivo principal en el control de drones es optimizar la eficiencia de la recolección de datos para la navegación. Esto implica la

selección adecuada de sensores inerciales y la determinación de la cantidad de datos de navegación a promediar para reducir la interferencia de ruido.

En la implementación actual del control por gestos, se aplican 6 movimientos principales: rotación horaria, antihoraria, arriba, abajo, derecha o izquierda. La predicción de gestos se descarta si se detecta un movimiento extraño o diferente. Esta predicción se basa en la interferencia esperada entre gestos, como las rotaciones que tiene la mano. Si bien este marco es suficiente, futuras investigaciones podrían explorar alternativas como un clasificador o una máquina de estados que procese todas las predicciones de gestos (DelPreto, 2020).

Como explica Calleja (2016), la unidad de medición inercial (IMU) captura los valores de rotación de la mano del usuario en los ejes X, Y, Z. Posteriormente, se necesita transformarlos en valores de Pitch, Roll y Yaw que se enviarán al dron para su control. El objetivo es que, al inclinar la mano hacia adelante o atrás, el dron se desplace en la misma dirección (Pitch), como se observa en la figura 15. De igual manera, al inclinar la mano a derecha o izquierda, el dron se desplazará en esa dirección (Roll), según se muestra en la figura 15. Finalmente, al girar la mano sobre el eje vertical, el dron realizará la misma rotación (Yaw), tal y como se muestra en la figura 15.

**Figura 15.** ***Movimientos del dron Pitch, Roll y Yaw***




*Nota:* La imagen muestra el control del dron por medio de gestos de la mano como en el recuadro a), que muestra el movimiento del dron hacia adelante, el recuadro b) indica el movimiento del dron hacia atrás, en el recuadro c) se observa el movimiento del dron hacia la izquierda, en el recuadro d) se visualiza el movimiento del dron hacia la derecha, en el recuadro e) se muestra el movimiento del dron en sentido antihorario y el recuadro f) indica el movimiento del dron en sentido horario. Tomado de *Control Remoto de un drone* (p. 11), por J. Calleja, 2016, Universidad Politécnica de Cataluña.

**Aplicaciones de los drones**

Los drones no solo se limitan a aplicaciones militares o de seguridad, sino que también pueden ser herramientas valiosas para promover la paz y el bienestar de la

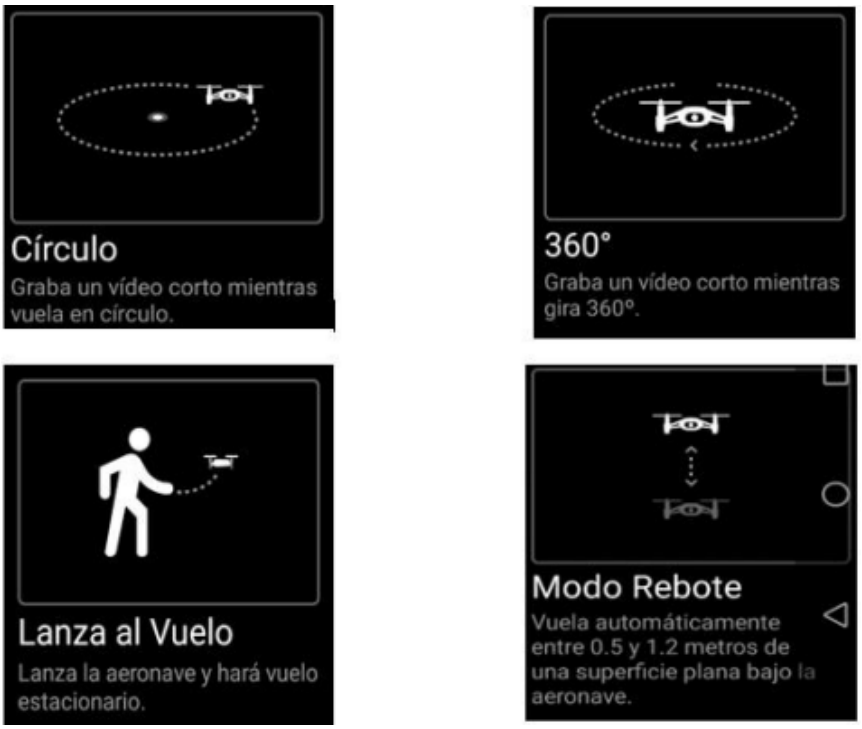
ciudadanía, especialmente en misiones humanitarias (Rodríguez, 2022). Estas aeronaves no tripuladas pueden realizar tareas que no necesariamente deben ser llevadas a cabo por humanos, brindando apoyo en situaciones de emergencia o áreas de difícil acceso.

Existen diversos tipos de drones, cada uno con componentes específicos diseñados para cumplir una función determinada. Para comprender mejor el funcionamiento básico de un dron, es necesario describir las partes comunes a todos ellos, es decir, los componentes esenciales para su operación (Llordes, 2016).

Según Bai (2021), la tecnología de drones tiene un gran potencial para integrarse en los planes de estudio, reforzando la comprensión de conceptos fundamentales y apoyando el proceso de aprendizaje de los estudiantes. Esta tecnología puede contribuir al desarrollo del pensamiento crítico y el razonamiento en los alumnos.

El objetivo educativo de los drones es enseñar a personas sin experiencia previa a volar dichos vehículos aéreos. Como menciona Jiménez (2019), se pueden utilizar diferentes formas, una de ellas puede ser enviando comandos específicos al dron para realizar acciones cortas o básicas como se muestra en la figura 16.

**Figura 16.** ***Comandos para acciones cortas***




*Nota:* En la imagen se observa los modos que puede tener un dron, como se indica en el recuadro a) puede ser el movimiento en círculo, en el recuadro b) se muestra el modo de 360°, en el recuadro c) se indica que se puede lanzar al dron desde la mano para hacerlo volar, en el recuadro d) se visualiza el modo rebote en donde el dron puede descender y ascender por sí solo. Tomado de *Using Drone Tello Edu for educational purposes* (p. 11), por J, Jiménez, 2019, Universidad Politécnica de Cataluña.

La interacción de los estudiantes con drones facilita el aprendizaje y la asimilación de conceptos teóricos, algoritmos y técnicas. Los ejercicios promueven el paradigma del aprendizaje activo. Por lo general, se desarrollan dentro de un marco robótico específico, utilizando una herramienta o conjunto de enseñanza donde el comportamiento del dron se programa utilizando un lenguaje particular. Cabe destacar la

diversidad de plataformas de hardware utilizadas para respaldar el componente experimental de los cursos de robótica (Cañas, 2020).

La evidencia sugiere que los tiempos han cambiado y los robots voladores programables están entrando en el segmento de la educación robótica y, por lo tanto, también estarán más presentes en las competiciones de robótica (Petrovic, 2023).

Por ejemplo, Drone Challenge es un concurso para equipos de estudiantes, en el que cada equipo debe programar el sistema de navegación automática de un dron. De hecho, aunque los sistemas de navegación se prueban en un entorno de simulación, el premio final es un dron comercial para cada uno de los miembros del equipo ganador (Bermúdez, 2019).

Debido a la diversidad de drones disponibles en el mercado, es importante seleccionar el modelo más adecuado para fines educativos y experimentales. La elección de un dron se debe basar en la disponibilidad de información y recursos relacionados para su construcción (Saini, 2021).

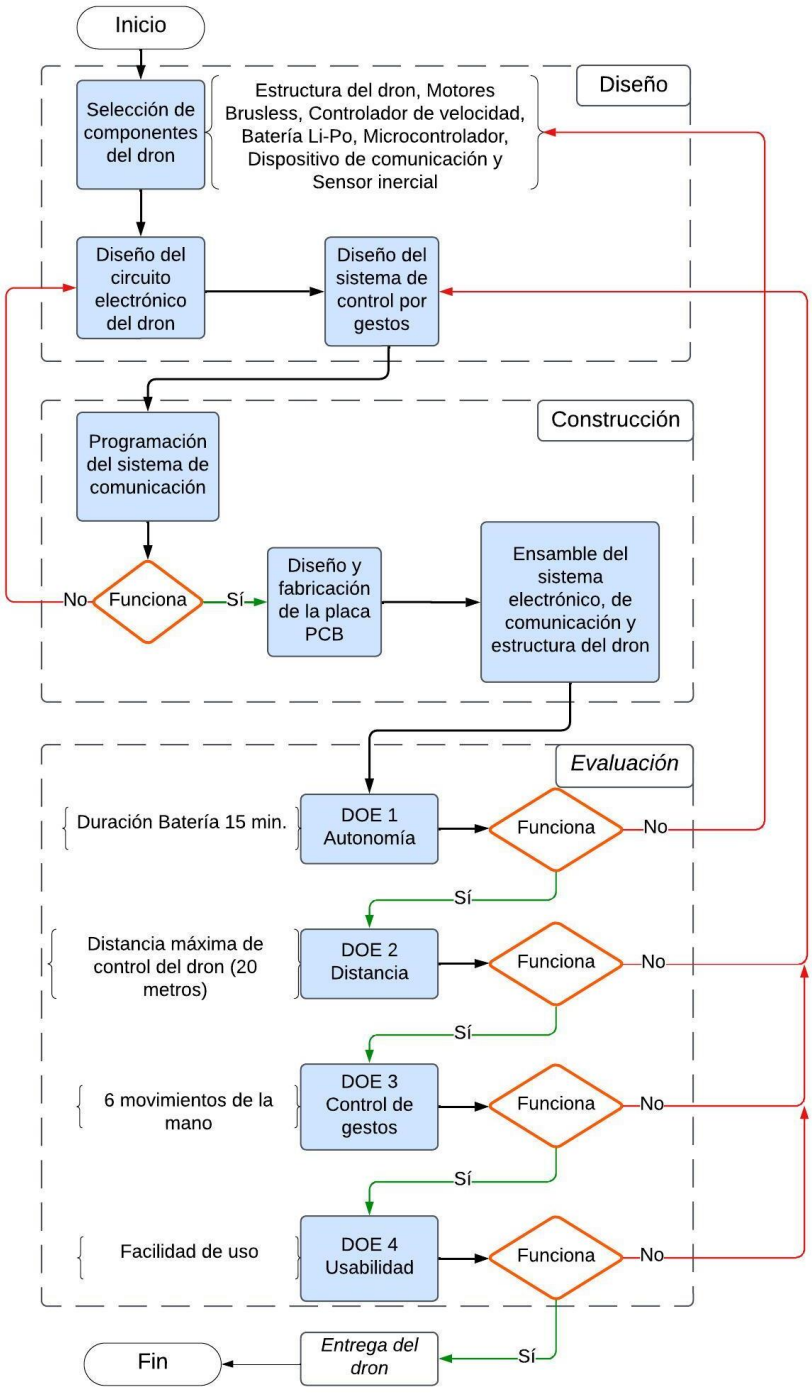
## Metodología

La presente investigación se fundamenta en un enfoque cuantitativo, en donde el objetivo es medir el rendimiento de un dron de cuatro ejes controlado por gestos manuales. Este enfoque se seleccionó debido a que busca evaluar de manera cuantitativa el desempeño del dron.

La investigación cuantitativa será experimental ya que, se presentará un prototipo del dron con las funciones necesarias para cumplir el objetivo principal. Adicionalmente, se contempla la aplicación de este dron en el ámbito educativo como herramienta de manipulación para el aprendizaje de sistemas de control**.** Para tal fin, se utilizarán métodos de observación directa, análisis de video, y experimentos**.** Estos métodos permitirán recopilar datos cuantitativos sobre el rendimiento del dron en un ambiente controlado.

En síntesis, se propone el siguiente diagrama de procesos del desarrollo de la investigación. El cual está divido en 3 etapas globales, etapa 1 diseño, etapa 2 construcción y etapa 3 evaluación como se muestra en el diagrama 3.

**Diagrama 3. Diagrama de procesos del dron**



**Selección de componentes**

La construcción de un dron implica la cuidadosa selección de componentes para garantizar un vuelo seguro, eficiente y satisfactorio. Este proceso requiere considerar diversos factores, comenzando por la definición del propósito del dron. En este caso el dron será utilizado para fines educativos lo que implica que puede ser más robusto y grande sin necesidad de que sea rápido.

Dentro de la etapa de diseño el primer paso que se realizó fue la selección de componentes para construir el dron. A continuación, se enlista los materiales utilizados para la construcción del dron:

1. **Chasis**: Se eligió el chasis DJI Frame 450mm debido a su facilidad para conectar la batería con los ESC y su tamaño adecuado para la instalación de los demás componentes.
2. **Motores Brushless:** Se seleccionaron motores brushless A2212 13T 1000KV, considerando el peso estimado del dron y su mayor potencia.
3. **ESC:** Se eligió el controlador Simonk 30A ESC, en base al amperaje máximo que puede consumir el motor (12A).
4. **Hélices**: Se seleccionaron hélices Mgsystem 10x45 2212 para dron, eligiendo el tamaño en base a la mayor eficiencia del motor indicada en la hoja de datos.
5. **Controlador de vuelo**: Se eligió el Arduino Uno por su facilidad de uso para los estudiantes y flexibilidad para programar.

6. **Sensor inercial:** Se eligió el sensor MPU6050 por sus características, incluyendo un acelerómetro y giroscopio.
7. **Módulo de comunicación:** Se eligió el módulo HC-05 Bluetooth por su compatibilidad con Arduino y facilidad de configuración.
8. **Batería:** Se eligió una batería Gens LiPo 3S 11.1V 4000mAh 60C, debido a su compatibilidad con los motores y ESC. En donde, se estimó una duración de vuelo de 15 minutos.
9. **Resistencias:** Se seleccionaron resistencias de 1k ohm, 500 ohm y 220 ohm para realizar el divisor de voltaje y la conexión del led indicador del voltaje de la batería.
10. **Diodo:** Se eligió un diodo 1N4001 en base al amperaje que puede soportar.
11. **Led**: Se eligió un LED de 5 mm de color rojo por su visibilidad y atractivo para el usuario.
12. **Protoboard:** Se eligió una Protoboard para Arduino Uno por su facilidad de implementación en el microcontrolador.

Ahora bien, para el sistema de control por gestos manuales se utilizó la siguiente lista de materiales:

1. **Controlador:** Se eligió un Arduino Nano por su tamaño compacto, su facilidad de programación y compatibilidad con el sensor junto con sistema de comunicación.

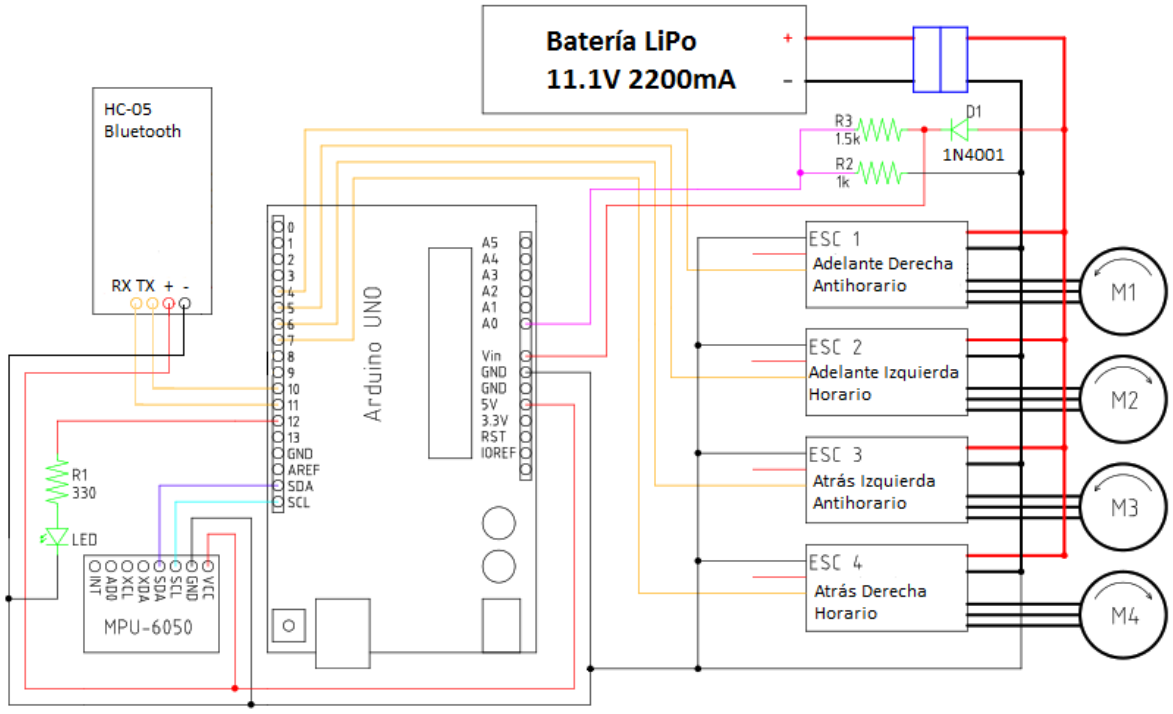
2. **Sensor inercial**: Se eligió el sensor MPU6050 por sus características de giroscopio y compatibilidad con Arduino.
3. **Módulo de comunicación**: Se eligió el módulo HC-05 Bluetooth por su facilidad de comunicación y configuración con Arduino.
4. **Batería:** Se eligió una batería Anera LiPo 2S 500mAh 35C por su relación de tamaño y duración.
5. **Resistencia:** Se eligió una resistencia de 1k ohm para evitar falsos contactos con el pulsador.
6. **Pulsador:** Se eligió un pulsador de 6 x 6 x 5 mm de 2 pines por su tamaño compacto.
7. **Interruptor:** Se eligió un interruptor deslizable pequeño de 3 pines por su tamaño compacto.
8. **Protoboard:** Se eligió una Protoboard para Arduino Nano por su facilidad de implementación en el microcontrolador.
9. **Sujeción:** Se eligió una tira de velcro de 24 cm por su facilidad de implementación, versatilidad y adaptabilidad.

**Diseño del sistema electrónico**

Tras seleccionar los componentes, se procedió al diseño del sistema electrónico, dividiéndolo en dos partes: dron y mando. Para esto, se realizó un esquema de conexiones utilizando el programa Paint donde se aprecia el diseño de manera más comprensible como se muestra en el diagrama 4. Para el diseño 2D de los componentes se utilizó

imágenes de internet asegurando la correspondencia entre la representación gráfica y la funcionalidad del circuito.

**Diagrama 4.** ***Esquema general del circuito del dron***

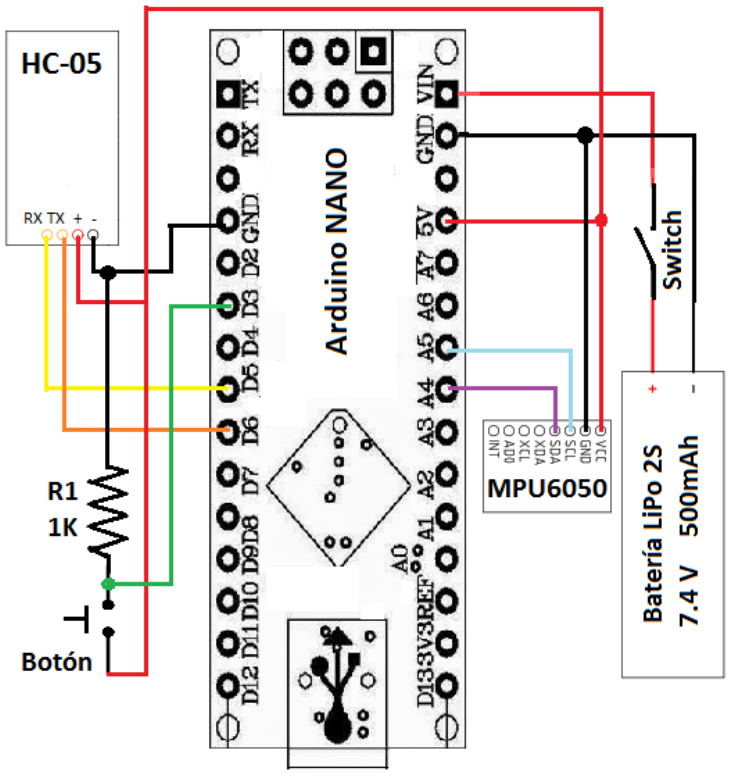



*Nota:* El diagrama muestra el circuito general del dron, principalmente se muestra la conexión de los motores con los ESC y la respectiva disposición en los pines digitales del Arduino Uno. También se indica la conexión del módulo bluetooth y del sensor MPU6050. Además, se visualiza la conexión de la batería LiPo y su divisor de voltaje con resistencias en conjunto de un diodo para la protección del Arduino Uno.

De forma similar al diseño del dron, el sistema electrónico del mando a distancia también se creó utilizando Paint. Esta herramienta facilitó la comprensión y el ensamblaje del circuito electrónico, permitiendo visualizar la ubicación de los

componentes y las conexiones entre ellos como se observa en el diagrama 5. En donde se puede observar la implementación de un botón y un interruptor para obtener más control sobre el mando. El interruptor cumple el papel de encender y apagar el mando en caso de ser necesario. Mientras, que el botón cumple el papel de empezar el ciclo del dron, es decir que este calibra el mando y empieza a emitir señales al dron.

**Diagrama 5.** ***Esquema general del circuito del mando***




*Nota:* El diagrama muestra el circuito general del mando, principalmente se muestra la conexión del módulo bluetooth y del sensor MPU6050. Además, se visualiza la conexión de la batería LiPo con su interruptor y la del botón que activa la programación.

**Sistema de control por gestos**

El sistema de control por gestos manuales permite operar el dron mediante movimientos intuitivos de la mano. El sistema se basa en la detección de la inclinación de la mano en los tres ejes espaciales (X, Y, Z) para generar comandos de control para el dron. Para esto, el sistema utiliza el sensor inercial MPU6050 para medir la inclinación de la mano en los tres ejes. El sensor proporciona datos de aceleración y de giroscopio que se procesan para determinar la orientación de la mano en el espacio.

Los valores de inclinación brutos obtenidos del sensor MPU6050 se transforman en ángulos utilizables para el control del dron. Esta transformación implica la aplicación de algoritmos matemáticos para convertir los datos de aceleración y giroscopio en representaciones angulares significativas.

Los ángulos de inclinación de la mano se envían al dron mediante Bluetooth utilizando el módulo HC-05. La comunicación Bluetooth permite transmitir los datos de control de forma inalámbrica entre la mano del usuario y el dron. Se han definido tres movimientos básicos para controlar el dron utilizando el sistema de gestos manuales:

- **Aceleración (throttle):** Controla la velocidad vertical del dron, determinando si asciende o desciende.
- **Inclinación (pitch):** Controla el movimiento angular del dron hacia adelante o hacia atrás.
- **Balanceo (roll):** Controla el movimiento angular del dron hacia la izquierda o hacia la derecha.

**Construcción del dron**

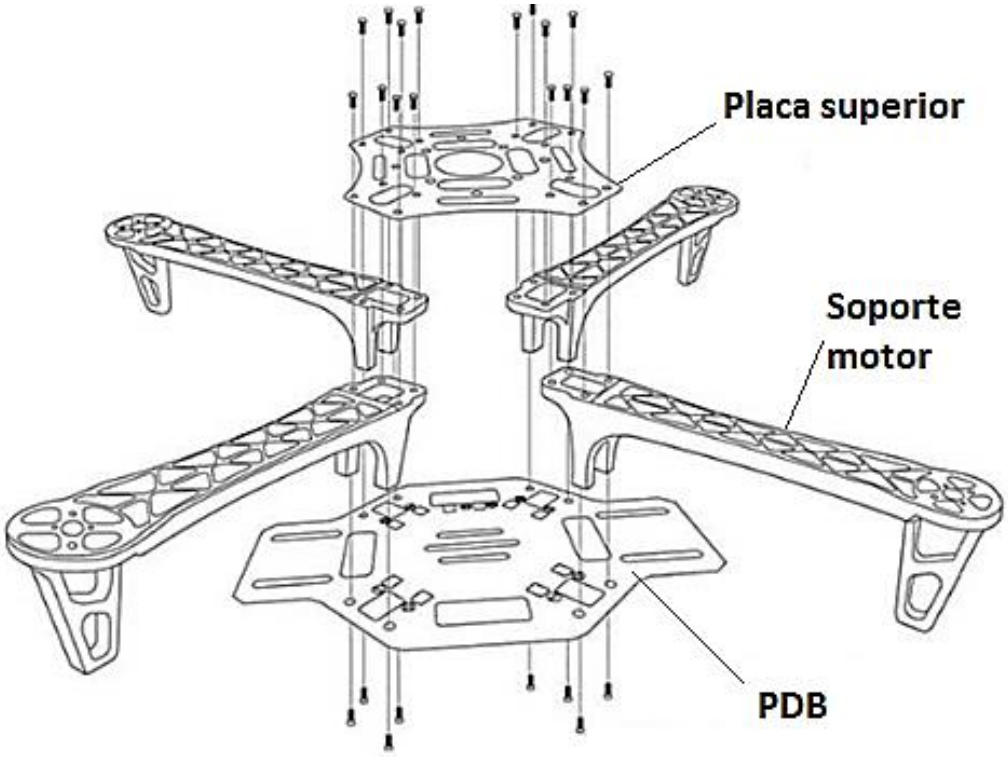
Después de definir los componentes y los diseños de los circuitos, se procedió a la etapa de construcción. Para asegurar un proceso ordenado y eficiente, se siguió una secuencia específica de pasos:

1. **Ensamblaje del chasis:** Se inició con el ensamblaje del chasis, la estructura principal del dron, siguiendo las instrucciones del manual del fabricante.
2. **Montaje de motores y controladores:** Una vez ensamblado el chasis, se procedió al montaje de los motores y sus respectivos controladores de velocidad (ESC). Se verificó la correcta conexión de los cables entre los motores, ESC y el circuito electrónico.
3. **Instalación de la batería:** Se instaló la batería de LiPo en el chasis, asegurando su fijación adecuada y la conexión correcta con el circuito electrónico. Se tomó en cuenta la distribución de peso para mantener el equilibrio del dron.
4. **Integración del circuito electrónico:** Finalmente, se integró el circuito electrónico, montado previamente en una protoboard, en el chasis del dron. Se fijó la protoboard de manera segura y se realizaron las conexiones necesarias con los demás componentes.

Ahora bien, como se observa en el diagrama 6, el chasis seleccionado está compuesto por varias piezas que deben ensamblarse:

- **4 Brazos:** Estos soportes permiten la instalación segura y eficiente de los motores en el chasis.
- **2 Placas:** Una de estas placas es la PDB integrada, mientras que la otra se utiliza para montar los demás componentes.
- **16 Tornillos 450-M3 x 8:** Estos tornillos se utilizan para montar los motores a los brazos.
- **24 Tornillos 450-M2.5 x 6:** Estos tornillos se utilizan para ensamblar los brazos con las placas.

**Diagrama 6.** ***Estructura del chasis DJI Frame 450***

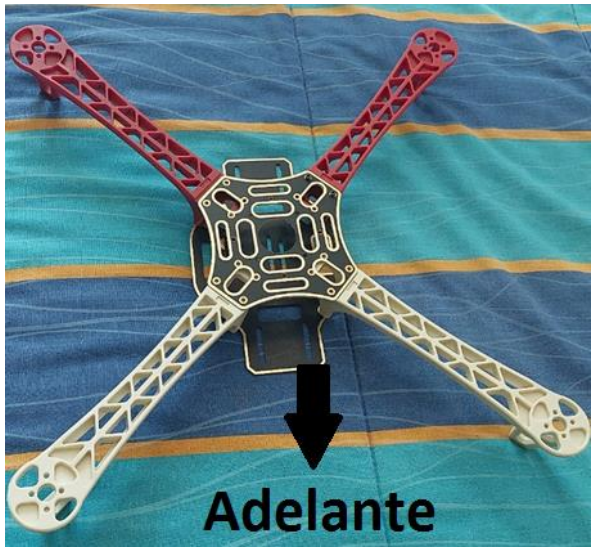



Fuente: (DJI, 2015).

Para facilitar la identificación de la parte frontal del dron durante el vuelo y la programación, se implementó un esquema de dos colores en los brazos del chasis, como se ilustra en la figura 17. Los dos brazos blancos del chasis conforman la parte frontal del dron, mientras que los dos brazos rojos conforman la parte trasera.

Esta diferenciación de colores ofrece varias ventajas. En primer lugar, facilita la visualización clara de la parte frontal del dron durante el vuelo. En segundo lugar, simplifica la programación de los motores y otros componentes, ya que se puede establecer una referencia clara para los comandos de control. Finalmente, también facilita la numeración de motores, ya que se puede asignar una numeración lógica en función de la ubicación frontal o trasera de cada brazo.

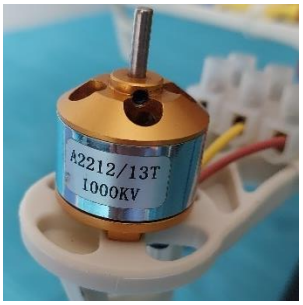
**Figura 17.** ***Identificación de la parte frontal del dron***



A continuación del ensamblaje del chasis, se procedió al montaje de los motores brushless y sus respectivos ESC en cada brazo del dron. Para ello, se siguieron los siguientes pasos:

1. **Fijación de los motores:** Se aseguraron los motores brushless a cada brazo del dron utilizando los tornillos 450-M3 x 8 proporcionados por el fabricante del chasis como se muestra en la figura 18.

**Figura 18.** ***Fijación de motores al dron***

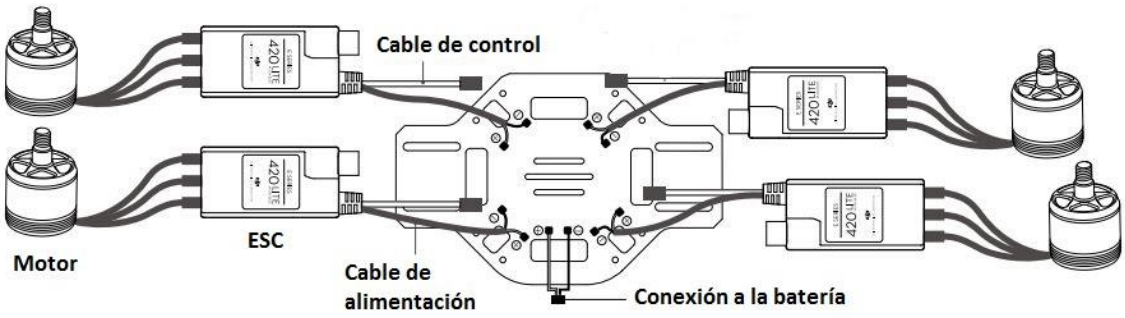



2. **Montaje de los ESC:** Se utilizaron bridas plásticas para fijar los ESC a los brazos del chasis. Esta sujeción permitió una instalación flexible y ajustable, facilitando la conexión de los cables y la distribución del peso del dron como se muestra en la figura 19.

**Figura 19.** ***Fijación de los ESC a los brazos del dron***

3. **Conexión de la alimentación:** Se soldaron los cables de alimentación de cada ESC a la placa PDB, la cual se conecta a su vez a la batería como se muestra en el diagrama 7.

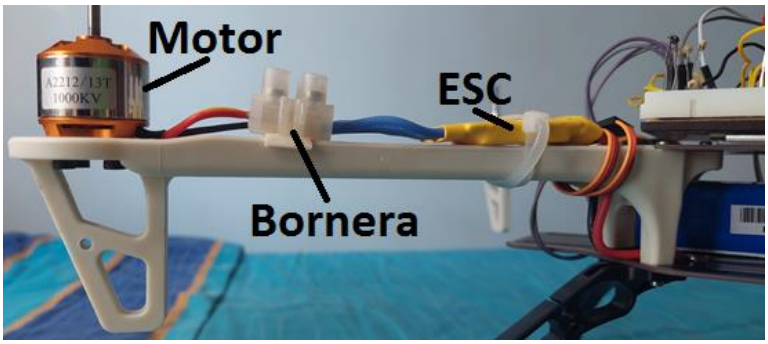
**Diagrama 7.** ***Placa PDB con ESC y motores***




Fuente: (DJI, 2015)

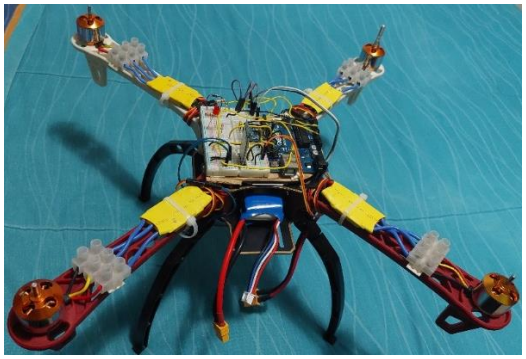
El montaje del motor y el ESC en cada brazo del dron se detalla en la figura 20. Para facilitar la conexión entre el motor y su respectivo ESC, se utilizó una bornera de 3 pines. Esto resulta particularmente útil, ya que el sentido de giro del motor es crucial para el correcto funcionamiento del dron. El sentido de giro puede modificarse intercambiando dos cables del ESC.

**Figura 20.** ***Montaje del circuito del dron en protoboard***



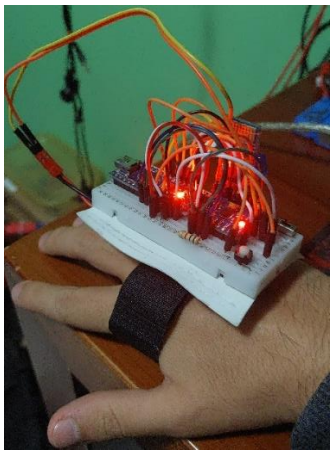

A continuación, se procedió al ensamblaje del circuito electrónico del dron y del mando a distancia. Para ello, se utilizó una protoboard, tal como se ilustra en la figura 21. La protoboard facilitó el proceso de montaje y conexión de los componentes electrónicos, proporcionando una plataforma flexible y modular para el ensamblaje.

**Figura 21.** ***Montaje del circuito del dron en protoboard***



De igual forma, se colocaron cuidadosamente los componentes electrónicos del mando sobre la protoboard, siguiendo el diagrama de conexiones predefinido. La disposición de los componentes se realizó de manera organizada y eficiente, teniendo en cuenta las características y conexiones de cada elemento como se observa en la figura 22.

**Figura 22.** ***Montaje del circuito del mando en protoboard***



**Programación del dron**

La programación del dron se inició sin la presencia de hélices en los motores para evitar riesgos potenciales durante las primeras etapas de pruebas. Se utilizó el software gratuito Arduino IDE para desarrollar y cargar el código necesario para el control y funcionamiento del dron. Para ello, se empezó con una comprobación exhaustiva de cada componente con el fin de conocer su comportamiento y las librerías que necesita.

El orden que se siguió para la comprobación de los componentes es la siguiente:

1. Motores y ESC
2. MPU6050
3. HC-05
4. Divisor de voltaje
5. Control PID

A continuación, se indica las respectivas comprobaciones que se realizó a cada componente:

***Motores y ESC***

La calibración de los ESC es un paso crucial para garantizar el correcto funcionamiento de los motores brushless en un dron. Este proceso implica ajustar los rangos de señal PWM que el controlador de vuelo envía a los ESC para controlar la velocidad de los motores.

En este caso, la calibración de los ESC se realizó mediante el envío de señales PWM con un periodo entre 1ms y 2ms**.** Este rango de valores ajusta los valores mínimos y máximos que puede alcanzar la velocidad de los motores. Para realizar la calibración, se empleó la librería Servo.h.

El código de Arduino que se muestra en la figura 23 se utiliza para controlar los cuatro motores de un dron utilizando la librería Servo.h. Esta programación permite calibrar los motores y ajustar la velocidad mediante comandos enviados a través del monitor serial. El resto del código se puede visualizar de mejor forma en el anexo A.

El código se divide en dos secciones principales: la configuración inicial en la función “setup()” y el bucle principal en la función “loop()”. A continuación, se describen las partes más relevantes del código.

1. **Declaración de variables y librerías:** En la primera parte del código, se declaran las variables necesarias para el proceso de calibración y se incluye la librería Servo.h.
2. **Inicialización:** En esta sección, se inicializan los pines del Arduino que se utilizarán para controlar los ESC y se configuran los parámetros de la librería Servo.h.
3. **Bucle de calibración:** En esta sección, se implementa un bucle que repite el proceso de envío de señales PWM a los ESC. El bucle itera hasta que se alcanza el valor máximo de PWM (2ms) y luego decrementa el valor hasta llegar al valor mínimo (1ms). Durante cada iteración, se observa el comportamiento de los motores para verificar que la calibración se realiza correctamente.

**Figura 23.** ***Programación para la calibración de los ESC con la librería Servo.h***

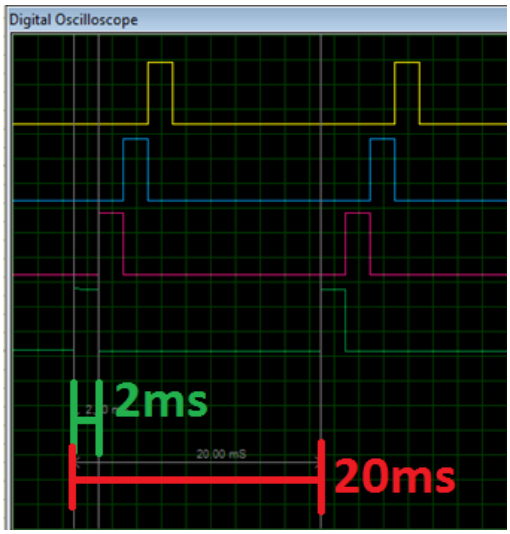
```
#include <Servo.h> //Cargar librería Servo.h

#define MAX_SIGNAL 2000 //Variable para la velocidad máxima del motor (2ms)
#define MIN_SIGNAL 1000 //Variable para la velocidad mínima del motor (1ms)
#define MOTOR_PIN1 4 //Define el pin para el motor 1
#define MOTOR_PIN2 5 //Define el pin para el motor 2
#define MOTOR_PIN3 6 //Define el pin para el motor 3
#define MOTOR_PIN4 9 //Define el pin para el motor 4

//Variable para guardar el valor de entrada del monitor serial
int DELAY = 1000;


//Asigna un nombre para cada motor con la librería servo
Servo motor1;
Servo motor2;
Servo motor3;
Servo motor4;
```

```
void setup() {
  Serial.begin(9600); //Inicializa el monitor serial
  delay(2000);
  //Inicia imprimiendo que el programa se empezó a ejecutar
  Serial.println("Program begin...");
  //Asigna el nombre de cada motor con su pin de salida digital
  motor1.attach(MOTOR_PIN1);
  motor2.attach(MOTOR_PIN2);
  motor3.attach(MOTOR_PIN3);
  motor4.attach(MOTOR_PIN4);

  Serial.print("Now writing maximum output: (");
  Serial.print(MAX_SIGNAL);
  Serial.print(" us in this case)");
  Serial.print("\n");
  //Indica que es momento de conectar la batería y esperar 2 segundos
  //antes de presionar cualquier tecla
  Serial.println("Turn on power source, then wait 2 seconds and press any key.");
  //Se envía una señal se 2ms a cada motor
  motor1.writeMicroseconds(MAX_SIGNAL);
  motor2.writeMicroseconds(MAX_SIGNAL);
  motor3.writeMicroseconds(MAX_SIGNAL);
  motor4.writeMicroseconds(MAX_SIGNAL);


  //Espera a que se presione cualquier tecla
  while (!Serial.available());
  Serial.read();
  //Imprime que se va enviar la señal mínima a cada motor
  Serial.println("\n");
  Serial.println("\n");
  Serial.print("Sending minimum output: (");
  Serial.print(MIN_SIGNAL);
  Serial.print(" us in this case)");
  Serial.print("\n");
  //Envía la señal mínima de 1ms a cada motor
  motor1.writeMicroseconds(MIN_SIGNAL);
  motor2.writeMicroseconds(MIN_SIGNAL);
  motor3.writeMicroseconds(MIN_SIGNAL);
  motor4.writeMicroseconds(MIN_SIGNAL);
  //Indica que los motores están calibrados y se puede enviar señales PWM
  //Las señales pueden ser desde 1ms hasta 2ms
  Serial.println("The ESC is calibrated");
  Serial.println("----");
  Serial.println("Now, type a values between 1000 and 2000 and press enter");
  Serial.println("and the motor will start rotating.");
  Serial.println("Send 1000 to stop the motor and 2000 for full throttle");
}
```

```
void loop() {
  //Si el monitor serial está disponible leer el valor de la entrada
  if (Serial.available() > 0)
  {
    //Asigna el valor de la entrada del monitor serial a la variable Delay
    int DELAY = Serial.parseInt();
    //Si el Delay es mayor a 1000us enviar la señal a cada motor
    if (DELAY > 999)
    {
      //Prende cada motor con el intervalo de Delay en microsegundos
      motor1.writeMicroseconds(DELAY);
      motor2.writeMicroseconds(DELAY);
      motor3.writeMicroseconds(DELAY);
      motor4.writeMicroseconds(DELAY);
      //Imprime el porcentaje de potencia que se está enviando a los motores
      float SPEED = (DELAY-1000)/10;
      Serial.print("\n");
      Serial.println("Motor speed:");
      Serial.print("  ");
      Serial.print(SPEED);
      Serial.print("%");
    }
  }
}
```

Cabe mencionar que la frecuencia con la que la librería Servo.h genera las señales PWM es de 20ms. Una forma para observar que se genera los pulsos en el intervalo correcto y con la duración adecuada se utilizó el software Proteus 8 con la licencia educativa. Esto debido a que cuenta con un osciloscopio digital como se muestra en la figura 24. Como se puede observar la frecuencia de la señal PWM es de 20 ms mientras que el ciclo de trabajo es de 2 ms.

**Figura 24.** ***Visualización de las señales PWM para los motores***




***MPU6050***

La precisión y confiabilidad de los datos proporcionados por el sensor inercial MPU6050 fueron evaluadas meticulosamente. Este sensor integra un acelerómetro y un giroscopio, permitiendo medir la aceleración lineal y la velocidad angular del dron en tres ejes (X, Y, Z).

Para validar el correcto funcionamiento del sensor MPU6050, se emplearon las librerías Simple_MPU6050 y Wire.h. La librería Simple_MPU6050 proporciona funciones específicas para la comunicación y adquisición de datos del sensor, mientras que la librería Wire.h facilita la comunicación I2C necesaria para interactuar con el dispositivo

Para observar los grados de inclinación proporcionados por el sensor MPU6050, se utilizó la librería Simple_mpu, como se ilustra en la figura 25. Esta librería simplifica la obtención de datos de inclinación en grados para los tres ejes (X, Y, Z).

El código presentado en la figura 25 muestra las funciones y estructuras utilizadas para procesar los datos del sensor MPU6050. El resto del código se observa en el anexo B y las características principales se indican a continuación:

- **Función "spamtimer":** Esta función crea un temporizador sin necesidad de utilizar tiempos de espera, lo que optimiza el rendimiento del código.
- **Función para imprimir valores del sensor:** Esta función permite visualizar los valores de aceleración y giroscopio en cada eje.
- **Función para leer y procesar datos del sensor:** Esta función extrae los datos del sensor, los procesa y los convierte en valores de inclinación en grados.
- **Almacenamiento de datos en un array:** Los datos de inclinación en grados se almacenan en un array para su posterior visualización.
- **Impresión de datos para los tres ejes:** Los valores de inclinación en grados se imprimen en la consola para cada uno de los ejes (X, Y, Z).
- **Inicialización de la comunicación I2C y el monitor serial:** En la función void setup(), se inicializa la comunicación I2C con el sensor MPU6050 y se configura el monitor serial para la visualización de datos.

- **Espera de la respuesta del monitor serial:** El código espera una señal del monitor serial antes de comenzar a leer los datos del sensor, permitiendo un control manual del proceso de adquisición de datos.

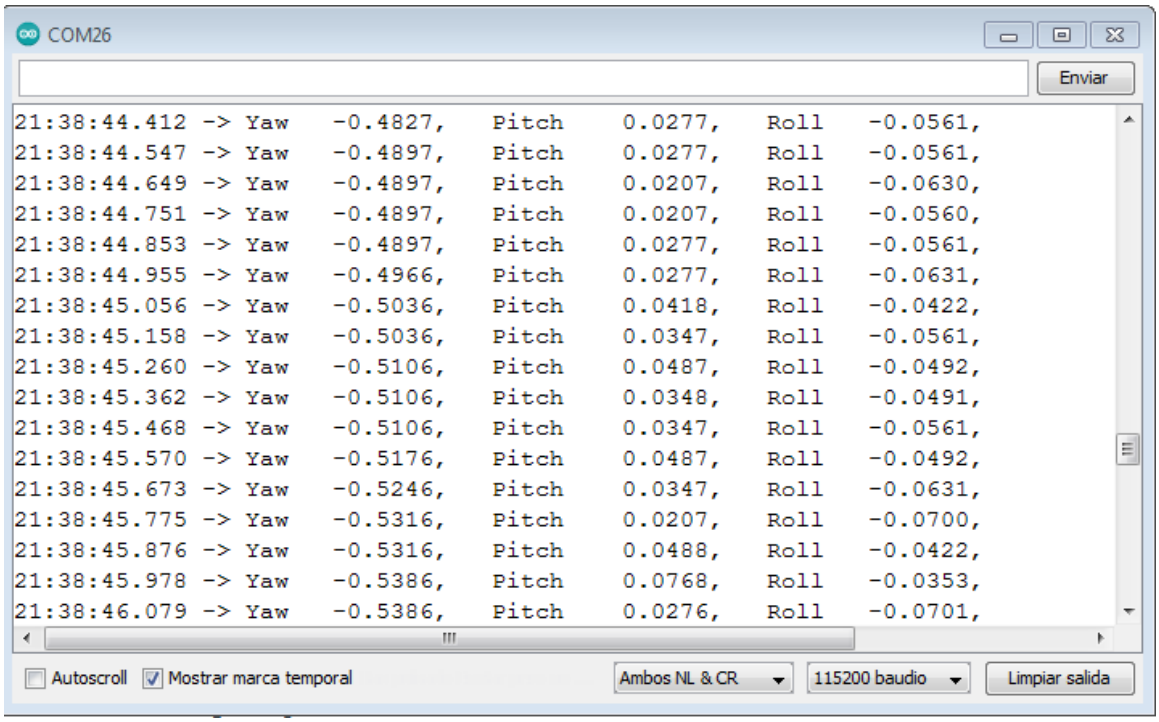
**Figura 25.** ***Programación del MPU6050 con la librería Simple_MPU6050.h***

```
// mostrar_valores funcion que es llamada cada vez que hay datos disponibles desde el sensor
void mostrar_valores (int16_t *gyro, int16_t *accel, int32_t *quat, uint32_t *timestamp) {
  uint8_t SpamDelay = 100;        // demora para escribir en monitor serie de 100 mseg
  Quaternion q;                   // variable necesaria para calculos posteriores
  VectorFloat gravity;            // variable necesaria para calculos posteriores
  float ypr[3] = { 0, 0, 0 };     // array para almacenar valores de yaw, pitch, roll
  float xyz[3] = { 0, 0, 0 };     // array para almacenar valores convertidos a grados
  spamtimer(SpamDelay) {          // si han transcurrido al menos 100 mseg entonces proceder
    mpu.GetQuaternion(&q, quat);    // funcion para obtener valor para calculo posterior
    mpu.GetGravity(&gravity, &q);   // funcion para obtener valor para calculo posterior
    mpu.GetYawPitchRoll(ypr, &q, &gravity);   // funcion obtiene valores de yaw, ptich, roll
    mpu.ConvertToDegrees(ypr, xyz);           // funcion convierte a grados sexagesimales
    Serial.printfloatx(F("Yaw")   , xyz[0], 9, 4, F(",   ")); // muestra rotacion de eje Z, yaw
    Serial.printfloatx(F("Pitch"), xyz[1], 9, 4, F(",   ")); // muestra rotacion de eje Y, pitch
    Serial.printfloatx(F("Roll") , xyz[2], 9, 4, F(",   ")); // muestra rotacion de eje X, roll
    Serial.println();       // salto de linea
  }
}
```

```
void loop() {
// funcion que evalua si existen datos nuevos en el sensor y llama
  mpu.dmp_read_fifo();
}
```

La ejecución de este código permite observar la inclinación detectada por el sensor MPU6050. Los datos impresos en el monitor serial corresponden a los grados de inclinaci**ón** en cada eje (X, Y, Z), como se ilustra en la figura 26. La información de inclinación proporcionada por el sensor MPU6050 resulta crucial para determinar la dirección de la inclinación y, en consecuencia, identificar el eje en el que se realiza la acción. Esta información es fundamental para el posterior control del dron en vuelo.

**Figura 26.** ***Visualización del monitor serial para el sensor MPU6050***




***HC-05***

Para establecer la conexión entre el dron y el mando de forma automática, se empleó el método maestro-esclavo. Este método permite conectar ambos dispositivos sin necesidad de realizar la configuración manualmente.

Para ello, se utilizaron comandos AT para definir los roles de maestro y esclavo en los módulos Bluetooth. El módulo Bluetooth del mando se configuró como maestro, mientras que el módulo Bluetooth del dron se configuró como esclavo.

El proceso de configuración y conexión se realizó utilizando el monitor serial de Arduino y el código que se muestra en la figura 27. Este código incluye los comandos AT

necesarios para establecer la comunicación entre los módulos Bluetooth y verificar la conexión exitosa.

**Figura 27.** ***Configuración HC-05 con comandos AT***

```
#include <SoftwareSerial.h>
SoftwareSerial BT(10,11);
//5856:00:00205C ESCLAVO

void setup() {
  BT.begin(38400);
  Serial.begin(9600);
  Serial.println("LISTO");
}

void loop() {
  if (BT.available()){
    Serial.write(BT.read());
    }

  if (Serial.available()){
    BT.write(Serial.read());
    }
}
```

Para entrar en modo AT primero se debe conectar el módulo HC-05 al Arduino. Después se debe presionar el botón del módulo mientras se lo alimenta con 5V, el led del módulo debería parpadear lentamente indicando que se encuentra en modo AT. Posteriormente, es necesario utilizar los siguientes comandos en el monitor serial para configurar al módulo maestro:

- **AT** (Prueba la comunicación, debe responder con OK)
- **AT+ROLE=1** (Configura el módulo como maestro, debe responder con OK)
- **AT+ADDR?** (Obtén la dirección del módulo maestro, toma nota de esta dirección)

- **AT+CMODE=0** (Configura el módulo para conectarse a una dirección específica, debe responder con OK)
- **AT+INIT** (Inicializa el módulo en modo maestro, debe responder con OK)

Para configurar al módulo HC-05 en modo esclavo se debe seguir el mismo procedimiento que el del maestro. La diferencia radica en los siguientes comandos que se utilizan en el monitor serial:

- **AT** (Prueba la comunicación, debe responder con OK)
- **AT+ROLE=0** (Configura el módulo como esclavo, debe responder con OK)
- **AT+ADDR?** (Obtén la dirección del módulo esclavo, toma nota de esta dirección)
- **AT+UART=9600,0,0** (Configura la velocidad de transmisión (opcional, 9600 es común para Arduino))

Una vez establecida la conexión entre el dron y el mando, se procedió a leer los datos de inclinación del mando. Para ello, se utilizó el código que se muestra en la figura 28. Este código permite interpretar las señales enviadas por el mando y convertirlas en valores de inclinación para cada eje (X, Y, Z).

El código para la lectura de datos de inclinación del mando incluye las siguientes acciones:

1. **Establecimiento de pines RX y TX:** Se definen los pines RX y TX del Arduino que se utilizarán para la comunicación serial con el módulo Bluetooth del mando.

2. **Declaración de variables:** Se declaran variables para almacenar los valores de inclinación en cada eje (roll, pitch y yaw).
3. **Inicialización del monitor serial y la comunicación Bluetooth:** Se inicializa el monitor serial para mostrar los valores de inclinación y se establece la comunicación serial con el módulo Bluetooth del mando.
4. **Lectura de datos en el bucle principal (void loop()):** En el bucle principal del programa (void loop()), se utiliza la función BT_Lectura() para leer los datos enviados por el mando a través del módulo Bluetooth.
5. **Conversión de datos y visualización:** La función BT_Lectura() convierte la cadena de datos recibida del mando en números enteros que representan los valores de inclinación para cada eje. Estos valores se separan y se muestran en el monitor serial para su visualización.

**Figura 28.** ***Programación para la lectura de los datos del mando***

```
#include <SoftwareSerial.h>
SoftwareSerial BT(10, 11);
int valorx, valory, valorz;

void setup() {
  Serial.begin(115200); // Iniciar comunicación serial con el monitor serial
  BT.begin(9600); // Iniciar comunicación serial con el módulo Bluetooth
}

void loop() {
  if (BT.available()) {
    BT_Lectura();

    // Imprimir los valores recibidos en el monitor serial
    Serial.print(valorx);
    Serial.print("\t");
    Serial.print(valory);
    Serial.print("\t");
    Serial.println(valorz);
  }
}


void BT_Lectura(){
    // Leer la cadena de datos enviada por Bluetooth
    String datosRecibidos = BT.readStringUntil('\n');

    // Buscar las posiciones de las comas en la cadena
    int coma1 = datosRecibidos.indexOf(',');
    int coma2 = datosRecibidos.lastIndexOf(',');

    // Extraer los substrings que contienen los valores de las tres variables
    String valor1_str = datosRecibidos.substring(0, coma1);
    String valor2_str = datosRecibidos.substring(coma1 + 1, coma2);
    String valor3_str = datosRecibidos.substring(coma2 + 1);

    // Convertir los substrings a valores enteros
    valorx = valor1_str.toInt();
    valory = valor2_str.toInt();
    valorz = valor3_str.toInt();
  }
```

**Lectura del voltaje de la batería**

El monitoreo del voltaje de la batería es crucial para evitar daños o fallas en las celdas de la batería de LiPo**.** Estas baterías son propensas a sufrir daños si se descargan excesivamente, por lo que es esencial mantener un control preciso del voltaje durante su funcionamiento.

Para realizar el monitoreo del voltaje de la batería, se implementó un divisor de voltaje. Este divisor de voltaje reduce el voltaje de la batería a un nivel compatible con las entradas analógicas del Arduino, permitiendo que el microcontrolador pueda leer y procesar este valor.

El divisor de voltaje se conecta a una entrada analógica del Arduino. La elección de la entrada analógica en este caso es del A0, mientras que la salida digital para el Led es la 12.

La programación que se muestra en la figura 29 permite leer el valor del voltaje de la batería a través del divisor de voltaje y procesarlo para determinar el nivel de carga restante. El código incluye las siguientes funciones:

1. **Lectura del valor analógico:** Se lee el valor analógico de la entrada analógica conectada al divisor de voltaje.
2. **Cálculo del voltaje real de la batería:** Se utiliza una fórmula matemática para calcular el voltaje real de la batería a partir del valor analógico leído y la relación de división del divisor de voltaje.

3. **Evaluación del nivel de carga:** Se compara el voltaje real de la batería con un **umbral predefinido** (9,9 V en este caso). Si el voltaje real de la batería es inferior al umbral, se considera que el nivel de carga es bajo.
4. **Visualización del nivel de carga:** Se utiliza un LED para indicar el nivel de carga de la batería. Si el nivel de carga es bajo, el LED se enciende, alertando al usuario de la necesidad de recargar la batería.

**Figura 29.** ***Programación para la lectura del voltaje de la batería***

```
int sensorPin = A0;    //Entrada del pin para la batería
int ledPin = 13;      //Salida del pin del LED
float sensorValue = 0;  //Variable para guardar el valor del voltaje
float battery_voltage = 7;

void setup() {
  //Declarar salidas
  pinMode(ledPin, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  Serial.println(sensorValue);
  battery_voltage = (((analogRead(0)) * 4.97)/1023);
  Serial.println(battery_voltage);
  if (battery_voltage <= 3.60){
      digitalWrite(ledPin, HIGH);
    }
  else{
      digitalWrite(ledPin, LOW);
    }
  delay(1000);
}
```

***Control PID***

Por último, se realizó el control PID (Proporcional-Integral-Derivativo). Este algoritmo de control automático es fundamental para estabilizar el dron y garantizar un vuelo preciso y maniobrable. A su vez, un aspecto crucial en la programación del dron es la optimización del tiempo de respuesta. Este parámetro determina la rapidez con la que el dron reacciona a los comandos del piloto y a los cambios en el entorno. Un tiempo de respuesta más rápido se traduce en un control más preciso y eficiente del dron.

El control PID es un elemento fundamental para el funcionamiento del dron, ya que permite estabilizarlo y controlarlo con precisión. En este proyecto, el código del control PID no se implementó de forma aislada, sino que se integró con el código de los demás componentes del dron para optimizar su rendimiento.

La base del código del control PID se obtuvo de Arduproject (2024) y se adaptó para su uso en este dron. La principal modificación consistió en la incorporación del control mediante un mando bluetooth para lectura de gestos manuales, lo que permite al usuario maniobrar el dron de manera intuitiva.

La utilización del código de Arduproject como base para el control PID presenta las siguientes ventajas:

- **Código probado y confiable:** El código original ha sido probado y validado en proyectos anteriores, lo que garantiza su confiabilidad y robustez.

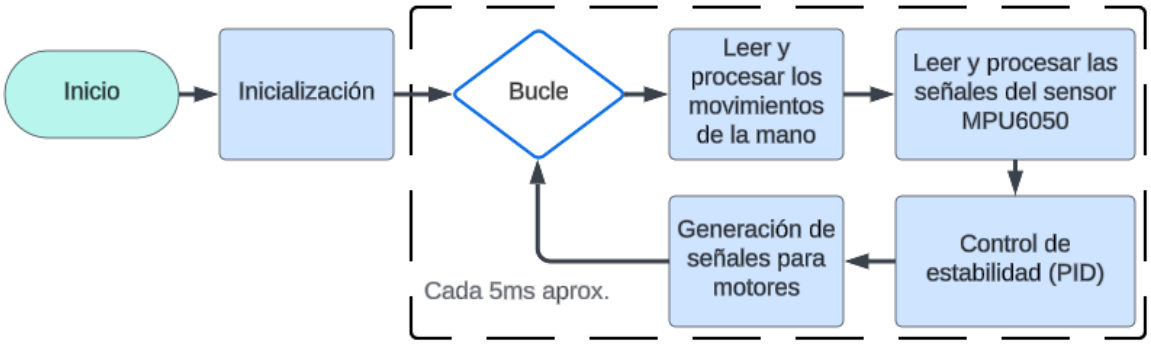
- **Similitud de componentes:** Los componentes utilizados en este dron son similares en tamaño y características a los utilizados en el proyecto original de Arduproject, lo que facilita la implementación del código.

Durante la implementación del control PID, se detectó un problema de generación de pulsos para los motores. Este problema ocasionaba un desfase en los pulsos, lo que provocaba que los motores se detuvieran o vibraran en lugar de girar suavemente.

## Código del dron

Para la programación del dron, se siguió la lógica de programación que se muestra en el diagrama 8. Esta lógica define el flujo de ejecución del programa, incluyendo la lectura de datos de los sensores, el procesamiento de comandos, el control de los motores y la gestión de la energía.

**Diagrama 8.** ***Lógica de programación***




La fase de inicialización del dron se centró en la definición de entradas y salidas**,** la toma inicial de valores del sensor inercial y el establecimiento de la conexión entre el

dron y el mando a distancia. El bucle principal de control del dron se divide en cuatro etapas fundamentales:

**1. Lectura del mando a distancia:**

- Se reciben y decodifican los datos del mando a distancia para obtener los comandos de control del piloto.
- Se registran los valores de inclinación del sensor MPU6050 para determinar la orientación del mando a distancia.
- Los datos del mando a distancia y del sensor inercial se envían al controlador para su procesamiento.

**2. Lectura de la inclinación del dron:**

- Se obtienen los valores de aceleración del sensor inercial MPU6050 del dron.
- Se calculan los ángulos de inclinación (pitch y roll) del dron en base a los valores de aceleración.
- La información de inclinación del dron se envía al controlador para su procesamiento.

**3. Control de estabilidad:**

- Se implementa un controlador PID (Proporcional-Integral-Derivativo) utilizando los valores de inclinación del dron y los comandos del mando a distancia.

- El controlador PID genera señales de control para estabilizar el dron en vuelo y responder a los comandos del piloto.
- Las señales de control estabilizadas se envían al siguiente paso para el control de los motores.

**4. Generación de señales para los motores:**

- Se utilizó la librería “Servo.h” de Arduino para controlar los motores del dron.
- Las señales de control estabilizadas generadas en el paso anterior se convierten en señales PWM (Pulse Width Modulation) adecuadas para los motores.
- Las señales PWM se envían a los ESC (Electronic Speed Controllers) para controlar la velocidad de rotación de los motores.

**Código del mando**

A diferencia del código del dron, la programación del mando es mucho más corta ya que principalmente se basa en la lectura de los datos del sensor MPU6050. Además, del envío de datos mediante bluetooth por lo cual el uso de librerías facilita el código que se observa en la figura 30. El resto del código se indica de mejor forma en el Anexo C.

**Figura 30.** ***Programación del mando***

```
void mostrar_valores (int16_t *gyro, int16_t *accel, int32_t *quat, uint32_t *timestamp) {
  uint8_t SpamDelay = 20;
  Quaternion q;
  VectorFloat gravity;
  float ypr[3] = { 0, 0, 0 };
  float xyz[3] = { 0, 0, 0 };
  spamtimer(SpamDelay) {
    mpu.GetQuaternion(&q, quat);
    mpu.GetGravity(&gravity, &q);
    mpu.GetYawPitchRoll(ypr, &q, &gravity);
    mpu.ConvertToDegrees(ypr, xyz);

    int GiroX = xyz[0];
    int GiroY = xyz[1];
    int GiroZ = xyz[2];

    String datos = String(GiroX) + "," + String(GiroY) + "," + String(GiroZ);
    Serial.println(datos);
    BT.println(datos);
  }
}


void setup() {
  BT.begin(9600);
  Serial.begin(9600);
  pinMode(boton,INPUT);
  pinMode(13,OUTPUT);

  uint8_t val;
#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
  Wire.begin();
  Wire.setClock(400000);
#elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE
  Fastwire::setup(400, true);
#endif
  while (digitalRead(boton)== HIGH);
  while (!digitalRead(boton)== HIGH);
  while (digitalRead(boton)== HIGH);
  mpu.SetAddress(MPU6050_ADDRESS_AD0_LOW).CalibrateMPU().load_DMP_Image();
  mpu.on_FIFO(mostrar_valores);
  BT.println('a');
  digitalWrite(13,HIGH);
}

void loop() {
  mpu.dmp_read_fifo();
}
```

**Ensamble Final del dron**

Una vez que se realizó la programación, se procedió al ensamblaje final del dron. Este proceso involucra la integración del sistema electrónico, el controlador de vuelo y la colocación de la batería cargada en el chasis del dron. Antes de realizar el primer vuelo, se llevaron a cabo verificaciones exhaustivas para asegurar el correcto funcionamiento del dron. Estas verificaciones incluyeron:

- **Colocación de los motores:** Se verificó la correcta instalación de los motores en el chasis del dron, asegurando su alineación y fijación adecuadas.
- **Dirección de giro:** Se comprobó la dirección de giro de cada motor para garantizar un vuelo estable y controlado.
- **Señales PWM:** Se observó que la generación de señales PWM no se desfasará o se interrumpiera.
- **Señales de inclinación del mando:** Se visualizó mediante el monitor serial si la inclinación que existía en el mando sea acorde con lo que se leía en el dron
- **Señales de inclinación del dron:** Se verificó que la inclinación que existía en el dron sea acorde a lo que se mostraba en el monitor serial

Posteriormente, tras superar las verificaciones previas al vuelo, se procedió al montaje de las hélices en los motores. Luego, se encendió el sistema para observar si existen vibraciones o anomalías que puedan afectar el vuelo. Estas pruebas iniciales permiten identificar y corregir posibles problemas antes de realizar el primer vuelo real.

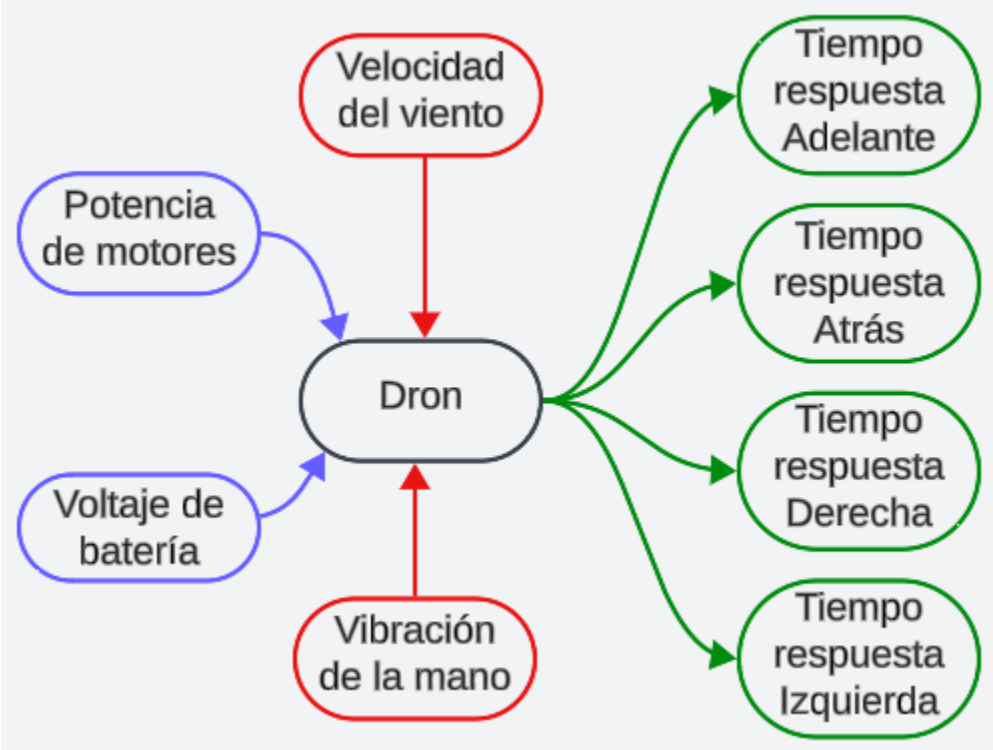
**Evaluación del dron**

Para la evaluación del dron se utilizó una base anclada que permitió un cierto grado de movimiento al dron y a su vez protección ante fallos. Esto permitió que se probara la respuesta generada en el control PID para la estabilización del dron como se muestra en la figura 31.

**Figura 31.** ***Primera prueba para la estabilización del dron***

Tras verificar el correcto funcionamiento del dron, se procedió a evaluar tres aspectos fundamentales: autonomía de la batería, distancia de alcance máxima y control por gestos manuales. Para ello, se implementó un Diseño de Experimentos (DOE) como se muestra en el diagrama 9 con el objetivo de determinar la relación entre la potencia utilizada por los motores y la duración de la batería.

**Diagrama 9.** ***Diseño del DOE para evaluar funcionalidad del dron***




*Nota:* En el diagrama 9 se observa el diseño del DOE para la evaluación del dron en la cual la parte con filo azul indica las variables de entrada. Además, se muestra las variables de ruido con filo rojo y de color verde la respuesta deseada del experimento.

Finalmente, se elaboró la tabla 2 con el objetivo de registrar los valores que se obtuvieron en el DOE. Esto con el fin de evaluar la funcionalidad, respuesta y duración del dron en 10 pruebas en total. En donde, el principal factor a medir fue el tiempo transcurrido entre el gesto de la mano y la reacción del dron. Cabe mencionar que, factores como un ambiente controlado y la vibración de la mano son elementos de ruido a considerar.

**Tabla 2.** ***Evaluación del dron***

| | Pot. del Mot. (%) | Voltaje inicial batería (V) | Voltaje final batería (V) | Tiempo respuesta Adelante (s) | Tiempo respuesta Atrás (s) | Tiempo respuesta Derecha (s) | Tiempo respuesta Izquierda (s) | Duración batería (min) |
|---|---|---|---|---|---|---|---|---|
| 1 | 50 | | | | | | | |
| 2 | 50 | | | | | | | |
| 3 | 50 | | | | | | | |
| 4 | 50 | | | | | | | |

## Resultados

En primer lugar, se realizó la selección de componentes para el ensamblaje del dron y su control remoto por gestos manuales. Los componentes seleccionados se detallan en la tabla 3.

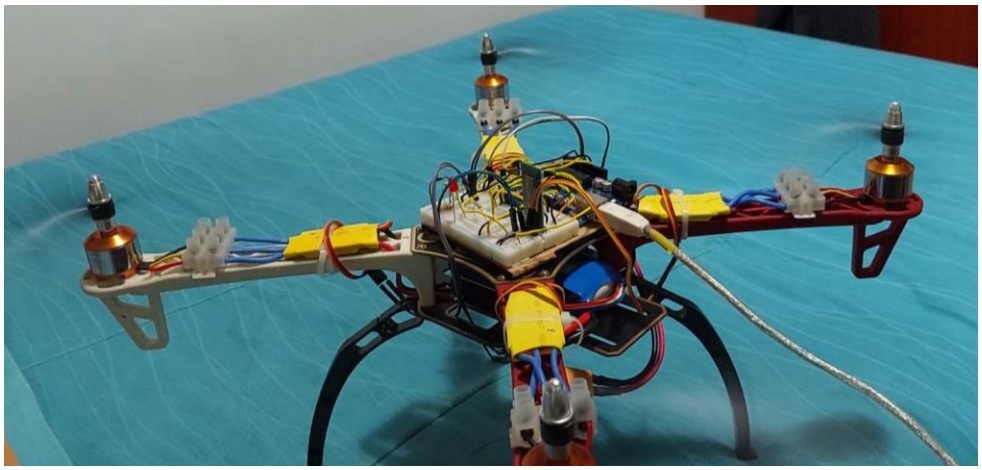
**Tabla 3.** ***Listado de componentes del dron y del mando***

| Nombre del componente | Cantidad de componentes |
|---|---|
| DJI Frame 450mm | 1 |
| Motor Brushless A2212 13T 1000KV | 4 |
| Simonk ESC 30A | 4 |
| Mgsystem hélice 10x45 | 4 |
| Arduino Uno | 1 |
| MPU6050 | 2 |

| | |
|---|---|
| Módulo Bluetooth HC-05 | 2 |
| Batería Li-Po 3S 11.1V 4000mAh 60C | 1 |
| Resistencia de 1K | 3 |
| Resistencia de 500 ohm | 1 |
| Resistencia de 220 ohm | 1 |
| Diodo 1N4001 | 1 |
| Led rojo de 5mm | 1 |
| Protoboard | 2 |
| Arduino Nano | 1 |
| Batería LiPo 2S 500mAh 35C | 1 |
| Pulsador 6x6x5 mm | 1 |
| Interruptor deslizable pequeño 3 pines | 1 |
| Tira de velcro de 24 cm | 1 |

Posteriormente, se ensambló un dron con las características necesarias para el vuelo autónomo y la facilidad de manipulación, como se observa en la figura 32. El diseño prioriza la simplicidad en el reemplazo de componentes, utilizando borneras para la conexión a motores y una protoboard para la parte electrónica junto con el Arduino Uno. Siguiendo la misma filosofía de simplicidad y facilidad de uso, se diseñó un control remoto compacto y amigable para el usuario.

**Figura 32.** ***Dron ensamblado***



Así mismo, el sistema electrónico se dividió en dos subsistemas principales:

- **Subsistema de alimentación y motores:** Encargado de suministrar energía a los motores y controlar su velocidad mediante los ESC.
- **Subsistema de control y comunicación:** Responsable de la adquisición de datos del sensor inercial, el procesamiento de la información y la comunicación con el usuario a través del módulo Bluetooth.

Durante la calibración y programación de los motores, se identificaron tres tipos de problemas principales:

1. **Rigidez**: El motor presenta dificultad para girar. Se identifica al intentar girarlo manualmente. Si la hélice se detiene casi de inmediato, hay rigidez. La solución es reemplazar el motor, ya que este problema genera mayor temperatura y riesgo de quemar el ESC.

2. **Desincronización**: Se presenta un sonido y tiempo de respuesta diferentes en los motores durante la inicialización de la calibración. Esto provoca vibración y giro a destiempo del motor. Se recomienda reemplazar el ESC, ya que podría no ser de la misma marca que los demás.
3. **Baja potencia**: El motor no gira al enviar una potencia menor del 20% durante la calibración, aunque se observe una leve vibración. Al enviar un valor superior al 20%, el motor funciona, pero el dron no debería volar con este problema, ya que se considera anómalo. La solución principal es reemplazar el ESC.

En cambio, el subsistema de control y comunicación presenta un problema de latencia con respecto al usuario. Esto debido a, la velocidad de comunicación que tiene el módulo Bluetooth. La programación de este subsistema se detalla en los anexos B y D, incluyendo la configuración del módulo HC-05 y la implementación del control PID para la estabilización del dron.

El diagrama 10 ilustra el funcionamiento principal del sistema electrónico del dron. El proceso es el siguiente:

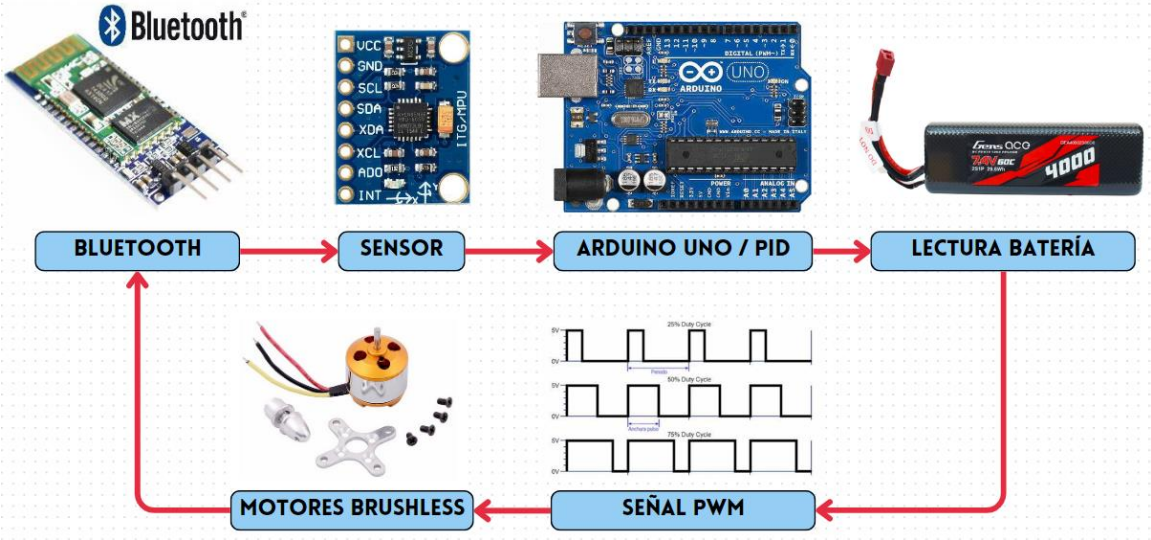
1. Se obtienen los valores del mando para conocer la dirección de movimiento deseada.
2. Estos valores se procesan en el sensor inercial para indicar la dirección de inclinación del dron.

3. La información procesada se utiliza para controlar los motores y realizar el movimiento deseado del dron.

Posteriormente, tras recibir la información del subsistema de control y comunicación, el dron realiza las siguientes acciones:

1. **Cálculo de potencia de motores con PID:** El controlador PID calcula la potencia necesaria para cada motor en función de la información del sensor inercial y la dirección de movimiento deseada. El objetivo es que el dron se incline de forma estable y precisa.
2. **Comprobación de voltaje de la batería:** Se verifica el voltaje de la batería para evitar su descarga excesiva. En caso de detectar niveles críticos, se toman medidas preventivas, como reducir la potencia de los motores y avisar al usuario.
3. **Generación de señal PWM:** Se genera una señal PWM para controlar la velocidad de cada motor. La anchura del pulso en la señal PWM determina la potencia del motor correspondiente. El valor de la señal PWM depende directamente de los cálculos realizados anteriormente por el control PID.

**Diagrama 10.** ***Diagrama de funcionamiento del dron***




El sensor MPU6050 se utiliza para estabilizar el dron, pero no cuenta con magnetómetro, por lo que el giro en el eje Z (rotación sobre su propio eje) no es preciso. Los ejes de rotación principales (X e Y), como se observa en la Figura 33, permiten al dron avanzar, retroceder y desplazarse de izquierda a derecha. Para lograr una estabilización más rápida y precisa, se implementó un control PID.

Un divisor de voltaje monitorea el voltaje de la batería para controlar su tiempo de duración. Se mide el voltaje entre 0V y 5V (equivalente a valores de 0 a 1023) para avisar al usuario cuando la batería esté a punto de agotarse. Esta información se visualiza en el monitor serial, permitiendo al usuario conocer el tiempo de vuelo restante.

Por otro lado, se fabricó un prototipo funcional del sistema de manejo del dron por gestos manuales, donde se usó el sensor de movimiento inercial MPU6050 para

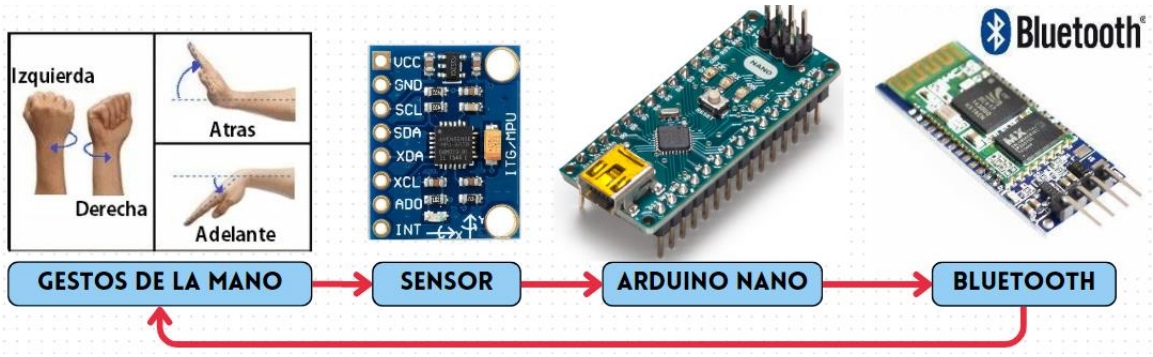
detectar la orientación y movimiento de la mano del usuario. El sistema se basa en la inclinación de la mano para controlar la dirección del dron, mientras que el potenciómetro se utiliza para regular la velocidad de los motores.

Sin embargo, debido a la ausencia de un magnetómetro en el sensor MPU6050, el giro en el eje Z no se controla de manera precisa. Inicialmente, se consideró utilizar este eje para controlar la velocidad de ascenso y descenso del dron, pero esta implementación fue descartada debido a la complejidad y poca precisión del control.

Como alternativa, se implementó el control de la velocidad de los motores mediante un potenciómetro, lo que permite un control más preciso y confiable. Esta solución permite controlar al dron con la inclinación de la mano en dos ejes (X e Y) para realizar movimientos de avance, retroceso, desplazamiento lateral y giros en el plano horizontal. El funcionamiento principal del mando se puede observar en el diagrama 11.

El sensor MPU6050 mide la inclinación de la mano y envía este valor al dron mediante el módulo HC-05.en donde el sensor MPU6050 es el encargado de medir la inclinación de la mano. Para posteriormente, enviar este valor mediante el módulo HC-05 al dron.

**Diagrama 11.** ***Diagrama de funcionamiento del mando***

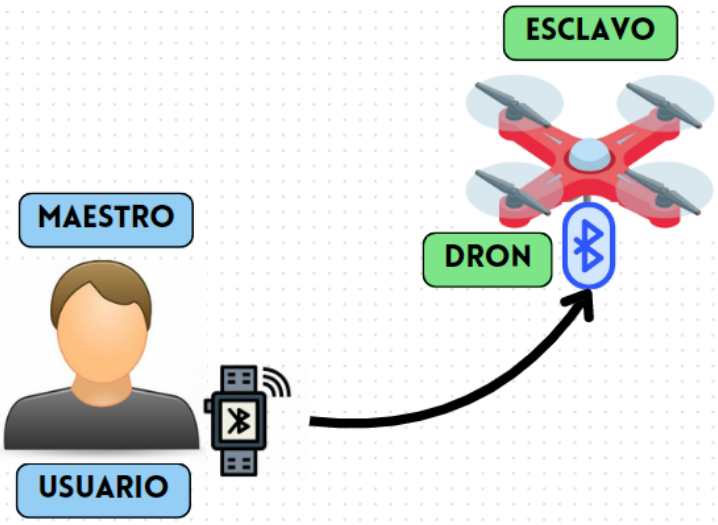



Ahora bien, se evaluó el funcionamiento del dron para validar su usabilidad pedagógica mediante pruebas de vuelo y control por gestos manuales. Para ello, se fijaron las patas del dron a un par de pesas para evitar movimientos erráticos y facilitar la observación de su comportamiento como se puede visualizar en la figura 33.

**Figura 33.** ***Anclaje de las patas del dron.***

El funcionamiento principal del dron se puede visualizar de mejor forma en el diagrama 12, en donde se indica la configuración de la comunicación del sistema general. En donde, el envío de datos se realiza en un solo sentido es decir que el mando se encarga de enviar valores sin necesidad de recibir señales. En este mismo sentido, el dron se encarga de recibir datos mas no de enviarlos.

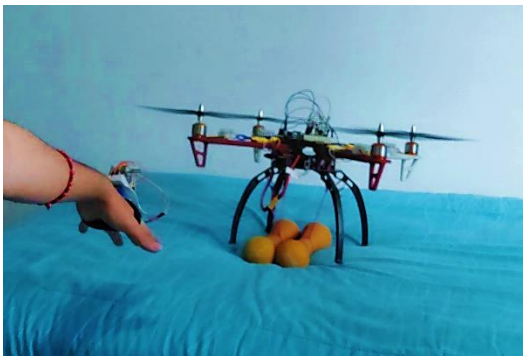
**Diagrama 12.** ***Funcionamiento general del dron***




*Nota:* El diagrama presenta la comunicación que se realiza en el dron donde el medio de envío de datos es por Bluetooth. Cabe mencionar, que el usuario solo se encarga de enviar valores de inclinación al dron mientras que este se especializa en recibir y leer dichos datos para su desplazamiento.

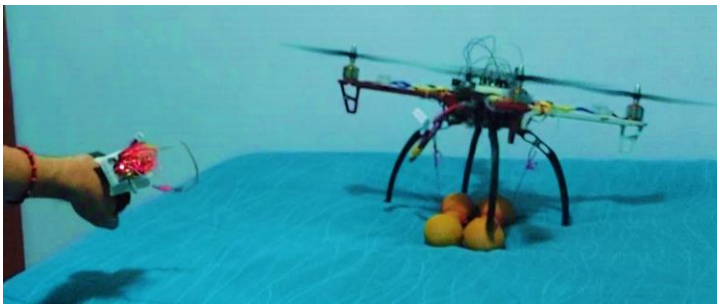
Por otro lado, las pruebas que se realizaron para conocer si el sistema por control de gestos manuales funciona se puede observar en las siguientes figuras 34, 35, 36, 37.

**Figura 34.** ***Prueba 1 del dron desplazamiento hacia adelante***



*Nota*: La figura presenta la prueba realizada para observar el comportamiento del dron cuando se inclina la mano hacia abajo por lo que indica al dron desplazarse adelante. En este caso se puede visualizar que los motores traseros presentan mayor potencia que los delanteros.

**Figura 35.** ***Prueba 2 del dron desplazamiento hacia la derecha***



*Nota*: La figura presenta la prueba realizada para observar el comportamiento del dron cuando se inclina la mano hacia la derecha por lo que indica al dron desplazarse en esa dirección. En este caso se puede visualizar que los motores izquierdos presentan mayor potencia que los derechos.

**Figura 36.** ***Prueba 3 del dron desplazamiento hacia la izquierda***

*Nota*: La figura presenta la prueba realizada para observar el comportamiento del dron cuando se inclina la mano hacia la izquierda por lo que indica al dron desplazarse en esa dirección. En este caso se puede visualizar que los motores derechos presentan mayor potencia que los izquierdos.

**Figura 37.** ***Prueba 4 del dron desplazamiento hacia la atrás***

*Nota*: La figura presenta la prueba realizada para observar el comportamiento del dron cuando se inclina la mano hacia arriba por lo que indica al dron desplazarse atrás. En este caso se puede visualizar que los motores delanteros presentan mayor potencia que los traseros.

Los resultados de la evaluación se pueden observar en la tabla 4 en donde se muestran los valores de tiempo de vuelo y la respuesta promedio a los gestos manuales. Se puede observar que el tiempo de vuelo del dron es de 15 minutos con una potencia del 50%. Mientras que, el tiempo de vuelo del dron al 80% es de 9 minutos.

**Tabla 4.** ***Evaluación del tiempo de respuesta del dron***

| | Pot. del Mot. (%) | Voltaje inicial batería (V) | Voltaje final batería (V) | Tiempo respuesta Adelante (s) | Tiempo respuesta Atrás (s) | Tiempo respuesta Derecha (s) | Tiempo respuesta Izquierda (s) | Duración batería (min) |
|---|---|---|---|---|---|---|---|---|
| 1 | 50 | 12.62 | 10.55 | 2.5 | 2.5 | 2.5 | 2.5 | 15,04 |
| 2 | 50 | 12,52 | 10,02 | 2,7 | 2,5 | 2,6 | 2,5 | 15,26 |
| 3 | 50 | 12,50 | 10,05 | 2,4 | 2,3 | 2,8 | 2,3 | 14,48 |
| 4 | 50 | 12,49 | 10,12 | 2,6 | 2,4 | 2,7 | 2,2 | 14,24 |
| 5 | 50 | 12,50 | 10,17 | 2,4 | 2,5 | 2,9 | 2,4 | 15,39 |
| 6 | 80 | 12,50 | 10,33 | 1,4 | 1,8 | 2,1 | 1,4 | 9,03 |
| 7 | 80 | 12,50 | 10,31 | 1,3 | 1,5 | 2,2 | 1,6 | 9,12 |
| 8 | 80 | 12,50 | 10,34 | 1,2 | 1,6 | 2,4 | 1,4 | 9,05 |
| 9 | 80 | 12,49 | 10,30 | 1,3 | 1,5 | 2,1 | 1,5 | 9,08 |
| 10 | 80 | 12,50 | 10,33 | 1,4 | 1,2 | 1,9 | 1,6 | 9,02 |

## Discusión de Resultados

El resultado de la determinación de los requisitos técnicos y funcionales del dron se puede observar en la tabla 2. En donde, se muestra la lista de componentes que se utilizó para ensamblar un dron controlado por gestos manuales. Como menciona Manrique (2022), la selección de los componentes es fundamental para que exista compatibilidad entre los diferentes sistemas. Así mismo, Llaguano (2018) indica que conocer los componentes del dron facilita su instalación sobre todo si se realiza como controlador de vuelo un Arduino Uno. Cabe mencionar que, este resultado concuerda con el que presenta Llaguano (2018), ya que la mayoría de componentes se asemeja. Sin embargo, el mando de control del dron difiere ya que en este proyecto se utilizó el microcontrolador Arduino Nano con un giroscopio para la lectura de los gestos de la mano.

La elección de componentes de fácil montaje, como el chasis o los demás que se utilizaron en este proyecto resultan fundamentales en un contexto educativo. Esto debido a que, permite a los estudiantes ensamblar, modificar y configurar el dron de manera autónoma fomentando la comprensión de los principios básicos de la electrónica, la mecánica y la programación. De igual forma, la implementación de borneras para la conexión de los motores con sus respectivos controladores facilita el remplazo en el caso de que existan fallas o daños en dichos componentes. Además, el uso de Arduino como controlador de vuelo proporciona al estudiante un entorno familiar en el cual experimentar y realizar diversas pruebas con respecto a la programación.

Por otro lado, el diagrama 10 ilustra de manera más detallada el diseño del sistema electrónico y de comunicación. Este diseño presenta similitudes con los sistemas empleados en drones convencionales. Como señala Saltos (2018), la comunicación Bluetooth, operando a una frecuencia de 2.4 GHz, facilita la telemetría inalámbrica entre dispositivos.

Sin embargo, es importante destacar que, según Lago (2017), el control de drones comúnmente se realiza mediante señales de radio a 2.4 GHz, las cuales envían pulsos modulados con las órdenes del mando. Esta diferencia con respecto a los hallazgos de Lago (2017) se debe principalmente al enfoque de este estudio. Mientras que Lago se centró en el vuelo autónomo mediante sensores, este proyecto está orientado a un control manual del dron a través de un mando. Esta distinción explica la elección de la tecnología Bluetooth para la comunicación en este sistema.

Respecto a la eficiencia del sistema electrónico y de comunicación, el uso de Arduino como controlador de vuelo y Bluetooth para la transmisión de datos presenta tanto ventajas como limitaciones. Arduino, por su naturaleza de plataforma abierta y de bajo costo, ofrece una gran flexibilidad para la implementación de algoritmos de control y la integración de nuevos sensores. Sin embargo, su capacidad de procesamiento limitado puede ser un cuello de botella en aplicaciones que requieran cálculos en tiempo real más complejos, siendo el caso si se implementa redes neuronales. Por otro lado, Bluetooth, aunque ofrece una conexión sencilla y de bajo costo, presenta limitaciones en

términos de alcance y tasa de transferencia de datos, lo cual puede afectar la estabilidad del vuelo en condiciones adversas.

Ahora bien, la figura 35 presenta el prototipo funcional del sistema de control del dron mediante gestos manuales. En este sistema, la inclinación de la mano determina la dirección de movimiento del dron. Como explica Calleja (2016), los ejes de inclinación principales para el control de drones son Roll, Pitch y Yaw, además de la potencia de los motores (Throttle). Milán (2017) señala que actualmente los drones presentan mandos de radio frecuencia para indicar el movimiento en los diferentes ejes de inclinación. Sin embargo, en este proyecto, se propone una alternativa más accesible y educativa, reemplazar los controles convencionales por un mando construido con módulos educativos como Arduino. Para ello, se plantea el uso de sensores inerciales compatibles con Arduino, donde se busca desarrollar un mando capaz de controlar el dron de manera efectiva. La idea central es ofrecer una solución más económica y didáctica para la implementación de un sistema de control gestual.

En este caso, el control por gestos representa una herramienta educativa valiosa, especialmente en el ámbito universitario. Al utilizar Arduino como plataforma de desarrollo, los estudiantes pueden ampliar sus conceptos de electrónica, programación, control de sistemas y procesamiento de señales. Además, el control gestual puede servir como base para proyectos más avanzados, como la creación de interfaces hombre-máquina intuitivas o la implementación de sistemas de asistencia para personas con discapacidad. Sin embargo, es importante destacar que la implementación de un sistema

de control gestual robusto requiere una comprensión profunda de los principios de la robótica e incluso visión artificial de ser el caso.

Por otra parte, las pruebas de vuelo se realizaron en un entorno controlado para evitar daños al prototipo. Se evaluaron aspectos como la estabilidad, la respuesta a los comandos y la duración de la batería. Si bien se obtuvieron resultados prometedores, el prototipo aún presenta algunas limitaciones, como la vibración del motor 1 y la falta de estabilidad en el eje Yaw. Estas limitaciones sugieren que se requiere un mayor ajuste de los parámetros del sistema de control y una calibración más precisa de los sensores.

A pesar de estas limitaciones, este proyecto demuestra la viabilidad de construir un dron controlado por gestos utilizando componentes comerciales junto con software libre, y sienta las bases para futuros desarrollos en el campo de la mecatrónica educativa. Además, la adquisición local de los componentes favoreció al tiempo de ensamblaje del dron frente a la alternativa de importar los materiales. Así mismo, se evitó el uso de impresión 3D, el cual es un proceso más complejo y que requiere mayor tiempo para un primer prototipo. Esto debido a que, el enfoque principal es el manejo de un dron mediante gestos manuales para uso pedagógico. Sin embargo, no se descarta la posibilidad de la creación de modelos de chasis más eficientes a través de impresión 3D, lo que permitiría una mayor personalización y optimización del diseño.

## Conclusiones

En conclusión, el desarrollo de un dron controlado por gestos manuales representa un avance significativo en la educación técnica de la carrera de mecatrónica. Este proyecto ha demostrado que es posible crear un dispositivo de aprendizaje accesible y atractivo, que integra múltiples disciplinas STEM y fomenta el desarrollo de habilidades prácticas y teóricas.

Al utilizar componentes de bajo costo y fácil acceso, como Arduino y el sensor MPU6050, se ha logrado un prototipo funcional que puede ser replicado y mejorado por otros estudiantes. El sensor MPU6050 ha demostrado ser eficaz para obtener la inclinación en los ejes roll y pitch, permitiendo un control preciso del dron. Sin embargo, se ha evidenciado una deficiencia en la medición del eje yaw, lo que sugiere la necesidad de complementar el sistema con un magnetómetro en futuras investigaciones.

Si bien el Bluetooth ha demostrado ser una opción viable para la comunicación en interiores, su limitado alcance restringe las aplicaciones en ambientes abiertos o muy extensos. La implementación de tecnologías de comunicación más robustas, como sistemas de radiofrecuencia, permitiría ampliar el rango de operación del dron y explorar nuevas aplicaciones en entornos industriales y comerciales.

Por otro lado, las pruebas de vuelo realizadas en un entorno controlado han permitido evaluar el desempeño del prototipo y han identificado áreas de mejora. En este caso, la optimización del software de control y la implementación de algoritmos de

estabilización más robustos como redes neuronales o directamente inteligencia artificial. Aunque aún no se han realizado pruebas con estudiantes, se espera que este prototipo sirva como base para futuros proyectos educativos.

En conclusión, este proyecto sienta las bases para el desarrollo de sistemas de control gestual más sofisticados y versátiles. Ya que, la aplicación del control por gestos en la industria podría revolucionar la forma en que interactuamos con las máquinas, facilitando tareas complejas y mejorando la eficiencia de los procesos. Por ejemplo, los drones controlados por gestos podrían utilizarse en la inspección de infraestructuras, la agricultura de precisión o incluso en búsqueda y rescate. En este caso, al facilitar el acceso a tecnologías avanzadas, se contribuye a la formación de futuros ingenieros, y se abren nuevas posibilidades para la innovación en diversos sectores.

## Recomendaciones

En primer lugar, es fundamental mejorar la precisión y la velocidad de respuesta del sistema de control por gestos. Esto puede lograrse mediante la optimización de los algoritmos de procesamiento de señales y la implementación de filtros avanzados para reducir el ruido. Además, se sugiere ampliar la gama de gestos reconocidos para permitir un control más versátil y preciso del dron.

En cuanto a la seguridad y fiabilidad del sistema, se recomienda implementar redundancias en los componentes críticos, como baterías de respaldo y sistemas de emergencia. Asimismo, la incorporación de protectores de hélices y estructuras de

absorción de impactos puede minimizar los daños en caso de colisiones. Para extender el tiempo de vuelo, se sugiere utilizar baterías de mayor capacidad y optimizar la gestión de energía.

Para mejorar la autonomía y la seguridad del dron, se propone la integración de una variedad de sensores, como ultrasónicos, de altura, de presión y de temperatura. Estos sensores permitirán al dron detectar obstáculos, mantener una altura de vuelo constante y adaptarse a diferentes condiciones ambientales.

La implementación de sensores adicionales, como magnetómetros, puede mejorar significativamente la precisión del sistema de control por gestos. Al complementar la información proporcionada por el acelerómetro y el giroscopio, se puede obtener una estimación más precisa de la orientación del dron. Además, la integración de redes neuronales y algoritmos de inteligencia artificial puede permitir al sistema adaptarse a diferentes estilos de vuelo y reconocer patrones más complejos.

Además, para evaluar el desempeño del dron e identificar áreas de mejora, se sugiere realizar pruebas exhaustivas en diversos entornos. Es importante implementar un sistema de recopilación y análisis de datos para obtener información detallada sobre el tiempo de vuelo, la estabilidad, la precisión de control y el consumo de energía. Adicionalmente, para facilitar el aprendizaje y el uso del dron, es fundamental desarrollar una interfaz de usuario intuitiva y amigable. La implementación de sistemas de retroalimentación en tiempo real, como indicadores visuales o auditivos, puede proporcionar al usuario información clara sobre el estado del dron.

Dentro de la parte educativa se plantea crear manuales detallados y guías de usuario que expliquen paso a paso la operación y el mantenimiento del dron, así como la interpretación de gestos. Además, se podría desarrollar programas de capacitación específicos para profesores y estudiantes, incluyendo módulos prácticos sobre el manejo y la programación de drones. Así mismo, se podría fomentar proyectos interdisciplinarios que integren el uso del dron en áreas como la robótica, la programación, la electrónica e incluso simulación. Para ello, se podría organizar competencias o retos educativos que promuevan la creatividad y la innovación entre los estudiantes, utilizando el dron como herramienta principal.

Se recomienda implementar al dron como parte de proyectos de laboratorio y asignaturas prácticas dentro del currículo de la carrera de mecatrónica como lo es sistemas de control, electrónica de potencia o microcontroladores. Dentro de esto, se podría realizar talleres de formación para capacitar a estudiantes junto con profesores en el manejo y programación del dron. Incluso, se podría realizar esto como parte del club de robótica de la universidad e incentivar a los estudiantes para la integración de otras tecnologías. Para ello, se recomienda realizar pruebas constantes con los estudiantes para recoger su retroalimentación y ajustar el diseño junto con las funcionalidades del dron en base a las opiniones recolectadas.

Además, es importante mencionar que se debería revisar constantemente actualizaciones sobre las normativas y regulaciones vigentes en cuanto al uso de drones en el ámbito educativo y adaptarse a los cambios para asegurar el cumplimiento legal. Y

si es el caso, asegurarse de que toda la documentación necesaria y los permisos correspondientes estén en orden para evitar problemas legales u operacionales.

**Lista de referencias**

Alizada, T., Sabziev, E., & Heydarov, N. (2023). Improving the Efficiency of the MPU-6050 Sensor Module for Inertial Drone Navigation. *Modeling Control and Information Technologies*, *6*, 83–86. https://doi.org/10.31713/mcit.2023.023

Arduproject. (2018, Julio 12). Arduino Drone. *ArduProject.es*. https://arduproject.es/material-necesario-y-montaje-de-los-componentes-hardware/

Bai, O., Chu, H., Liu, H., & Hui, G. (2021). Drones in Education: A Critical Review. *Turkish Journal of Computer and Mathematics Education*, *12*(11), 1722–1727. https://doi.org/10.17762/turcomat.v12i11.6107

Baustista, J., Hernández, R., Orcos, L., & Magreñan, Á. (2020). Robótica educativa y drones en la interacción humano-robot (IHR) en el marco del modelo STEM. *Tecnologías Educativas y Estrategias Didácticas*, 1042–1051. https://dialnet.unirioja.es/servlet/articulo?codigo=7787789

Berarache, B. (2023). *Manoeuvring drone (Tello Talent) using eye gaze and or fingers gestures*. https://upcommons.upc.edu/handle/2117/395857

Bermejo, D. (2020). Navegación autónoma de un dron en entornos de interior mediante mapas topológicos visuales con aprendizaje profundo de redes neuronales convolucionales para la detección de objetos y la descripción de escenas mediante un lenguaje de tipo natural. *E.T.S. de Ingenieros Informáticos (UPM)*. https://oa.upm.es/63648/

Bermúdez, A., Casado, R., Fernández, G., Guijarro, M., & Olivas, P. (2019). Drone challenge: A platform for promoting programming and robotics skills in K-12 education. *International Journal of Advanced Robotic Systems*, *16*(1). https://doi.org/10.1177/1729881418820425

Budiyanto, A., Ramadhan, M. I., Burhanudin, I., Triharminto, H. H., & Santoso, B. (2021). Navigation control of Drone using Hand Gesture based on Complementary Filter Algorithm. *Journal of Physics: Conference Series*, *1912*(1). https://doi.org/10.1088/1742-6596/1912/1/012034

Calleja Hernández, J. M. (2016). *Control remoto de un drone* [Universitat Politècnica de Catalunya]. https://upcommons.upc.edu/handle/2117/88658

Cañas, J. M., Martín-Martín, D., Arias, P., Vega, J., Roldán-álvarez, D., García-Pérez, L., & Fernández-Conde, J. (2020). Open-Source Drone Programming Course for Distance Engineering Education. *Electronics 2020, Vol. 9, Page 2163*, *9*(12), 2163. https://doi.org/10.3390/ELECTRONICS9122163

Çetin, E. (2023). Counter a drone via deep reinforcement learning. *TDX (Tesis Doctorals En Xarxa)*. https://doi.org/10.5821/DISSERTATION-2117-400823

Chico, A. (2022). *Integración de un sistema de reconocimiento de gestos de la mano basado en el sensor Myo Armband con el simulador de robots Coppeliasim para el control de un manipulador virtual de 6 grados de libertad.* http://bibdigital.epn.edu.ec/handle/15000/22259

Chowdhury, A. R., Dalal, A., & Sen, S. (2020). *Accelerography: Feasibility of Gesture Typing using Accelerometer*. http://arxiv.org/abs/2003.14310

Cruz León, J. de la. (2021). *Diseño y fabricación de un dron mediante diseño generativo e impresión 3D*. https://uvadoc.uva.es/handle/10324/47245

Cubillos, E. (2016). *DISEÑO Y DESARROLLO DE UNA PLATAFORMA DE INTEGRACIÓN DIGITAL ENTRE UN DRONE Y UN DISPOSITIVO DE RECONOCIMIENTO GESTUAL*. https://repositorio.uptc.edu.co/handle/001/1715

de Las Heras Molins, P. (2019). *Disseny i construcció d'una maqueta de control basada en un dron*. https://upcommons.upc.edu/handle/2117/174075

DelPreto, J., & Rus, D. (2020). Plug-and-play gesture control using muscle and motion sensors. *ACM/IEEE International Conference on Human-Robot Interaction*, 439–448. https://doi.org/10.1145/3319502.3374823

DJI. (2015). *Flame Wheel 450*. DJI. https://dl.djicdn.com/downloads/flamewheel/en/F450_User_Manual_v2.2_en.pdf

Dong, Z., Li, F., Ying, J., & Pahlavan, K. (2020). A Model-Based RF Hand Motion Detection System for Shadowing Scenarios. *IEEE Access*, *8*, 115662–115672. https://doi.org/10.1109/ACCESS.2020.3004513

Fiesco, J. P. (2022). *Diseño y validación de una herramienta de optimización para la selección de componentes de drones de carreras* [Universitat Politècnica de València]. https://riunet.upv.es/handle/10251/186843

Fouché, G. J., & Malekian, R. (2018). Drone as an autonomous aerial sensor system for motion planning. *Measurement*, *119*, 142–155. https://doi.org/10.1016/J.MEASUREMENT.2018.01.027

Gio, N., Brisco, R., & Vuletic, T. (2021). Control of a drone with body gestures. *Proceedings of the Design Society*, *1*, 761–770. https://doi.org/10.1017/pds.2021.76

Jeong, H., Lin, X., Shaker, A., Cardenas, I., Kim, J.-H., Robins, J., & Chi, Y.-S. (2023, October 5). *Design of a Low-Cost, Open Source, and Open Hardware Educational Drone*. https://doi.org/10.5038/WJYS4081

Jiménez, D. (2020). *Programa de robótica educativa basado en drones con un enfoque en educación STEM*. http://hdl.handle.net/1992/48517

Jimenez, J. (2019). *Using drones for educational purposes*. https://upcommons.upc.edu/handle/2117/173834

Juárez Arnau, Á. (2024). *Diseño y construcción de un dron*. https://upcommons.upc.edu/handle/2117/404910

Kangunde, V., Jamisola, R. S., & Theophilus, E. K. (2021). A review on drones controlled in real-time. *International Journal of Dynamics and Control*, *9*(4), 1832–1846. https://doi.org/10.1007/S40435-020-00737-5/FIGURES/16

Lago, J. (2017). *Diseño y control de un dron con evasión de obstáculos*. http://hdl.handle.net/2117/111801

LLaguno, I. (2018). *Movimiento autónomo de un dron entre dos puntos en espacios interiores*. http://hdl.handle.net/10810/29451

Llordés, I. (2016). *Estudi i anàlisi sobre la implantació dels drons com a una mesura antiincendis*. https://upcommons.upc.edu/handle/2117/97935

Manrique, T. (2022). *Diseño del chasis de un dron de cuatro motores para contribuir en las tareas de logística de la empresa Casa del Ruliman del Ecuador S.A.* [Quito: Universidad Tecnològica Indoamèrica]. https://repositorio.uti.edu.ec//handle/123456789/4896

Martínez, J. (2023). *Control de un brazo robótico controlado con gestos manuales*. https://digibuo.uniovi.es/dspace/handle/10651/68806

Masoud, M., Jaradat, Y., Farhood, M., & Almedalaleh, A. (2016). *On Implementing a Low Cost Quadcopter Drone with Smartphone Control*. https://www.researchgate.net/publication/301787539_On_Implementing_a_Low_Cost_Quadcopter_Drone_with_Smartphone_Control

Mendez, J. E. (2015). *Implementación de un dron cuadricóptero con Arduino*. https://hdl.handle.net/2454/19208

Milán, M. (2017). *Control de vuelo de un dron utilizando un acelerómetro*. https://upcommons.upc.edu/handle/2117/104981

Molina, A., Villoria, M., & García, F. (2020). SISTEMA AUTOMÁTICO PARA EL CONTROL DE UN BRAZO ROBÓTICO UTILIZANDO UNA BANDA DE CONTROL POR GESTOS Y TÉCNICAS DE APRENDIZAJE DE MÁQUINA. *Encuentro Internacional de Educación En Ingeniería*, 1–9. https://doi.org/10.26507/PONENCIA.844

Montejo Fernández, G. (2023). *Desenvolupament d'un sistema de comunicacions per operacions remotes amb drons*. https://upcommons.upc.edu/handle/2117/403491

Múnera, S. (2022). *UAVoxWeb: Arquitectura multicomponente para el apoyo a rutas de vuelo con drones*. http://hdl.handle.net/1992/62548

Natarajan, K., Nguyen, T.-H. D., & Mete, M. (2018). *Hand Gesture Controlled Drones: An Open Source Library*. http://arxiv.org/abs/1803.10344

Petrovič, P., & Verčimák, P. (2023). *USING PROGRAMMABLE DRONE IN EDUCATIONAL PROJECTS AND COMPETITIONS*. https://arxiv.org/ftp/arxiv/papers/2402/2402.17409.pdf

Prieto Bailo, L. E. (2023). *Diseño e implementación de un nuevo prototipo de dron cuadricóptero controlado mediante STM32 utilizando el entorno de Arduino*. https://upcommons.upc.edu/bitstream/handle/2117/391713/memoria.pdf?sequence=2&isAllowed=y

Pueyo Vidal, A. A. (2018). *Disseny d'un dron de baix cost per docència i missions humanitaries*. https://upcommons.upc.edu/handle/2117/114926

Ríos, Y. (2018). Uso de drones como herramienta educativa en las universidades. *Revista Plus Economía, ISSN-e 2644-4046, ISSN 2411-0353, Vol. 6, Nº. 2, 2018 (Ejemplar Dedicado a: Revista Plus Economía)*, *6*(2), 3. https://dialnet.unirioja.es/servlet/articulo?codigo=9110401&info=resumen&idioma=SPA

Rodríguez, A. (2022). *Diseño de arquitectura, control y programación de robot móvil controlado por gestos*. https://hdl.handle.net/11441/137452

Rodríguez, A. L. T. (2022). Dronética y diseño responsable. *Contrastes. Revista Internacional de Filosofía*, *27*(2), 111–129. https://doi.org/10.24310/CONTRASTESCONTRASTES.V27I2.12820

Rodríguez, E., Vergara, A., & Osorio Verde, I. (2022). INTERFAZ GESTUAL PARA LA MANIPULACIÓN DE SISTEMAS ROBÓTICOS (GESTURAL INTERFACE FOR ROBOTIC SYSTEMS MANIPULATION). *Pistas Educativas*, *43*(141). https://pistaseducativas.celaya.tecnm.mx/index.php/pistas/article/view/2791

Saini, T. (2021). *Manoeuvring drone (Tello ans Tello EDU) using body poses or gestures*. https://upcommons.upc.edu/handle/2117/356499

Saltos, E. J. (2018). *Diseño de un prototipo de sistema de parqueo inteligente para el edificio de la FIE utilizando tecnologías basado en el Internet de las Cosas.* http://dspace.espoch.edu.ec/handle/123456789/10930

Sanz Gómez, J. (2019). *Montaje y configuración de un dron de carreras. Estudio de empuje y autonomía*. http://hdl.handle.net/10016/29681

Shin, S. Y., Kang, Y. W., & Kim, Y. G. (2019). Obstacle Avoidance Drone by Deep Reinforcement Learning and Its Racing with Human Pilot. *Applied Sciences 2019, Vol. 9, Page 5571*, *9*(24), 5571. https://doi.org/10.3390/APP9245571

Tezza, D., & Andujar, M. (2019). The State-of-the-Art of Human-Drone Interaction: A Survey. *IEEE Access*, *7*, 167438–167454. https://doi.org/10.1109/ACCESS.2019.2953900

Torres, A. (2021). *Diseño de un vehículo aéreo cuatrirotor de bajo coste con plataforma Raspberry Pi* [Universitat Politècnica de València]. https://riunet.upv.es/handle/10251/170455

# Anexos

## Anexo A. Programación para la calibración de los motores

```
#include <Servo.h> //Cargar librería Servo.h

#define MAX_SIGNAL 2000 //Variable para la velocidad máxima del motor (2ms)
#define MIN_SIGNAL 1000 //Variable para la velocidad mínima del motor (1ms)
#define MOTOR_PIN1 5 //Define el pin para el motor 1
#define MOTOR_PIN2 6 //Define el pin para el motor 2
#define MOTOR_PIN3 9 //Define el pin para el motor 3
#define MOTOR_PIN4 10 //Define el pin para el motor 4

//Variable para guardar el valor de entrada del monitor serial
int DELAY = 1000;
//Asigna un nombre para cada motor con la librería servo
Servo motor1;
Servo motor2;
Servo motor3;
Servo motor4;

void setup() {
  Serial.begin(9600); //Inicializa el monitor serial
  delay(2000);
  Serial.println("Inicio del programa...");
  //Asigna el nombre de cada motor con su pin de salida digital
  motor1.attach(MOTOR_PIN1);
  motor2.attach(MOTOR_PIN2);
  motor3.attach(MOTOR_PIN3);
  motor4.attach(MOTOR_PIN4);

  Serial.print("Escribiendo el valor máximo en la salida: (");
  Serial.print(MAX_SIGNAL);
  Serial.print("en este caso de:");
  Serial.print("\n");
  //Indica que es momento de conectar la batería y esperar 2 segundos
  //antes de presionar cualquier tecla
  Serial.println("Conectar la batería y esperar 2 segundos antes de presionar cualquier tecla");
  //Se envía una señal se 2ms a cada motor
  motor1.writeMicroseconds(MAX_SIGNAL);
  motor2.writeMicroseconds(MAX_SIGNAL);
  motor3.writeMicroseconds(MAX_SIGNAL);
  motor4.writeMicroseconds(MAX_SIGNAL);
```

```
  //Espera a que se presione cualquier tecla
  while (!Serial.available());
  Serial.read();
  //Imprime que se va enviar la señal mínima a cada motor
  Serial.println("\n");
  Serial.println("\n");
  Serial.print("Enviando el valor mínimo: (");
  Serial.print(MIN_SIGNAL);
  Serial.print(" en este caso");
  Serial.print("\n");
  //Envía la señal mínima de 1ms a cada motor
  motor1.writeMicroseconds(MIN_SIGNAL);
  motor2.writeMicroseconds(MIN_SIGNAL);
  motor3.writeMicroseconds(MIN_SIGNAL);
  motor4.writeMicroseconds(MIN_SIGNAL);

  //Indica que los motores están calibrados y se puede enviar señales PWM
  Serial.println("El ESC está calibrado");
  Serial.println("----");
  Serial.println("Ahora escriba un valor entre 1000 hasta 2000 y presione enter");
  Serial.println("y el motor empezará a girar.");
  Serial.println("Envíe un valor de 1000 para detener al motor y 2000 para girar a máxima potencia");
}

void loop() {
  //Si el monitor serial está disponible leer el valor de la entrada
  if (Serial.available() > 0)
  {
    //Asigna el valor de la entrada del monitor serial a la variable Delay
    int DELAY = Serial.parseInt();
    //Si el Delay es mayor a 1000us enviar la señal a cada motor
    if (DELAY > 999)
    {
      //Prende cada motor con el intervalo de Delay en microsegundos
      motor1.writeMicroseconds(DELAY);
      motor2.writeMicroseconds(DELAY);
      motor3.writeMicroseconds(DELAY);
      motor4.writeMicroseconds(DELAY);
      //Imprime el porcentaje de potencia que se está enviando a los motores
      float Velocidad = (DELAY-1000)/10;
      Serial.print("\n");
      Serial.println("Velocidad motor:");
      Serial.print("  ");
      Serial.print(Velocidad);
      Serial.print("%");
    }
  }
}
```

## Anexo B. Programación del sensor inercial MPU6050

```
#include "Simple_MPU6050.h"         // incluye libreria Simple_MPU6050
#define MPU6050_ADDRESS_AD0_LOW     0x68      // direccion I2C con AD0 en LOW o sin conexion
#define MPU6050_ADDRESS_AD0_HIGH    0x69      // direccion I2C con AD0 en HIGH
#define MPU6050_DEFAULT_ADDRESS     0x68
Simple_MPU6050 mpu;       // crea objeto con nombre mpu
#define spamtimer(t) for (static uint32_t SpamTimer; (uint32_t)(millis() - SpamTimer) >= (t); SpamTimer = millis())
// spamtimer funcion para generar demora al escribir en monitor serie sin usar delay()

// mostrar_valores funcion que es llamada cada vez que hay datos disponibles desde el sensor
void mostrar_valores (int16_t *gyro, int16_t *accel, int32_t *quat, uint32_t *timestamp) {
  uint8_t SpamDelay = 5;      // demora para escribir en monitor serie de 100 mseg
  Quaternion q;         // variable necesaria para calculos posteriores
  VectorFloat gravity;        // variable necesaria para calculos posteriores
  float ypr[3] = { 0, 0, 0 };     // array para almacenar valores de yaw, pitch, roll
  float xyz[3] = { 0, 0, 0 };     // array para almacenar valores convertidos a grados de yaw, pitch, roll
  spamtimer(SpamDelay) {      // si han transcurrido al menos 100 mseg entonces proceder
    mpu.GetQuaternion(&q, quat);    // funcion para obtener valor para calculo posterior
    mpu.GetGravity(&gravity, &q);   // funcion para obtener valor para calculo posterior
    mpu.GetYawPitchRoll(ypr, &q, &gravity); // funcion obtiene valores de yaw, ptich, roll
    mpu.ConvertToDegrees(ypr, xyz);   // funcion convierte a grados sexagesimales
    Serial.printfloatx(F("Yaw")   , xyz[0], 6, 2, F(",   "));  // muestra en monitor serie rotacion de eje Z, yaw
    Serial.printfloatx(F("Pitch"), xyz[1], 6, 2, F(",   "));  // muestra en monitor serie rotacion de eje Y, pitch
    Serial.printfloatx(F("Roll") , xyz[2], 6, 2, F(",   "));  // muestra en monitor serie rotacion de eje X, roll
    Serial.println();       // salto de linea
  }
}

 void setup() {
   uint8_t val;
#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE  // activacion de bus I2C a 400 Khz
   Wire.begin();
   Wire.setClock(400000);
#elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE
   Fastwire::setup(400, true);
#endif
   Serial.begin(115200);     // inicializacion de monitor serie a 115200 bps
   Serial.println(F("Inicio:"));   // muestra texto estatico
   Serial.println(F(" No se establecieron Offsets, haremos unos nuevos.\n" // muestra texto estatico
                    " Colocar el sensor en un superficie plana y esperar unos segundos\n"
                    " Colocar los nuevos Offsets en #define OFFSETS\n"
                    " para saltar la calibracion inicial \n"
                    " \t\tPresionar cualquier tecla y ENTER"));
   while (Serial.available() && Serial.read());    // lectura de monitor serie
   while (!Serial.available());        // si no hay espera
   while (Serial.available() && Serial.read());    // lecyura de monitor serie
   mpu.SetAddress(MPU6050_ADDRESS_AD0_LOW).CalibrateMPU().load_DMP_Image();  // inicializacion de sensor
   mpu.on_FIFO(mostrar_valores);   // llamado a funcion mostrar_valores si memoria FIFO tiene valores
}

 void loop() {
   mpu.dmp_read_fifo();    // funcion que evalua si existen datos nuevos en el sensor y llama
}       // a funcion mostrar_valores si es el caso
```

## Anexo C. Programación del mando del dron

```
#include "Simple_MPU6050.h"
#define usCiclo 80000
#define MPU6050_ADDRESS_AD0_LOW     0x68
#define MPU6050_ADDRESS_AD0_HIGH    0x69
#define MPU6050_DEFAULT_ADDRESS     MPU6050_ADDRESS_AD0_LOW
#include <SoftwareSerial.h>

SoftwareSerial BT(4, 5);
const int boton = 3, led = 13;
const int PotPin = A3;
long loop_timer;
int PotValue = 0, Throttle = 0;
Simple_MPU6050 mpu;
#define spamtimer(t) for (static uint32_t SpamTimer; (uint32_t)(millis() - SpamTimer) >= (t); SpamTimer = millis())

void mostrar_valores(int16_t *gyro, int16_t *accel, int32_t *quat, uint32_t *timestamp) {
  uint8_t SpamDelay = 20;
  Quaternion q;
  VectorFloat gravity;
  float ypr[3] = { 0, 0, 0 };
  float xyz[3] = { 0, 0, 0 };
  spamtimer(SpamDelay) {
    mpu.GetQuaternion(&q, quat);
    mpu.GetGravity(&gravity, &q);
    mpu.GetYawPitchRoll(ypr, &q, &gravity);
    mpu.ConvertToDegrees(ypr, xyz);
    int GiroX = xyz[0];
    int GiroY = xyz[1];
    int GiroZ = xyz[2];
    PotValue = analogRead(PotPin);
    Throttle = map(PotValue, 0, 1023, 900, 1800);
    String datos;
    datos = String(Throttle) + "," + String(GiroY) + "," + String(GiroZ);
    if (Throttle >= 1500){
      digitalWrite(led, LOW);
    }else{
      digitalWrite(led, HIGH);
    }
    Serial.println(datos);
    BT.println(datos);
  }
}
```

```
void setup() {
  BT.begin(9600);
  Serial.begin(115200);
  pinMode(boton, INPUT_PULLUP); // Configurar el botón con resistencia pull-up interna
  pinMode(led, OUTPUT);

  uint8_t val;
#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
  Wire.begin();
  Wire.setClock(400000);
#elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE
  Fastwire::setup(400, true);
#endif
  while (digitalRead(boton) == HIGH);
  while (digitalRead(boton) == LOW);
  while (digitalRead(boton) == HIGH);


  mpu.SetAddress(MPU6050_ADDRESS_AD0_LOW).CalibrateMPU().load_DMP_Image();
  mpu.on_FIFO(mostrar_valores);
  digitalWrite(led, HIGH);
  delay(1000);
}

void loop() {
  mpu.dmp_read_fifo();
}
```

## Anexo D. Programación principal del dron

```
#include <Servo.h>            //Motores
#include <Wire.h>             //Sensor MPU6050
#include <SoftwareSerial.h>   //Bluetooth
#define usCiclo 23000         //Tiempo del ciclo = 6000 microsegundos
#define MAX_SIGNAL 2000
#define MIN_SIGNAL 1000

bool visu = 1;
int visu_select = 0;    // 0: mando RC, 1: giro, 2: acc, 3: ang, 4: esc, 5: PID

SoftwareSerial BT(8, 9);
int valorx, valory, valorz;
int pin = 13;
int led_Batery = 12;

#define MOTOR_PIN1 3
#define MOTOR_PIN2 5
#define MOTOR_PIN3 6
#define MOTOR_PIN4 11

Servo motor1;
Servo motor2;
Servo motor3;
Servo motor4;

// Modificar estos parámetros apara ajustar los PID
float Roll_ang_Kp  = 1, Roll_ang_Ki  = 0, Roll_ang_Kd  = 0;
float Pitch_ang_Kp = 1, Pitch_ang_Ki = 0, Pitch_ang_Kd = 0;
float Pitch_W_Kp   = 1,   Pitch_W_Ki   = 0, Pitch_W_Kd   = 0;
float Roll_W_Kp    = 1,   Roll_W_Ki    = 0, Roll_W_Kd    = 0;
float Yaw_W_Kp     = 1,   Yaw_W_Ki     = 0, Yaw_W_Kd     = 0;

int PID_W_sat1   = 380;  // Limitar parte integral PID velocidad
int PID_W_sat2   = 380;  // Limitar salida del PID velocidad
int PID_ang_sat1 = 130;  // Limitar parte integral PID ángulo
int PID_ang_sat2 = 130;  // Limitar salida del PID ángulo

float PID_ang_Pitch_error, PID_ang_Pitch_P, PID_ang_Pitch_I, PID_ang_Pitch_D, PID_ang_Pitch_OUT;
float PID_ang_Roll_error, PID_ang_Roll_P, PID_ang_Roll_I, PID_ang_Roll_D, PID_ang_Roll_OUT;
float PID_ang_Yaw_error, PID_ang_Yaw_P, PID_ang_Yaw_I, PID_ang_Yaw_D, PID_ang_Yaw_OUT;
float PID_W_Pitch_error, PID_W_Pitch_P, PID_W_Pitch_I, PID_W_Pitch_D, PID_W_Pitch_OUT;
float PID_W_Roll_error, PID_W_Roll_P, PID_W_Roll_I, PID_W_Roll_D, PID_W_Roll_OUT;
float PID_W_Yaw_error, PID_W_Yaw_P, PID_W_Yaw_I, PID_W_Yaw_D, PID_W_Yaw_OUT;
float PID_W_Pitch_consigna, PID_W_Roll_consigna;
```

```
//MPU6050
#define MPU6050_adress 0x68
float angulo_pitch, angulo_roll, angulo_yaw, angulo_pitch_acc, angulo_roll_acc, temperature;
float angulo_pitch_ant, angulo_roll_ant, angulo_yaw_ant;
int gx, gy, gz, gyro_Z, gyro_X, gyro_Y, gyro_X_ant, gyro_Y_ant, gyro_Z_ant;
float gyro_X_cal, gyro_Y_cal, gyro_Z_cal;
float ax, ay, az, acc_X_cal, acc_Y_cal, acc_Z_cal, acc_total_vector;
bool set_gyro_angles, accCalibOK = false;
float tiempo_ejecucion_MPU6050, tiempo_MPU6050_1;

// TIEMPOS
long loop_timer, loop_timer1, tiempo_motores_start, tiempo_ON, tiempo_1, tiempo_2;

// LECTURA TENSIÓN DE BATERÍA
bool LOW_BAT_WARING = false;
float tension_bateria, lectura_ADC;
int LOW_BAT_WARING_cont;

/// SEÑALES PWM
float ESC1_us, ESC2_us, ESC3_us, ESC4_us;
float Throttle_consigna, Pitch_consigna, Roll_consigna;
```

```
void setup() {
  Wire.begin();
  pinMode(pin, OUTPUT);
  pinMode(led_Batery, OUTPUT);
  BT.begin(9600);
  Serial.begin(115200);
  Serial.println("Etapa 0");

  motor1.attach(MOTOR_PIN1);
  motor2.attach(MOTOR_PIN2);
  motor3.attach(MOTOR_PIN3);
  motor4.attach(MOTOR_PIN4);
  delay(2000);

  MPU6050_iniciar();
  Serial.println("Etapa 1");
  MPU6050_calibrar();
  Serial.println("Etapa 2");
  Lectura_tension_bateria();
  Serial.println("Etapa 3");

  motor1.writeMicroseconds(MAX_SIGNAL);
  motor2.writeMicroseconds(MAX_SIGNAL);
  motor3.writeMicroseconds(MAX_SIGNAL);
  motor4.writeMicroseconds(MAX_SIGNAL);
  delay(2000);

  digitalWrite(pin, HIGH);
  motor1.writeMicroseconds(MIN_SIGNAL);
  motor2.writeMicroseconds(MIN_SIGNAL);
  motor3.writeMicroseconds(MIN_SIGNAL);
  motor4.writeMicroseconds(MIN_SIGNAL);
  delay(6000);

  while (BT.available() && BT.read());    // lectura de Bluetooth
  while (!BT.available());         // si no hay espera
  while (BT.available() && BT.read());   // lectura de Bluetooth
  digitalWrite(pin, LOW);
  loop_timer = micros();
}
```

```
void loop() {
  while (micros() - loop_timer < usCiclo);
  loop_timer = micros();

  PWM();                // Generar señales PWM para los motores
  MPU6050_leer();       // Leer sensor MPU6050
  MPU6050_procesar();   // Procesar datos del sensor MPU6050
  PID_ang();            // Obtener salida de los PID de inclinación
  PID_w();              // Obtener salida de los PID de velocidad
  Modulador();          // Modulador o generador de señales para PWM

  // Guardamos las lecturas del sensor MPU6050 para el siguiente ciclo (necesario para los PID)
  angulo_pitch_ant = angulo_pitch;
  angulo_roll_ant  = angulo_roll;
  angulo_yaw_ant   = angulo_yaw;
  gyro_X_ant = gyro_X; // Pitch
  gyro_Y_ant = gyro_Y; // Roll
  gyro_Z_ant = gyro_Z; // Yaw

  Visualizaciones();
}
```