Instituto Tecnológico de Costa Rica

Área Académica de Ingeniería Mecatrónica

Diseño de una herramienta para la predicción, detección y corrección de fallos en propulsores con multirrotores

Informe de Proyecto de Graduación
para optar por el título de

Ingeniero en Mecatrónica con el grado académico de Licenciatura

Dilan Andrey Loría Quesada

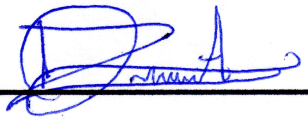
Cartago, julio de 2020

Declaro que el presente Proyecto de Graduación ha sido realizado enteramente por mi persona, utilizando y aplicando literatura referente al tema e introduciendo conocimientos propios.

En los casos en que he utilizado bibliografía, he procedido a indicar las fuentes mediante las respectivas citas bibliográficas.

En consecuencia, asumo la responsabilidad total por el trabajo de graduación realizado y por el contenido del correspondiente informe final.

Estudiante



Dilan Andrey Loría Quesada

Céd: 207620077

Cartago, 28 de julio de 2020

INSTITUTO TECNOLÓGICO DE COSTA RICA

CARRERA DE INGENIERÍA MECATRÓNICA

PROYECTO DE GRADUACIÓN

ACTA DE APROBACIÓN

El Profesor Asesor, da fe de que el presente Proyecto de Graduación ha sido aprobado y cumple con las normas establecidas por la Carrera de Ingeniería Mecatrónica como requisito para optar por el título de Ingeniero en Mecatrónica, con el grado académico de Licenciatura, del Instituto Tecnológico de Costa Rica.

Estudiante: Dilan Andrey Loría Quesada

Nombre del Proyecto: Diseño de una herramienta computacional para la predicción, detección y corrección de fallos en propulsores con multirotores

_____

Dr. Juan Luis Crespo Mariño

Profesor Asesor

Cartago, viernes 14 de agosto del 2020.

Resumen

El presente documento es un informe final para optar por el título de licenciado en ingeniería en mecatrónica, el cual presenta el desarrollo de un sistema capaz de detectar los fallos que se pueden presentar durante el vuelo de un multirrotor. En la primer parte del trabajo ese presenta una investigación de fondo, la cual permitió la caracterización de los fallos en el sistema de propulsión en un UAV multirrotor, luego se presenta el diseño de un sistema de adquisición de datos, el cual sirve como insumo para el diseño de una red neuronal recurrente, encargada de la clasificación y detección de fallos. Al final de este trabajo se logra el diseño de una herramienta capaz de detectar el 70 % de los fallos, entre las diez categorías establecidas dentro de la parte de caracterización, además se plantea la corrección inmediata de los fallos críticos y un protocolo para notificar al operador de la aparición de los fallos.

Palabras clave: *UAV, caracterización de fallos, motor sin escobillas. redes neuronales, LSTM, GRU, MAVLink.*

Abstract

This document is a final report report for the mechatronics engineering degree, which presents the development of a system capable of detecting failures that can occur during the flight of a multirotor. In the first part of the work there is an in-depth investigation which allowed the characterization of the failures of the propulsion systems in a multi-rotor UAV, then the design of a data acquisition system is presented, which serves as input for the design of a recurrent neural network, in charge of fault classification and detection. At the end of this work, the design of a tool capable of detecting 70 % of the failures is achieved, among the ten categories established within the characterization part, in addition an immediate correction mechanism is proposed for the most serious failures and a protocol to notify to the operator the actual status of the vehicle.

Palabras clave: *UAV,fault characterization, brushless motor. neural networks, LSTM, GRU, MAVLink.*

Para todos los que hicieron del viaje hasta aquí,
una experiencia valiosa.

Agradecimientos

En primer lugar, quiero agradecer a mi familia, en espacial a mí mamá y mi papá, que han sido un apoyo incondicional desde el comienzo de mis estudios, sin ellos nada de esto sería posible.

Quiero agradecerle al Dr. Ing. Juan Luis Crespo, quien además de brindarme su apoyo como profesor asesor en el desarrollo del proyecto, me extendió la posibilidad de realizar el mismo para el Grupo Integrado de Ingeniería. También quiero agradecer el Ing. Ronald Loaiza, quien también dio referencia de mi trabajo para obtener esta oportunidad, además de brindarme sus consejos para aplicar a mi proyecto.

A los investigadores del GII, que siempre estuvieron muy anuentes a guiarme y brindarme su apoyo, en especial al Dr. Ing. Félix Orjales Saavedra, quién tuvo una completa disposición para atender mis consultas, aun cuando se paso a la modalidad de trabajo a distancia. También agradezco a mis profesores y a todos aquellos que aportaron a mi formación en la carrera de ingeniería mecatrónica.

Agradezco finalmente a todos mis amigos y amigas , que de una u otra forma han sido un gran apoyo emocional a lo largo de mi vida y hacen mis días más entretenidos. A mi mejor amiga Fiorella, quien siempre ha esperado lo mejor de mí . A Jimena , quien siempre estuvo para escucharme y motivarme, que sin lugar a dudas me dio el mayor apoyo de todos y siempre quiso lo mejor para mí.

Contenido

Lista de figuras

Lista de tablas

Lista de algoritmos

Capítulo 1: Introducción

En el primer capítulo de este trabajo se citan algunas de las aplicaciones más famosas de los vehículos aéreos no tripulados en la actualidad, de igual forma se plantea el problema al que se le dará solución y finalmente se demarcan los objetivos que sirven como guía para el desarrollo del proyecto.

1.1 Antecedentes

En los últimos años el estudio de los drones o vehículos aéreos no tripulados (por sus siglas en inglés UAV), ha crecido de manera considerable, con ello se han desarrollado muchas aplicaciones para el uso de estos dispositivos. Si bien los vehículos no tripulados se utilizaban inicialmente para la guerra, hoy también se utilizan en el trasporte de medicamentos a zonas alejadas, inspección de terrenos y múltiples tareas que de llevarse a cabo por una persona, podrían ser más difíciles o peligrosas [1].

Los vehículos aéreos no tripulados son muy famosos por sus aplicaciones militares, sin embargo, en la actualidad se han explorado nuevas áreas de aplicación para estos dispositivos. Entre los principales beneficios de los UAVs están su bajo consumo de energía y que implican un menor riesgo para la vida humana. Esto los hace una herramienta ideal para la toma de distintos tipos de datos en zonas de difícil acceso [2].

La agricultura es un ámbito que se ve sumamente beneficiado con el desarrollo de los

UAVs. Con ayuda de algunos sensores y cámaras el dron puede recolectar información sobre los terrenos, variaciones en el clima, sobre la presencia y distribución de aguas o fertilizantes irrigados, con estos datos se puede hacer un mejor planeamiento de como se deben tratar las cosechas y mejorar la productividad [2].

La utilización de drones permite la protección de vidas humanas, al no exponerlos a condiciones de riesgo. Las compañías eléctricas han empezado a preferir los UAVs en procesos de inspección en líneas de alta tensión, que además se encuentran en lugares peligrosos. Algunas compañías ferroviarias también utilizan drones para la revisión de fallos en las vías. Actualmente algunas empresas realizan entregas de sus productos por medio de drones. Son además útiles en situaciones de emergencia, donde en ocasiones se han utilizado para la búsqueda de personas en situaciones de desastre o para la entrega de medicamentos a lugares inaccesibles por medios comunes [2].

Una sede donde se realizan estudios con drones es la Universidad de A Coruña (UDC), específicamente en el campus de Ferrol, donde el Grupo Integrado de Ingeniería (GII) desarrolla novedosas aplicaciones para llevar a cabo con ayuda de este tipo de dispositivos.

"El Grupo Integrado de Ingeniería es un grupo interdisciplinar de investigación aplicada en ingeniería, orientado a la transferencia de conocimiento y a la generación de nuevos productos en el entorno industrial. Creado en el año 1999 y ubicado en el Campus de Ferrol de la Universidad de A Coruña, el GII está formado por un grupo multidisciplinar de profesionales. Dado su origen y vinculación a las Escuelas de Ingeniería de la UDC, se da un predominio en las disciplinas de: ingeniería naval, industrial, eléctrica, informática, telecomunicaciones y diseño. Con más de 200 proyectos de investigación desarrollados, el GII se posiciona como un grupo de referencia en sus ámbitos de actuación, estableciendo colaboraciones estables con organismos relevantes nacionales e internacionales" [3].

1.2 Problema existente

Como se resaltó anteriormente, una de las mayores ventajas de los drones, en este caso específico, los que tiene multirrotores, es que ofrecen una gran maniobrabilidad, lo que les permite ser muy ágiles sin importar limitaciones del terreno o el espacio. El problema con este tipo de mecanismos es que son muy inestables por si solos, lo que los obliga a contar con un robusto sistema de control de lazo cerrado. El control de un UAV toma datos de diversos sensores a bordo para la estabilización y navegación del vehículo. Los controles modernos pueden incluso detectar cuando alguno de estos sensores se daña, de manera que se puedan tomar medidas inmediatas, como lo es un aterrizaje [4].

Los fallos más comunes en este tipo de vehículos se dan principalmente en sus componentes móviles, que son los motores y las hélices. El mayor problema que afrontan estos vehículos es que la principal función de sus sistemas de control es la de mantener la estabilidad y esto lo realizan demandando más o menos velocidad a cada uno de los motores, no explorando directamente la causa de la inestabilidad, la cual en ocasiones no puede ser resuelta por el sistema de control actual.[4].

Es específicamente en esa deficiencia de los sistemas de control que se enfocó el Grupo Integrado de Ingeniería, ya que se quería desarrollar un sistema capaz de detectar los fallos, de manera que se pueda saber específicamente que está ocurriendo, además en la medida de lo posible el sistema debía ser capaz de predecirlos, permitiendo así no tener perdidas totales del equipo, como los motores, los cuales pueden quemarse dependiendo de las exigencias a las que se sometan.

Actualmente el GII cuenta con una línea de investigación de UAV como parte de su rama de robótica autónoma. Donde se trabaja principalmente para desarrollarse en el área de realización de tareas autónomas, por lo que se considera de suma importancia poder detectar cualquier tipo de fallo a tiempo, de manera que no se entorpezca la

realización de sus pruebas.

1.3 Objetivos

El objetivo general que se plantea para este proyecto de graduación es diseñar una herramienta capaz de predecir, detectar y eventualmente corregir fallos en un UAV con multirrotores, por medio de un modelo de predicción, para el Grupo Integrado de Ingeniería.

1. Definir las circunstancias que pueden producir que el vehículo se desvíe de su funcionamiento normal y los variables de medición en las que se puedan ver reflejadas.

2. Diseñar un modelo de predicción y detección con base en los resultados de la etapa anterior.

3. Establecer los protocolos de control para dar respuesta a los diferentes fallos que se pueden presentar durante el vuelo.

4. Evaluar la funcionalidad del diseño por medio de pruebas de campo real o simulaciones.

Capítulo 2: Marco Teórico

En este capítulo gira en torno a tres temas principales. El primero es una descripción de la composición de un sistema multirrotor, así como las bases de funcionamiento. Seguidamente se trata el tema de fallos que se pueden presentar en un dispositivo de este tipo, así como las principales causas que los provocan. Finalmente se dedica una sección al uso de la inteligencia artificial a problemas de naturaliza principal, haciendo un énfasis especial en las redes neuronales recurrentes, las cuales fueron elegidas para el desarrollo del trabajo.

2.1 Multirrotores: Conceptos relacionados con sus sistemas y funcionamiento

Como se menciona en [5], se puede definir un multirrotor o multicóptero como un dispositivo que tiene múltiples hélices o propulsores de ascenso, lo que lo diferencia del helicóptero, el cual solo cuenta con un motor para esta función. Otra diferencia radica en la ausencia de un rotor de cola, que en los helicópteros se utiliza para dar un control de guiñada y contrarrestar el par inducido por el rotor principal. Como se puede observar en la Figura 2.1, existen diferentes configuraciones de multicópteros, dependiendo de la cantidad de rotores, en la figura se observan algunas de ellas como lo son el cuadricóptero, el tricóptero y el hexacóptero.

Figura 2.1: Vehículos multirrotores con diferente número de motores. [5]

Un UAV se refiere a todo aquel vehículo que no lleva un piloto a bordo, en su lugar se controla por medio de un mando remoto o algún programa autónomo. Por otra parte, en 2.1 se menciona que la Administración Federal de Aviación de los Estados Unidos, define un dron como un vehículo que vuela fuera de la línea de visión por lo que debe poseer la capacidad de vuelo autónomo. Con estos conceptos claros se puede definir que un multicóptero es un UAV, sin embargo, no se puede considerar directamente como un dron, ya que un dron puede ser multicóptero, pero no todo multicóptero es un dron.

2.1.1 Elementos y sistemas principales

Un multicóptero esta conformado por varios sistemas y elementos que son los que hacen posible que este vuele y se mantenga estable en el aire. Entre los principales sistemas se pueden mencionar el de propulsión, control y comunicación, los cuales se montan sobre un marco, conformando así el multicóptero. Los marcos se suelen manufacturar con materiales ligeros como lo son el aluminio o la fibra de carbono, de manera que se reduzca al mínimo posible el peso del vehículo [5]. Una mejor descripción de la composición de un vehículo multirrotor puede ser observada en la Figura 2.2.

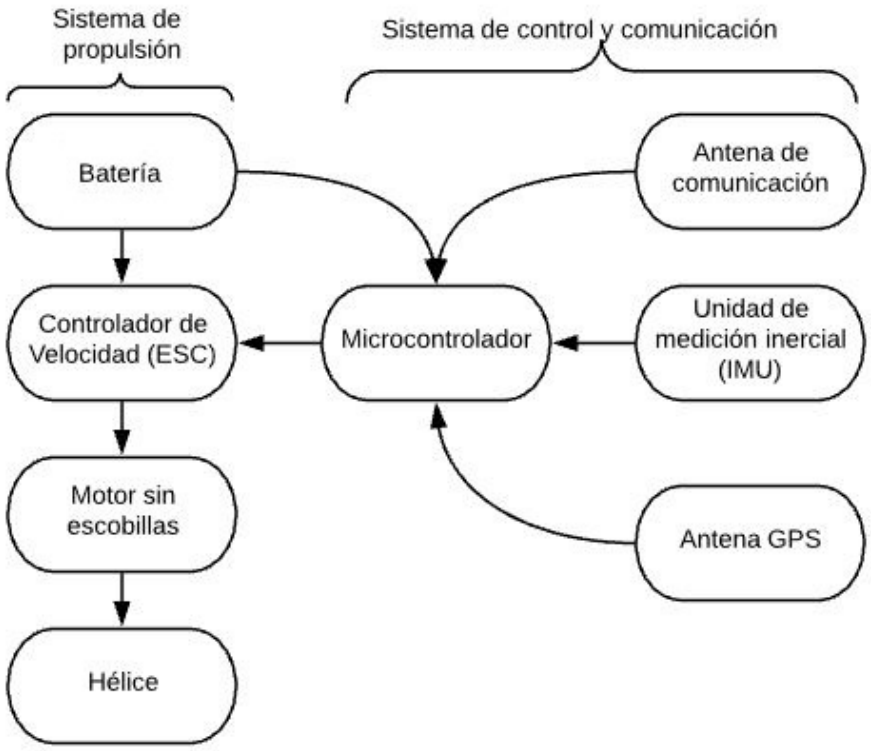


Figura 2.2: Principales elementos y sistemas de un dispositivo multirotor.

2.1.1.1 Sistema de propulsión

Los elementos que componen el sistema de propulsión son los motores sin escobillas y las hélices. Estos dos componentes son los que se encuentran bajo mayor presión durante el vuelo, debido a que son los encargados de generar la fuerza que levanta y mantiene en el aire al vehículo. En la Figura 2.3 se puede observar este sistema.

El tamaño de las hélices está directamente relacionado con la fuerza de empuje que estas pueden dar, una hélice más grande ejercerá una mayor fuerza. Sin embargo, un mayor tamaño también implica un mayor impacto o esfuerzo aplicado al cubo central de las palas. Es por esto por lo que las hélices deben ser de un material resistente. Además, las palas grandes también obligan al motor a ser capaz de contrarrestar el torque que estas ejercen, estos componentes trabajan de manera conjunta, por lo que se debe verificar muy bien la compatibilidad entre ellos para no dañar el equipo [5].

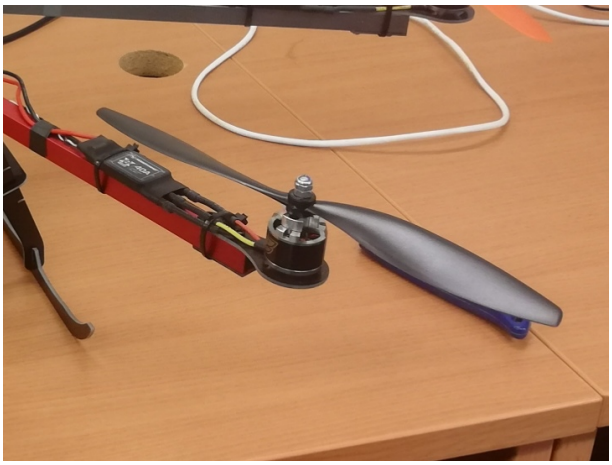


Figura 2.3: Sistema de propulsión de un multicóptero.

Los multicópteros utilizan motores sin escobillas, debido a que estos permiten obtener una mejor relación entre el torque y le tamaño, lo cual resulta de suma importancia en estos dispositivos, donde el peso es un factor crítico [6].

Los motores sin escobillas están conformados por un estator y un rotor, donde uno de estos se compone de imanes permanentes, mientras el otro tiene un cierto número de bobinados. Al hacer pasar una corriente por los bobinados se produce un campo magnético, por lo que, mediante fuerzas de atracción y repulsión, el rotor comienza a moverse de acuerdo con como se este aplicando la corriente a las diferentes bobinas. Por lo general se utiliza una configuración de tres fases en el bobinado y se controla la velocidad de conmutación por el fenómeno de la fuerza electromagnética reversa o por sensores de efecto Hall, que trabajan en conjunto con el controlador de velocidad [6]. Esta manera de realizar la conmutación es la que diferencia a este motor del clásico motor DC con escobillas.

2.1.1.2 Sistemas de control y comunicación

Se entiende por sistema de comunicación aquel que comunica el vehículo con el usuario. La principal comunicación humano-máquina se da por medio de un control remoto, que por medio de señales de radio le indican al multicóptero los movimientos que debe

realizar en el aire [5].

El sistema de control es aquel que, por medio de un controlador de vuelo, realiza un análisis de datos a una alta velocidad, con la información proveniente desde los sensores y el comando remoto, para posteriormente enviar una señal a los actuadores que realizarán los movimientos deseados en el multicóptero, además de mantener la estabilidad en el aire. Los sensores que componen este sistema por lo general son un magnetómetro, que permite conocer la dirección; un GPS que permite conocer la ubicación y una unidad de medición inercial (IMU), la cual proporciona información sobre la velocidad y orientación, que a su vez permite determinar el movimiento de cabeceo, balanceo y guiñada, esto ocurre varios miles de veces por segundo. Además suele contar con un barómetro para determinar la altitud [5].

El sistema de control trabaja con toda la información recolectada de estos sensores y la compara con las señales que llegan al receptor de radio, de manera que se crea una impresión de que esta haciendo el dron y que se desea que haga. Con todos estos datos el control se crea una idea de su estado actual y procede a enviar información de velocidad al controlador de velocidad, que a su vez actuará directamente sobre los motores, produciendo así el cambio al estado deseado [5].

2.1.1.3 Otros elementos que se pueden equipar

Con los sistemas descritos anteriormente un multicóptero ya es capaz de volar de manera correcta, por lo que esos son los sistemas esenciales para su funcionamiento. Sin embargo, en muchas ocasiones se adicionan otros elementos, que dependiendo de la aplicación que se le quiera dar, son necesarios.

Uno de los dispositivos extras que casi todo multicóptero lleva a bordo es la cámara, la cual permite guardar vídeos y fotografías de lo que ve el vehículo. Adicional a esta se puede instalar un estabilizador para su montaje, el cual se conecta a los demás sistemas, de manera que permite ajustarse al movimiento y mantener a la cámara estable. Este

sistema de visión por cámara suele contar con su propio transmisor a bordo, el cual permitiría al usuario ver en tiempo real la imagen captada [5].

2.1.2 Bases de su funcionamiento

El modelo del sistema de control de un octocóptero puede variar mucho, dependiendo de todas las fuerzas externas que se consideren, dependiendo de la precisión con la que se desee modelar. Algunas fuerzas que se consideran en el modelado son la fuerza de empuje, la gravedad o el torque de reacción provocado por las hélices. Sin embargo, lo que tienen en común la mayoría de estos sistemas es que trabajan con un lazo cerrado y el controlador más utilizado suele ser un PID [7].

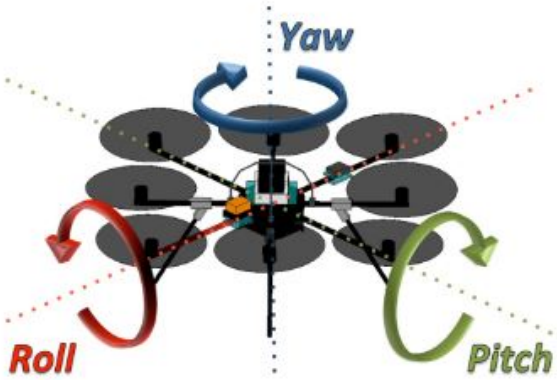


Figura 2.4: Movimientos básicos de los multicópteros [7] .

En la Figura 2.4 se pueden observar los ejes sobre los que un octocóptero puede moverse y en la Figura 2.5 se observan las maniobras que se pueden realizar sobre ellos. Como se visualiza, para llevar a cabo cada uno de estos movimientos fundamentales se utilizan los ocho motores. En la Figura 2.5 el color verde indica un aumento en la velocidad angular, el rojo una disminución y el gris que se mantiene una velocidad constante.

Para llevar a cabo cada una de las maniobras se realizan variaciones a la velocidad de los motores, por lo que se considera a la velocidad como la salida principal de este sistema. Por lo general el control de un octocóptero es una tarea compleja, sin embargo,

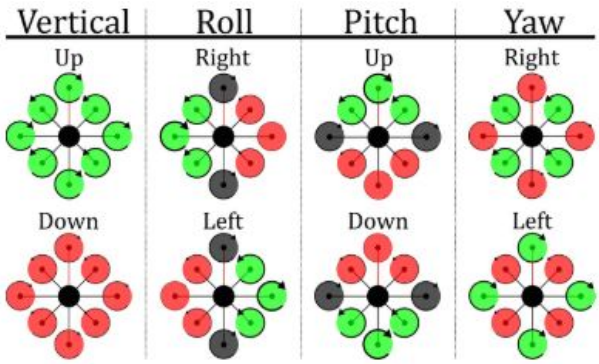


Figura 2.5: Las ocho maniobras básicas de un octocóptero [7].

de manera general se puede ver como un sistema de múltiples entradas y salidas, que recibe a la entrada una posición o maniobra a la que debe llevar al vehículo y esto lo hace mediante variaciones en la velocidad en los ocho motores [7].

Siendo la velocidad el factor más importante en el control de los movimientos de un vehículo multirrotor, se debe conocer como realiza estos cambios en la velocidad el controlador de velocidad (ESC por sus siglas en ingles). Un ESC controla la rotación de los motores al energizar selectivamente cada una de las fases. El proceso de energización de cada uno de los bobinados del motor se debe realizar siguiendo una secuencia, sin embargo, para el ESC, por sí solo, es imposible saber la posición del rotor, en un momento dado, sin ayuda de algún tipo de sensorización. La información de la posición del rotor se puede observar por medio de la señal producida por la fuerza contra-electromotriz. Cuando se observa esta señal de tensión, se puede ver la peculiaridad de que se presenta un cruce por cero, cuando el imán pasa por la bobina inactiva, de manera que esto revela la posición del rotor y permite saber cuál será la siguiente bobina que se energizará [8].

2.2 Caracterización de las fuentes de fallos presentes en el vuelo de UAVs con multirrotores

Los vehículos aéreos no tripulados son sistemas que se ven afectados por una gran variedad de factores externos, los cuales pueden provocar fallos a la estructura y sistemas que lo componen. Los fallos que se pueden presentar son de varios de tipos, dependiendo la forma y lugar en los que se presentan. Pueden tener un origen mecánico, eléctrico o incluso magnético. Muchos fallos pueden llevar a la pérdida parcial o total de motores, elemento fundamental para el vuelo.

2.2.1 Fallos debido a daños estructurales del marco y las hélices

A nivel de la estructura de un multirrotor, se pueden presentar fallos en el marco o en las hélices, que pueden conducir a vuelos inestables, lo cual puede a su vez llevar a pérdidas más valiosas como lo son los motores u otros componentes del vehículo.

En el trabajo de [4] se mencionan tres fallos principales a nivel estructural. Los cuales son el aflojamiento o pérdida de la base de montaje del motor, un desbalance en la hélice y finalmente un aflojamiento o desmontaje de la misma.

Como se observa en la Figura 2.6, existe un desplazamiento en el brazo de un cuadracóptero producto de la fuerza ejercida por los motores y las hélices. Este movimiento o desplazamiento se presenta con una magnitud mayor en los extremos, por lo que no es de extrañar que sean la parte la estructura donde se presenten mayor cantidad de problemas a nivel estructural.

Esto nos lleva sobre todo al primer problema que se menciona en [4], el cual es el aflojamiento del montaje. La presencia de vibraciones y movimientos en la estructura puede llevar consigo al aflojamiento de los tornillos que mantienen al motor unido al marco, además en el peor de los casos, se pueden soltar del todo. Este último caso puede

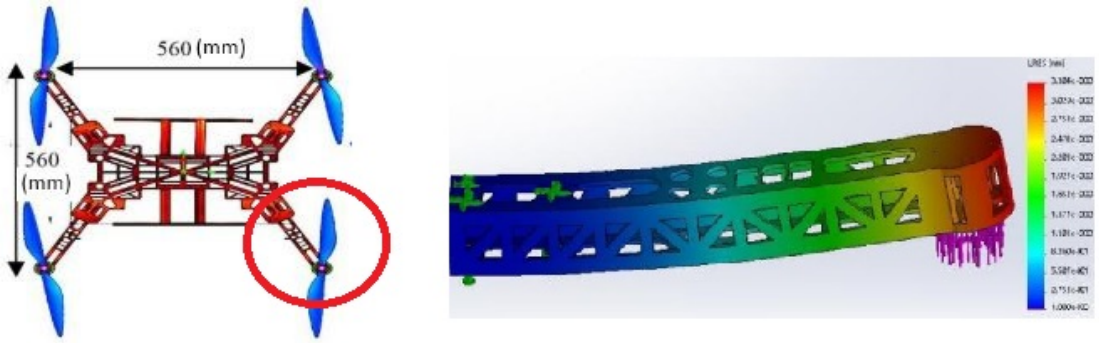


Figura 2.6: Diagrama de desplazamiento debido a la fuerza de empuje en un brazo de un multirrotor [9].

ocurrir también si ocurriese una ruptura en el marco, específicamente en el soporte sobre el que se atornilla el motor, el cual se puede fatigar debido a la carga cíclica que recibe del motor el cual pasa cambiando de velocidad. El resultado de estos fallos podrían ser la pérdida del motor o que continúe con un funcionamiento entorpecido, lo que inevitablemente crearía una gran inestabilidad en el vehículo y provocaría que se acumulen errores en los algoritmos de navegación, en cuanto a la estimación de errores en la velocidad y posición del UAV.

Figura 2.7: Hélices con daño diferente grado de daño estructural utilizado para pruebas en [10].

Los otros fallos mencionados en [4], se relacionan directamente con las hélices. El primero hace referencia a la presencia de un desbalance en las palas. Por lo general un propulsor de este tipo se compone por 2 o tres palas, las cuales deben encontrarse en un preciso equilibrio, para que se mantenga el balance del vehículo cuando estén girando a

altas velocidades. Un ejemplo de un daño estructural que puede dañar el equilibrio de las palas se presenta en la Figura 2.7. El otro fallo se presenta al aflojarse o perderse la hélice, cuando una hélice no se asegura correctamente esta propensa a desmontarse al empezar debido a altas velocidades de giro y a la vibración del multirrotor.

Tanto en [4] como en [10], se menciona el impacto directo que tienen estos fallos en la fuerza de empuje, la cual se reduce notablemente con tan solo daños estructurales como los observados en la Figura 2.7. Una reducción en la fuerza de empuje conlleva a una reducción en la corriente demandada, ya que se reduce la carga de los motores, lo que implica una menor demanda de potencia. Además, estos fallos tienen en común que conducen a un desbalance del multirrotor, reflejado en un aumento abrupto en las vibraciones de sus tres ejes.

2.2.2 Fallos debido al desgaste mecánico de los motores

Los motores sin escobillas están propensos a fallar por el desgaste mecánico de sus componentes, principalmente de aquellos que le facilitan la rotación, como lo son los rodamientos. Como se menciona en [11], los fallos en rodamientos son los responsables en mayor incidencia del fallo de los motores.

Los rodamientos están conformados por dos anillos, el anillo externo y el anillo interno, además de un conjunto de bolas, colocados en pistas de rodadura, que giran dentro de estos anillos. Como se menciona en [11], la tensión continua a la que se someten los rodamientos puede provocar fallas por fatiga, generalmente en las pistas de rodadura, donde pequeñas piezas se desprenden, en un proceso llamado descamación o astillamiento. Por lo general estas fallas provocan un funcionamiento brusco del motor, lo que genera vibraciones detectables y altos niveles de ruido. Además de estos problemas, el rodamiento se puede ver afectado por condiciones externas, como contaminación, corrosión, mala lubricación o una instalación inadecuada. En el peor de los casos, un daño en un rodamiento puede llevar al bloqueo del rotor, lo que se

traduciría en una gran carga para el motor, llevándolo al consumo de altas corrientes que pueden quemar los bobinados y dañar el motor.

En [11] se hace mención de como las excesivas vibraciones pueden acabar dañando el motor. Las vibraciones mecánicas en conjunto con múltiples ciclos de enfriamiento y calentamiento,de los bobinados del motor, pueden producir el desgaste del aislamiento de los bobinados, lo cual resulta normalmente en fallos que conducen a cortocircuitos y pérdida del motor.

2.2.3 Fallos asociados a problemas eléctricos o electromagnéticos

En el sistema eléctrico existen tres componentes principales que pueden verse expuestos a pérdidas totales, los cuales son la batería, el controlador de velocidad y el motor. Por lo general los fallos se dan producto de la presencia de corrientes mayores para las que están diseñados, por lo que pueden terminar quemándose estos elementos. Por otra parte la desmagnetización es un fenómeno a la que todos los imanes se pueden ver expuestos, como es el caso de los imanes permanentes presentes en los motores sin escobillas.

Como se indica en [12], las baterías de polímero de litio (LiPo), que son las utilizadas comúnmente para volar los multirrotores, están propensas a fallar si no se utilizan de manera correcta. Este tipo de baterías manejan una tensión nominal de 3.7 voltios, sin embargo, cuando se encuentran completamente cargadas alcanzan hasta 4.2 voltios debe seleccionar correctamente la batería de acuerdo con las necesidades del multicóptero, ya que estas tienen unas ciertas limitaciones, como la tasa de descarga. Esta se refiere a la cantidad de amperios por hora que puede entregar, si se tienen motores o controladores que puedan demandar más que lo establecido en que las especificaciones, es muy posible llegar a dañar la batería. Por lo general un daño en las baterías LiPo se evidencia por un aumento en la temperatura y una caída en la tensión, que se traduce en una caída de potencia en el motor. En la guía de [12] también se establece que la tensión de estas

baterías nunca debe llegar a ser menor a 3 voltios por celda, ya que esto degrada su vida útil.

El siguiente elemento propenso en el circuito encargado de alimentar los motores es el controlador de velocidad. Este elemento se compone por muchos componentes electrónicos que son propensos a degradarse por su repetido uso, reduciendo así el desempeño del controlador. Un incorrecto funcionamiento del ESC afecta directamente a la batería y el motor. En el trabajo de [13], se menciona que entre las consecuencias que al que este fallo puede llevar son a una mayor demanda de corriente, lo cual eleva la temperatura de los bobinados del motor y pueden conducir a una pérdida en la fuerza de empuje que este entrega.

Otro fallo que se puede presentar en el sistema es la desconexión de una o más de las fases del motor. Las uniones entre el ESC y los motores son propensas a soltarse debido a las vibraciones de los multicópteros y si estas no están debidamente aseguradas. Como se menciona en el trabajo de [14], la pérdida de una de las fases puede resultar en el fallo del equipo, ya que debido a la desconexión de una de sus fases, el motor comienza a demandar mucha más corriente en las dos restantes, lo que producirá un sobrecalentamiento en los bobinados del motor. Además de este fenómeno, se presenta una pérdida en la potencia que entrega el motor, lo cual puede terminar llevando al equipo a un estado de rotor bloqueado y por consiguiente dañar definitivamente el motor,si no existe una respuesta rápida.

Los motores sin escobillas están conformados por bobinados que funcionan como electroimanes que atraen o repelen a los imanes permanentes, los cuales pueden estar en el estator o en el rotor. Por lo general se utilizan imanes de neodimio en los motores sin escobillas, los cuales se pueden corroer y terminar desintegrándose a altas velocidades como se menciona en [11]. La desintegración, puede producir que fragmentos del imán provoquen un aumento en la fricción, que se convierte en un potencial daño al aislamiento de los bobinados. Además se señala en [11], sobre otro fallo conocido

como la desmagnetización de los imanes permanentes.

La desmagnetización es un fenómeno que se puede presentar debido a la exposición de un imán permanente a un campo magnético o a altas temperaturas que lo hacen perder sus propiedades magnéticas [15]. En un estudio realizado en [16], se pueden observar los efectos de la desmagnetización magnética, en el funcionamiento del motor sin escobillas. La desmagnetización se evidencia principalmente en un aumento en el torque y un aumento en la corriente, debido a que, al disminuir la atracción magnética de los imanes permanentes, se debe aumentar la de los bobinados por medio de un aumento en la corriente, que se convierte en un mayor torque a la salida. Los el torque de los motores sin escobillas se caracteriza por tener un pequeño rizado, sin embargo cuando existe una desmagnetización, este rizado tiende a ser mayor, como se puede observar en la Figura 2.8 , siendo otro de los aspectos que permite detectar el fenómeno de desmagnetización.

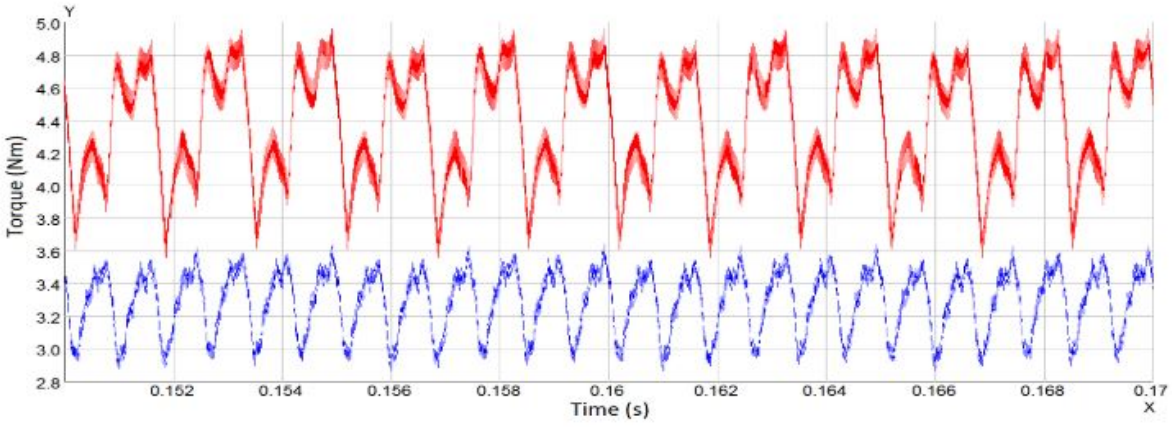


Figura 2.8: Comportamiento en el torque de un motor sin escobillas sano(azul) y con una desmagnetización de uno de sus imanes(rojo) [16].

2.2.4 Requisitos para el monitoreo de condiciones de fallo

Muchos fallos se pueden detectar o incluso predecir si se cuenta con los sensores correctos para el monitoreo de distintas variables. En el trabajo de [11], se agrupan las técnicas de monitoreo más prominentes en tres grupos principales, el monitoreo térmico, el

monitoreo de vibraciones y el monitoreo eléctrico. El monitoreo térmico se puede llevar a cabo con sensores de temperatura que se instalan en el motor, muchos fallos pueden conducir a una elevación de la temperatura del motor. El análisis de vibraciones permite detectar algunos fallos mecánicos, debido a que cada irregularidad que aporta una vibración suele tener su propia frecuencia de vibración, los acelerómetros permiten medir estas vibraciones en unidades de distancia, velocidad o aceleración. Finalmente, el monitoreo de la tensión y corriente es indispensable, ya que muchos fallos también se pueden identificar debido a un aumento en el consumo de eléctrico.

2.3 Uso de la inteligencia artificial en la detección de anomalías

Como se menciona en el trabajo de [17], los métodos tradicionales para la detección de anomalías, en los datos recolectados por sensores, se basan en la ingeniería de características específicas del dominio. Por lo general se utiliza el conocimiento que se posee del dominio para analizar los datos del sensor y crear características que se basan en estadísticas, de manera que con estas se entrenan modelos de aprendizaje automático para detectar y clasificar anomalías. Este método, si bien es utilizado, tiene la deficiencia de que la extracción de estas características requiere mucho trabajo por parte de los expertos en el dominio.

Existen infinidad de algoritmos y trabajos que se han realizado en torno a la detección de anomalías. Muchos de estas técnicas se basan en el establecimiento de características o limites, sin embargo como se mencionaba anteriormente, muchas veces establecer estas propiedades puede llegar a ser complicado. Es por esto por lo que en la actualidad se opta mucho por la utilización de técnicas basadas en extracción automática de características, como las que se muestran a continuación:

- **Enfoque basado en la lógica difusa**: En la investigación de [17], se explica

cómo los métodos de lógica difusa se utilizan mucho en la automatización de la producción de reglas de asociación. Los métodos modernos de lógica difusa utilizan un método de ventana deslizante para subdividir las secuencias temporales, de esta manera busca patrones en estas subsecuencias y produce automáticamente un conjunto de reglas de asociación, que se pueden definir por el orden de aparición, las relaciones de dependencia mutua u otras características. Un ejemplo de esta técnica se puede ver en el trabajo de [18], donde se utilizó lógica difusa para la detección de anomalías en las mediciones de precipitación y electrocardiogramas de arritmias.

- **Enfoque basado en programación genética**: De igual forma en el trabajo de [17], se describe esta técnica. La programación genética se ha utilizado en la selección de subconjuntos de características, de manera que se reduce el costo de computo durante el proceso de clasificación. En la actualidad esta técnica ha mejorado al punto en que tiene la capacidad de sintetizar nuevas características a partir de las existentes, o incluso a partir de datos sin procesar, es decir que esta técnica no asume ningún conocimiento previo sobre la distribución probabilística de los datos. En el trabajo de [19], se utilizó este método para detectar y clasificar fallos en los datos sin procesar de vibración de una maquina rotativa.

- **Enfoque profundo basado en redes neuronales**: En cuanto al uso de redes neuronales se han realizado investigaciones, donde se han utilizado dos tipos de redes para la detección y clasificación de anomalías en datos sin procesar, como se menciona en [17], los cuales son el Mapa auto-organizado (SOM) y las Redes neuronales de función de base radial (RBF). Los SOM funcionan encontrando relaciones estadísticas entre los puntos de datos en un estado dimensional alto y luego convierten estas relaciones geométricas en un mapa bidimensional, lo cual permite extraer de manera automática algunas características, un ejemplo de esta red se puede ver en el trabajo de [20], donde se utilizó este método para

la detección de anomalías en los datos de un sensor de solenoide en una válvula hidráulica de la NASA.

- **Redes neuronales recurrentes (RNN)**: Como se define en el trabajo de [17], una red neuronal recurrente se compone de múltiples copias del mismo módulo, donde cada uno de estos es una red neuronal que pasa su salida una a un módulo anterior . Son redes con dos entradas, el presente y el pasado inmediato, dado que cada uno de los módulos sirve como una celda de memoria que almacena el estado anterior. Esta estructura la convierten en una red ideal para el análisis de datos secuenciales, con una dinámica temporal, como lo son los datos de series temporales de un sensor. Las redes recurrentes simples tienen el problema de contar con una memoria de corto plazo, por lo que solo detectan la dependencia del tiempo en secuencias bastante cortas, debido a esto se ha le han realizado modificaciones para crear arquitecturas con una mayor capacidad de memoria. Las dos arquitecturas que más sobresalen son la *Long-Short Term Memory*(LSTM) y la *Gated Recurrent Unit* (GRU). Son arquitecturas similares, sin embargo, la LSTM es mucho más compleja, lo cual la hace más potente, por otra parte, se considera que la GRU es más eficiente computacionalmente. Ambas arquitecturas han sido utilizadas en la detección de anomalías, como se puede observar en el trabajo de [21], donde se utilizan en el análisis de series temporales de datos del tráfico de red.

Es debido a la complejidad que puede tener establecer características, que se utilizan alternativas diferentes, como lo son algoritmos de aprendizaje profundo. Como lo menciona [17], se ha demostrado que las redes neuronales profundas son muy efectivas en la extracción automatizada de las características abstractas en las tareas de clasificación.

Los datos que se toman de los sensores en un UAV oscilan mucho, debido a que es un vehículo en movimiento, que acelera y desacelera con frecuencia. Es debido a esto que el análisis de los datos no se puede limitar a momentos específicos en el tiempo,

ni se pueden delimitar características con facilidad. Los datos deben ser analizados en secuencias completas, es decir que es un problema dependiente del tiempo donde encontrar un patrón mediante cálculos comunes sería complejo, debido también al gran número de sensores. En este tipo de problemas lo ideal es utilizar redes del aprendizaje profundo capaces de tratar con series temporales de datos, como lo son las redes neuronales recurrentes.

2.3.1 Redes Neuronales Recurrentes(RNN)

Las redes neuronales recurrentes funcionan de manera similar a las redes neuronales del perceptrón multicapa, sin embargo, incorporan el concepto de recurrencia, este las convierte en algoritmos ideales para el análisis de una secuencia de datos. Las redes recurrentes, como se menciona en [22], se puede modelar una secuencia de manera generalizado, como un arreglo de la forma $\vec{X}^1, \vec{X}^2, ...\vec{X}^\tau$, en las redes recurrentes se da un procesamiento de secuencias con un largo arbitrario de τ, donde se implementa el concepto de parámetros compartidos.

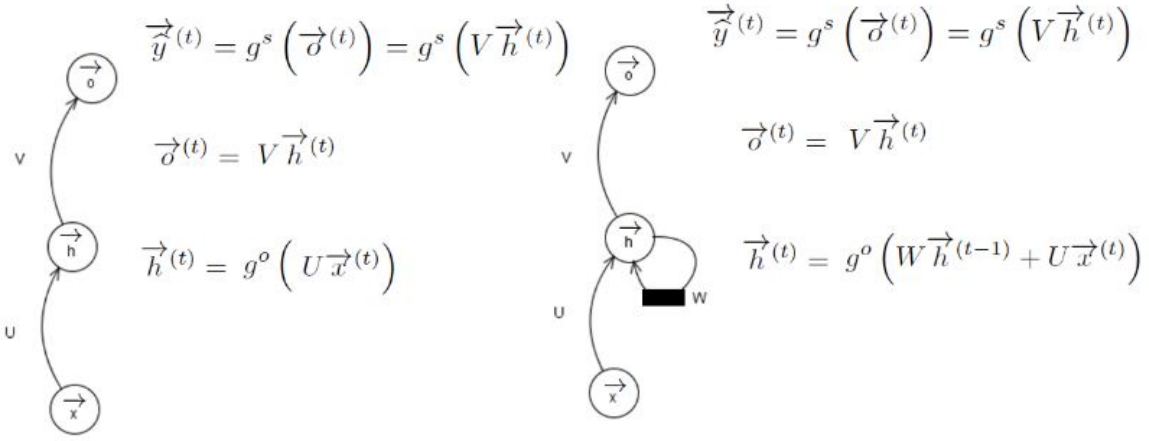


Figura 2.9: Red de perceptrón multicapa vs Red Recurrente [22].

En la Figura 2.9 se observa la diferencia entre una red de perceptrón multicapa(MLP) y una red recurrente(RNN). A la izquierda se puede ver como en el MLP las operaciones

de la red perceptrón para el calculo de la salida de cada neurona es siempre hacia adelante, la salida de una capa o entrada se multiplica por una matriz de pesos y ese valor va directo a la capa siguiente o salida, por otra parte en la RNN, se aprecia el elemento que aplica a recurrencia dentro de la red, donde para el cálculo del estado oculto $\vec{h}^{(t)}$, se considera el valor de $\vec{h}^{(t-1)}$, multiplicado por otra matriz de pesos W.

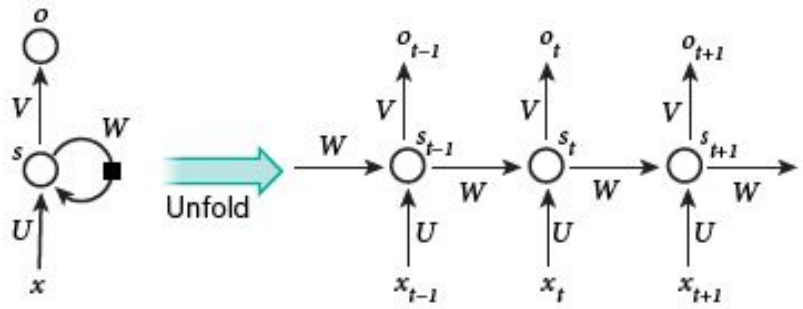


Figura 2.10: Red neuronal recurrente sencilla y su desarrollo en el tiempo [23].

La Figura 2.10 permite entender mejor la forma en la que se aplica la recurrencia en distintos instantes de tiempo. Como se observa el desenvolvimiento de la capa recurrente, el estado oculto s^t se calcula no considerando el estado anterior, que a su vez había sido calculado con el estado anterior. De esta manera se crea una recurrencia de los datos, lo que permite analizarlos como parte de una secuencia, donde la salida depende de un cálculo que considera, siempre el el estado t-1. Como lo dice [22], la tamaño de esta cadena de esta recurrencia va a estar dada por el longitud específica de la secuencia.

Las redes recurrentes se poseen los siguientes componentes esenciales, los cuales según como se menciona en [22], son usuales en arquitecturas completamente conectadas y convolucionales, que se basan el concepto de estado oculto, utilizado en los modelos ocultos de Markov. A continuación, se presentan estos elementos, basados en la Figura 2.10 :

- **Unidad de entrada**(x_t): Esta esta conformada por la secuencia de valores que son utilizados en el calculo de la salida, su conexión con la unidad oculta esta

dada por la matriz de pesos **U**.

- **Unidad oculta**(s_t):La unidad oculta se puede definir el resultado de aplicar una función de activación a la combinación lineal entrada x_t por la matriz **U** y el estado oculto anterior s_{t-1} por la matriz **W**, tal y como se expresa en la ecuación 2.1.

$$s_t = f(\mathbf{W}\vec{s}_{t-1} + \mathbf{U}\vec{x}_t) \tag{2.1}$$

- **Unidad de salida**(o_t): La unidad de salida se obtiene mediante el producto de la unidad oculta con la matriz de valores **V**. El valor final de la salida por lo general se obtiene al pasar el valor de (o_t) por una función de activación dando como resultado $y_{(t)}$.

- **Función de pérdida**: Esta es la función que permite determinar la distancia o diferencia entre la salida obtenida y la que se debía obtener. En el caso de la función de entropía cruzada, uno de los métodos m[as comunes cuando se trata de una red clasificación, se utiliza la ecuación 2.2, donde $y_{(t)}$ es la salida y $\hat{y}_t$ es el valor esperado.

$$-\sum \vec{y}_{(t)} log\left(\vec{\hat{y}}_{(t)}\right) \tag{2.2}$$

Una vez que una red neuronal calcula con la función de error, la diferencia entre la salida obtenida y la esperada, este se utiliza en el cálculo del gradiente de las distintas matrices de pesos. Como se menciona en [22], para desarrollar el cálculo de gradiente de una red recurrente se debe desenvolver el grafo en las unidades de tiempo de la secuencia, calcular y acumular el error para cada uno, volver a envolver y luego actualizar los pesos. Este proceso se denomina propagación hacia atrás a través del tiempo y es característico de las redes recurrentes, que consideran la naturaleza temporal. La actualización de

pesos se da empezando por la matriz **V**, que son los pesos de la unidad de salida, luego se actualizan **U** y **W**, pesos de la unidad oculta.

Las redes recurrentes se enfrentan comúnmente a un problema de desvanecimiento de gradiente. El cálculo de un gradiente se da por medio de derivadas parciales de los pesos con respecto a la función de error, como se observa en el trabajo de [22]. La aplicación de un gran número de derivadas parciales consecutivas puede llevar a la reducción del gradiente a valores muy pequeños o cercanos a cero, fenómeno conocido como desvanecimiento de gradiente. Las redes recurrentes analizan secuencias largas, con muchos elementos, por lo que se debe considerar que se realiza una derivada parcial por cada uno de estos y esto propicia la aparición de este problema.

2.3.1.1 Long Short-Term Memory(LSTM)

La red de LSTM es un tipo de la red recurrente la cual fue inventada con el objetivo de corregir el problema del desvanecimiento de gradiente. En el trabajo de [24] se menciona que la idea clave de estas redes fue la incorporación de controles no lineales y dependientes de datos, en la celda RNN, los cuales pueden entrenarse para garantizar que el gradiente de la función objetivo con respecto a la señal de estado no se desvanezca. Este control es conocido como compuerta de olvido, ya que permite exactamente eso, olvidar los estados de celdas anteriores, de manera que se disminuyan las multiplicaciones sucesivas en los parámetros de la matriz de pesos **W**.

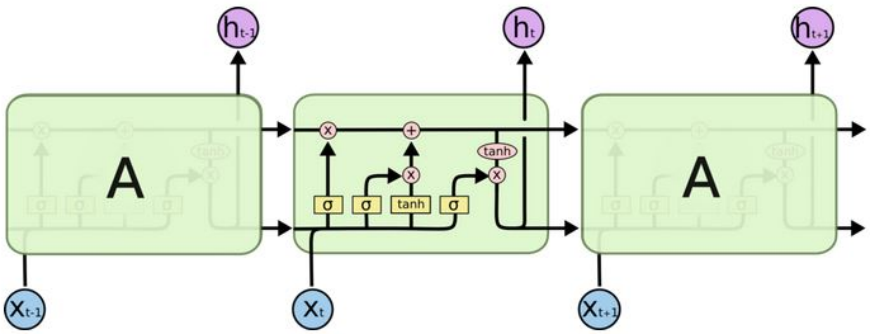
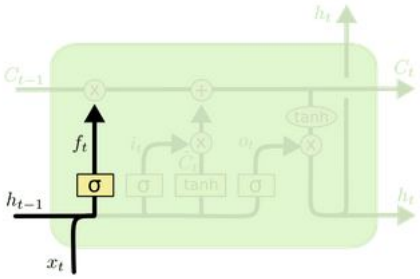


Figura 2.11: Estructura de una Red LSTM [25].

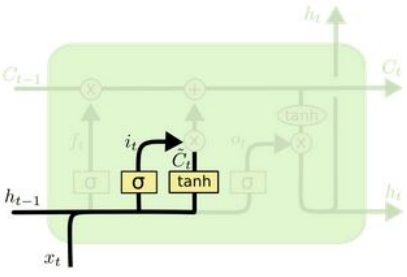
Las redes LSTM poseen una estructura básica como la que se observa en la Figura 2.11, en la parte superior del diagrama de la neurona LSTM, se puede observar cómo existe una línea horizontal que conecta las diferentes neuronas, como se menciona en el trabajo de [25], esta es la clave en el funcionamiento de la memoria a largo plazo en este tipo de redes, de manera que se evite el desvanecimiento de gradiente. Esta red se compone de tres compuertas principales, la del olvido, la de entradas externas y la de salida.



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Figura 2.12: Compuerta de olvido [25].

La primer compuerta en procesarse es la del olvido, que tiene la forma observada en la Figura 2.12, esta compuerta define que elementos deben ser eliminados de la celda de estados, esto lo realiza por medio de una multiplicación de los elementos por un valor de 0 y 1, el cual es asegurado por la función de activación sigmoidal a la que se somete la señal. Como lo menciona [22], un valor de 0 indicaría olvidar todo, mientras que 1 significaría seguirlo recordando.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Figura 2.13: Compuerta de entradas externas [25].

La segunda compuerta es la de entradas externas, es la que se encarga de valorar

que valores añadir a la memoria, su estructura se puede observar en la Figura 2.13. En la investigación de [25] se explica como esta compuerta se encarga por medio del vector i_t de decidir qué valores serán o no añadidos a la memoria, mientras que por otra parte el vector $\hat{C}_t$ almacena a los candidatos a ser incorporados. Finalmente definidos los valores que se van a agregar, la memoria se modifica como se observa en la Figura 2.14, donde el vector f_t multiplica directamente el estado pasado de la memoria, de manera que eliminará los elementos que no hacían un aporte y en se adicionarán los elementos establecidos en la compuerta de entradas externas, dando como resultado una memoria actualizada C_t.

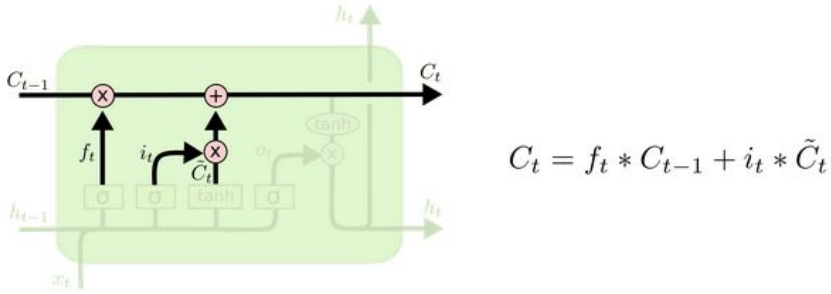


Figura 2.14: Actualización de la memoria [25].

Finalmente se define la compuerta de salida de la LSTM, en el trabajo de [25], como la que se observa en la Figura 2.15, en esta compuerta se calcula un vector, considerando la entrada y el estado oculto anterior, el cual permite definir cuales valores de la memoria serán utilizados en el calculo del nuevo estado oculto. Por otra parte, los candidatos son tomados directamente de la celda de memoria y son pasados por una función de activación tangente hiperbólica, ya que los estados ocultos se manejan en una escala de -1 a 1. Al multiplicar ambos vectores se obtiene el nuevo estado oculto para la siguiente neurona LSTM.

Como se puede observar en las ecuaciones presentes en las Figuras 2.12, 2.13, 2.14 y 2.15, se cuenta con matrices de pesos W, para cada una de las operaciones además de un sesgo b. Los valores de cada una de estas matrices son los componentes entrenables

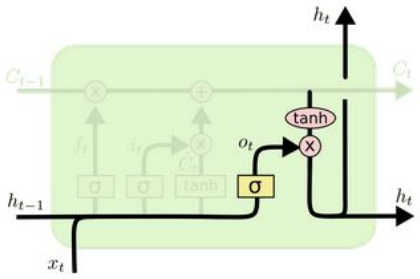


Figura 2.15: Compuerta de salida [25].

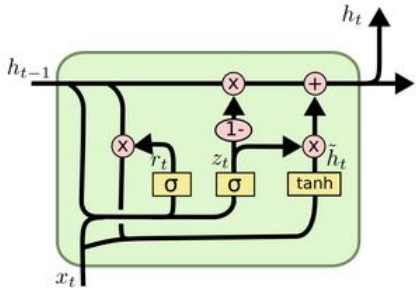
en la red LSTM, por lo que una vez llevado a cabo la propagación hacia atrás a través del tiempo, característica de las redes recurrentes, los valores de estos se actualizarán.

Como le menciona [26], en la actualidad las redes LSTM son muy utilizadas cuando se trata de redes recurrentes y aunque existen muchas variantes de la arquitectura, esta red es muy poderosa incluso con su arquitectura básica. Es una red ideal para estudio de secuencias de hasta 1000 unidades de tiempo sin problema, por lo que se han utilizado para resolver una gran variedad de problemas difíciles.

2.3.1.2 Gated Recurrent Unit (GRU)

La arquitectura GRU es una variación de las redes recurrentes, que al igual que la red LSTM solventan el problema de tener únicamente una memoria a corto plazo, a través de una celda de estado oculto, donde se almacena los datos como si fuera una memoria a largo plazo. Como se menciona en [27], la arquitectura GRU es una variante ligeramente más optimizada que a menudo ofrece un rendimiento comparable y es significativamente más rápido de calcular, debido a que esta tiene una mayor simplicidad que la LSTM.

La mayor diferencia entre la red LSTM y la GRU es el número de compuertas, ya que como se observa en la Figura 2.16, la red GRU carece de una compuerta de salida. Esta arquitectura cuenta con tan solo dos compuertas, la compuerta de reinicio dada por la función r_t y la compuerta de actualización dada por la función z_t. Y finalmente se da una actualización del estado oculto por medio de h_t. Pero no existe una compuerta



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Figura 2.16: Gated Recurrent Unit [25].

de salida como en el caso de la LSTM, se maneja el estado oculto como la celda de memoria.

La primera compuerta de esta arquitectura es la de reinicio, en esta se decide qué información de la memoria se debe olvidar. Toma como entradas el estado oculto anterior y la entrada y con base a estos decide que porción de información ya no le es útil. Una vez realizado esto tiene como salida a los posibles candidatos a seguir en la memoria. Luego sigue la compuerta de actualización, en la cual se determina que porción de la información se debe pasar al estado oculto próximo. Al igual que en la red LSTM, se utilizan funciones sigmoidales que oscilan entre 0 y 1, para darle a las compuertas en funcionamiento de válvulas completamente cerradas o abiertas.

2.4 Resumen

Un UAV es un dispositivo conformador por un sistema de propulsión, que se conforma por la batería, el ESC, el motor y la hélice. Además, cuenta con un controlador como componente principal del sistema de control, el cual también comprende los sensores y la antena de comunicación con el control remoto. Su funcionamiento se basa en un modelo que toma como entrada una posición o maniobra que alcanzar y como salida, la velocidad de los motores, de manera que, con diferentes velocidades en cada uno de los motores, puede lograr cada una de las maniobras de vuelo. Los UAV dispositivos propensos a fallar, tanto por causas externas como mal funcionamientos de los componentes electrónicos. En la Tabla 2.1, se presenta un resumen de estos fallos, de los detonadores que llevan a su aparición y las variables que permiten su detección.

Tabla 2.1: Detonadores y evidencias de los fallos que se pueden presentar en un sistema multirrotor

Fallo	Detonadores	Evidencia
Aflojamiento o rotura del soporte del motor	Vibraciones y movimientos en la estructura	Exceso de vibración Fluctuaciones en la velocidad
Desbalance de la hélice	Deformación plástica	Aumento de vibraciones Disminución de corriente
Aflojamiento o desprendimiento de la hélice	Mal ajuste de la hélice Altas velocidades	Aumento de vibraciones Disminución de corriente
Desgaste de rodamientos	Contaminación, corrosión, mala lubricación o una instalación inadecuada	Vibraciones con cierta frecuencia
Motor bloqueado	Obstrucción por contaminación Daño en rodamientos	Corrientes elevadas Calentamiento del motor
Batería dañada	Descargarla a menos de 3V Tasa de descarga elevada	Calentamiento de batería Caída de tensión
Mala operación del ESC	Deterioro por tiempo de uso	Altas corrientes Calentamiento del motor Calentamiento de batería
Desconexión de fases	Vibraciones Fases mal aseguradas	Calentamiento del motor Aumento en corriente
Desmagnetización del motor	Campos magnéticos externos Altas temperaturas	Aumento de torque Aumento de corriente
Motor dañado	Cortocircuito	Velocidad nula Aumento en vibraciones

La inteligencia artificial es una herramienta muy útil en la detección de anomalías. En problemas como la detección de patrones, en datos que tienen una naturaleza temporal, como los son los obtenidos por los sensores en un UAV, las redes recurrentes son una de las mejores opciones, sin embargo, las RNN tienen ciertas limitaciones cuando se trata de procesar largas cadenas de datos, en busca de solventar este problema se crearon variaciones a esta red como lo son la LSTM y la GRU. En la Figura 2.17, se puede observar la diferencia en la arquitectura de cada una de ellas.

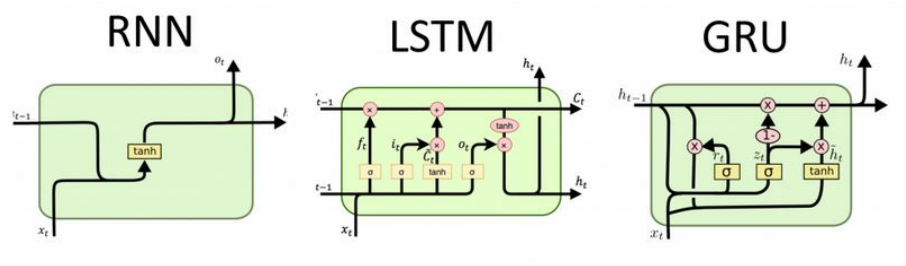


Figura 2.17: Arquitecturas de una red recurrente sencilla, una LSTM y una GRU[25].

Capítulo 3: Metodología

La mejor manera de resolver un problema es dividiendo las tareas principales en subtareas cada vez mas pequeñas. Por lo que en un inicio de planteó un objetivo general el cual fue dividido en cuatro específicos. En esta sección se desglosa cada uno de los objetivos en las diferentes a actividades, con las que se pueda llegar a una solución del problema, al cumplir con todas las subtareas. Además se presenta su relación con el proceso de diseño de ingeniería.

3.1 Enfoque metodológico

El diseño de este proyecto contempla el proceso de diseño de ingeniería, mostrado en la Figura 3.1. Este proceso sigue las etapas mencionadas en [28]. A continuación se describen las etapas de este proceso aplicadas en el desarrollo de este proyecto:

- **Realizar investigación de fondo:** Esta es la primera etapa de cualquier proceso de diseño, abarca las actividades de investigación que llevarán al desarrollo de una solución. Antes de resolver un problema este se debe definir primero, en el caso de este proyecto el problema se puede observar en la sección 1.2. Una vez que el problema es claro se debe levantar una lista de especificaciones de lo que se desea diseñar, está por lo general se lleva a cabo considerando los deseos del cliente o a características específicas definidas con base a una investigación teórica realizada

en esta etapa.

- **Generación y evaluación de conceptos:** Una vez que se tienen claras las especificaciones necesarias para el diseño y la investigación teórica que lo respalda, se puede proceder a la generación de conceptos, los cuales engloban las posibles soluciones o diseños que se podrían llegar a implementar. Este proyecto se divide en tres subsistemas: el sistema de adquisición de datos, el sistema de detección de fallos y finalmente el de corrección, el de detección es una solución únicamente de software mientras que los otros dos si cuentan con una parte de hardware, sin embargo, para todos se pueden implementar diversas soluciones. Una vez se han definido un cierto número de conceptos para el sistema, se debe tener un método para evaluarlos correctamente y así definir cual pasará a la siguiente etapa del proceso de diseño.

- **Desarrollo de conceptos:** Esta es la etapa en la que el concepto seleccionado se llevan a la etapa de desarrollo, se definen los elementos o sensores necesarios para la implementación de cada uno de estos subsistemas y se procede con su implementación.

- **Prueba del diseño:** Todo diseño debe tener una validación que asegure su correcto funcionamiento y confirme que se llevo a cabo un buen diseño. En esta etapa se debe primero definir de qué manera se llevarán a cabo estas pruebas, así que una rúbrica que permita definir si los resultados obtenidos cumplen con las expectativas. En el desarrollo de este proyecto se debe evaluar la funcionalidad del sistema de adquisición de datos y verificar que todos los componentes elegidos funcionen correctamente, además se debe verificar la funcionalidad de la herramienta computacional de detección y clasificación de fallos. En este caso no se planteó el desarrollo del sistema de corrección, por lo que no se tendrá que evaluar.

- **Mejorar diseño:** Si luego de las pruebas el diseño no cumple con las expectativas,

este puede caer en una etapa de rediseño, donde se proponen mejoras o se seleccionan otros conceptos, para luego realizar de nuevo las pruebas, hasta obtener un resultado satisfactorio. En el desarrollo de una herramienta computacional, como la del sistema de detección de fallos, es común que se apliquen varias modificaciones al programa hasta mejorar su funcionamiento. Además, para el sistema de recolección de datos se podrían seleccionar mejores componentes.

- **Documentar diseño:** Una vez que se ha llegado a un resultado satisfactorio en las pruebas, se procede a documentar el diseño, lo cual en este caso se refleja en el desarrollo de este informe, el cual contiene todos los pasos que llevaron al desarrollo de una solución, así como los resultados obtenidos con el diseño propuesto.

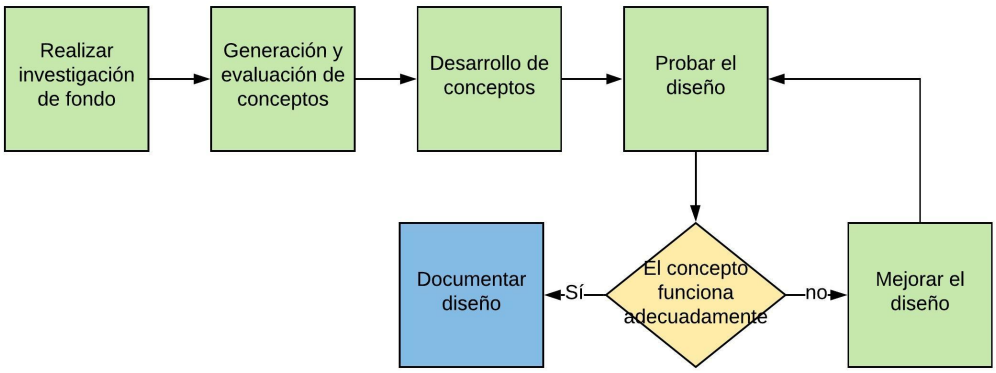


Figura 3.1: Metodología utilizada para el diseño de la solución.

3.1.1 Desglose de actividades por objetivo específico

Con el objetivo de simplificar el proceso de desarrollar una solución para el problema, se definieron una lista de actividades para cada uno de los objetivos específicos, de manera que se pudiera tener una mejor estructura a la hora de trabajar, sin perder nunca de vista el proceso de diseño en ingeniería que se utiliza para el desarrollo del proyecto.

3.1.1.1 Definir las circunstancias que pueden producir que el vehículo se desvíe de su funcionamiento normal y los variables de medición en las que se puedan ver reflejadas

En la Tabla 3.1 se pueden observar las actividades establecidas para la realización del primer objetivo. Este objetivo tiene un énfasis muy teórico y de investigación, por lo que la mayoría de sus resultados se pueden encontrar en el capítulo del marco teórico.

Tabla 3.1: Actividades del objetivo específico 1

Objetivo específico 1
1. Reunirse con los investigadores del GII con experiencia en volando los UAV.
2. Realizar un estudio del estado de arte sobre fallos en drones y las causas.
3. Identificar las variables que pueden brindar información sobre la aparición de fallos.
4. Desarrollar el algoritmo para la recolección de datos.
5. Desarrollo del PCB y elementos de sujeción para el sistema de recolección de datos.
Entregable: Diseño de un sistema de adquisición de datos, capaz de medir las variables en las que se pueden ver reflejados los posibles fallos.

La primer y segunda actividad permiten obtener la información de fondo para definir claramente el problema y empezar a generar especificaciones de lo que se espera de una solución. Luego en la tercera actividad se empieza a profundizar un poco más, en lo que serán las variables de medición de un sistema de adquisición de datos, una vez que se cuenta con esta información es posible pasar a la generación de conceptos para este diseño. Finalmente, cuando se ha seleccionado una opción se puede proceder a las últimas dos actividades, las cuales implican escoger los sensores o componentes que se utilizarán, diseñar un algoritmo de recolección de datos, un PCB y elementos de sujeción para el montaje sobre el UAV.

3.1.1.2 Diseñar un modelo de predicción y detección con base en los resultados de la etapa anterior

Las actividades en las que se desglosa el segundo objetivo se pueden observar en la Tabla 3.2. El segundo objetivo específico es uno de los más importantes, ya que abarca la mayor parte del trabajo que dará solución al problema. Se propone como resultado, una vez finalizadas todas las actividades, obtener un algoritmo para la predicción, detección y clasificación de fallos.

Tabla 3.2: Actividades del objetivo específico 2

Objetivo específico 2
1.Definir el formato de los datos con los que trabajará la herramienta.
2. Investigar sobre algoritmos de predicción que se puedan aplicar a los datos obtenidos.
3. Diseñar el algoritmo de predicción y detección.
4. Establecer los parámetros variables en las pruebas del algoritmo.
Indicador: Algoritmo que para la predicción, detección y clasificación de fallos.

Con el sistema de recolección de datos se pretende obtener datos de vuelo donde se provocarán algunos de los diferentes fallos establecidos en la Tabla 2.1. En caso de que los mismos no se puedan provocar o del todo no se puedan recolectar datos, se propone la generación de datos sintéticos, basados en la teoría obtenida en el objetivo especifico 1 y en datos recolectados por el GII en pruebas de vuelo realizadas en el pasado. Estos datos tendrían otro formato, dado que fueron recolectados con otro sistema de adquisición, sin embargo, se pueden procesar de manera que se genere un conjunto de datos que si corresponda al esperado del sistema de recolección.

La actividad 2 consiste principalmente en una investigación teórica sobre las técnicas o algoritmos utilizados comúnmente en la predicción y detección de anomalías, los

resultados de esta actividad pueden encontrarse en la sección 2.3 del marco teórico, donde se desarrolla concretamente el uso de la inteligencia artificial en la detección de anomalías.

La tercera actividad implica directamente el diseño de la herramienta de predicción y detección de fallos, existen diferentes algoritmos que podrían ser utilizados para la resolución de este problema, por lo que se generarán un par de conceptos a evaluar, para la selección de uno en concreto. Finalmente se establecerán los parámetros que se podrán variar en las pruebas de validación, con el objetivo de mejorar los resultados obtenidos.

3.1.1.3 Establecer los protocolos de control para dar respuesta a los diferentes fallos que se pueden presentar durante el vuelo

Las actividades correspondientes al objetivo específico 3 se pueden encontrar en la Tabla 3.3. Una vez completadas todas las actividades se pretende contar un claro diagrama de flujo que indique de qué manera se corregirán los fallos detectados por el modelo diseñado en el objetivo específico 2.

La primera actividad de este objetivo establece una investigación previa, sobre métodos de corrección de fallos utilizados en octocóptero. Una vez establecidos los fallos que se pretenden detectar, se debe investigar la mejor manera para poder corregirlos, ya sea por medio de una investigación teórica o consultándolo directamente con personas que tengan experiencia en el campo.

Una vez se haya investigado sobre técnicas de corrección de fallos, se propone como segunda actividad el agrupar los fallos en categorías, de manera que se establezca si una corrección se puede aplicar para más de un fallo a la vez.

Finalmente las últimas dos actividades condensan los resultados de este objetivo. La tercera actividad propone la confección de un diagrama de flujo, que permita una mejor visualización de cómo se llevarán a cabo las correcciones de los fallos. La cuarta

Tabla 3.3: Actividades del objetivo específico 3

Objetivo específico 3
1. Investigar sobre los protocolos comunes de corrección que se pueden implementar en un octocóptero.
2. Agrupar los diferentes tipos de fallos detectados o predichos en categorías con un protocolo específico.
3. Desarrollo de un diagrama de flujo para la visualización de los protocolos de control.
4. Diseñar el algoritmo que recibirá la salida del modelo de predicción y accionará el protocolo de control
5.Definir los componentes necesarios para la implementación de un sistema de corrección de fallos.
Indicador:Diagramas de flujo y pseudocódigo de los protocolos de control.

actividad propone realizar el diseño de un algoritmo, que interaccionaría con la salida del modelo de predicción y detección para tomar las decisiones de control y respuesta a fallos. Finalmente se propone como ultima actividad definir los componentes necesarios para la implementación de un sistema de corrección de fallos.

3.1.1.4 Evaluar la funcionalidad del diseño por medio de pruebas de campo real o simulaciones

En la Tabla 3.4 se pueden observar las actividades del último objetivo específico. Estas actividades giran en torno a la validación de la herramienta de predicción, detección y corrección de fallos diseñada.

La primera actividad de este objetivo permite establecer el método de validación que se utilizará para los sistemas diseñados. Posterior a eso se inicia con la validación del sistema de recolección de datos, se debe verificar que todos los sensores funcionen de manera correcta, que los datos recolectados correspondan a los esperados bajo un

Tabla 3.4: Actividades del objetivo específico 4

Objetivo específico 4
1. Seleccionar el método de validación de acuerdo con los resultados obtenidos en las etapas anteriores.
2. Verificar el correcto funcionamiento del sistema de adquisición de datos.
3.Generación de datos sintéticos para el desarrollo del modelo predictivo.
4. Llevar a cabo las pruebas de los algoritmos de predicción, detección y corrección.
5. Establecer el porcentaje de éxito obtenido en las pruebas.
Indicador: El modelo propuesto es capaz de detectar al menos un 60% de los fallos y responder en busca de una solución.

funcionamiento normal.

En la tercera actividad se da la generación de datos sintéticos de vuelo para las pruebas del algoritmo de predicción y detección. Finalmente se realizan las pruebas de este, realizando cambios en los parámetros establecidos hasta obtener altos valores de precisión. Una vez que se ha establecido un modelo final se procede a establecer un porcentaje de éxito, el cual como lo dice el indicador se espera que sea mayor a un 60% de éxito en sus predicciones.

Capítulo 4: Generación y evaluación de conceptos

En este capítulo detalla las primeras etapas del proceso de diseño establecidas en la metodología, aquí se definen los subsistemas en los que se divide el diseño final, de manera que se puedan generar los conceptos de una posible solución final, así como la evaluación de estos, para seleccionar el que finalmente se va a desarrollar.

4.1 Definición de requerimientos para el diseño

El proyecto se propone inicialmente como un sistema de para la predicción, detección y corrección de fallos, sin embargo, se puede dividir en subsistemas para comprender facilitar el proceso de diseño, y tener una idea mas clara de como trabajarlo. En la Figura 4.1 se puede observar un pequeño diagrama que muestra los subsistemas que conforman el producto final.

Esta subdivisión la componen el sistema de adquisición, compuesto por el conjunto de sensores que medirán las variables seleccionadas, en las que se vean reflejados cambios durante la presencia de un fallo. Luego estará el algoritmo de detección, el cual sería activado por el microcontrolador encargado de la adquisición y sería el programa encargado de la interpretación de los datos recolectados. Finalmente, está el subsistema de corrección de fallos, el cual se encarga de dar respuesta al fallo, así como comunicar al operador.

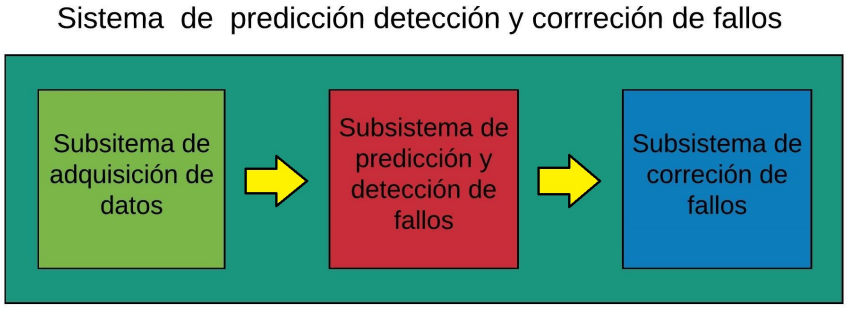


Figura 4.1: Descripción del sistema que se pretende diseñar.

Tabla 4.1: Lista de requerimientos preliminar del diseño

Num.	Subsistema	Necesidad	Importancia
1	R.Datos	El dispositivo es lo más pequeño posible.	4
2	R.Datos	Tiene la capacidad de medir altas corrientes.	4
3	R.Datos	La toma de datos se realiza a una alta velocidad.	4
4	R.Datos	Se puede alimentar de la batería.	3
5	R.Datos	El dispositivo no interrumpe el funcionamiento del vehículo.	5
6	R.Datos	El sistema puede leer valores analógicos.	4
7	R.Datos	Se puede colocar cerca del sistema de propulsión.	4
8	R.Datos	El rango de medición de vibraciones es adecuado.	4
9	R.Datos	Es capaz de medir velocidad en un rango establecido.	3
10	A. Detección	Permite analizar series de datos de naturaleza temporal.	4
11	A. Detección	Permite detectar patrones o irregularidades en los datos.	4
12	A. Detección	Tiene la capacidad de analizar rapidamente los datos.	5
13	S. Corrección	Es capaz de comunicar al usuario sobre el fallo.	4
14	S. Corrección	Integración con el sistema de recolección de datos.	3
15	S. Corrección	Permite cortar la alimentación del motor.	4
16	S. Corrección	Utiliza comunicación inalámbrica.	5
17	S. Completo	Los materiales resisten los esfuerzos presentes	4
18	S. Completo	No suma un peso significativo al vehículo.	4

1. La función es indeseable. No consideraría un producto con esta función.
2. La función no es importante, pero no me importaría tenerla.
3. Sería bueno tener esa función, pero no es necesaria.
4. La función es altamente deseable, pero consideraría un producto sin ella.
5. La función es de importancia crítica. No consideraría un producto sin esta función.

Una vez establecida estas subdivisiones se generó una tabla de especificaciones que se debían contemplar para el diseño, las cuales se pueden observar en la Tabla 4.1, donde se presentan divididas en subsistemas y se muestra la importancia de cumplir

Tabla 4.2: Especificaciones objetivo del producto.

Num.	Métrica	Unidad	Valor
1	Las dimensiones del soporte	mm	<200x100
2	Mayor a la corriente máxima del ESC	A	>40
3	Se toman al menos 10 datos por segundo	Hz	>10
4	Permite alimentación de una batería lipo de 3 celdas	V	12V
5	Los componentes no obstruyen el funcionamento normal del sistema de propulsión	—-	—-
6	Cuenta con un convertidor analogico digital de al menos un cierto numero de bits.	bits	>8
7	Es montable sobre el brazo del UAV	—-	—-
8	Mide vibraciones dentro del rango	m/s^2	-9.81-9.81
9	Mide velocidades dentro del rango	rpm	0-10.000
10	Los datos provienen de sensores que miden variables que varían en el tiempo	—-	—-
11	El algoritmo es capaz de relacionar los datos entre sí	—-	—-
12	Es capaz de tomar decisiones rápidamente	s	<1
13	La información sobre fallos se comunica al usuario	—-	—
14	La corrección es lograda en conjunto con el sistema de adquisición de datos	—-	—-
15	El sistema es capaz de desconectar el motor en caso de ser necesario	—-	—-
16	La comunicación inalámbrica se da a grandes distancias	m	>100
17	El factor de seguridad de del diseño es alto	—-	>1,5
18	El peso total del sistema de adquisición y corrección no es muy significativo.	Kg	0.2

con cada una de estas. Por otra parte, se presenta la Tabla 4.2 donde se muestran las métricas utilizadas para velar por el cumplimiento de cada especificación, así como las unidades y valores en caso de tenerlas. Para la confección de ambas tablas se utilizó tanto la información brindada por el GII, como la de algunos datos obtenidos de la investigación de fondo desarrollada en el capítulo 2.

4.2 Generación de conceptos

Una vez establecidas las principales especificaciones, se procede a la generación de distintos conceptos, de manera que se puede evaluar cual es el mejor y así llevarlo a la etapa de desarrollo. Para este proceso de confeccionó la Tabla 4.3, donde se muestran

distintas ideas para algunos de los rasgos más importantes en el diseño de la solución. La tabla presenta 2 conceptos para cada una de las características, de manera que al final se seleccionará la mejor combinación de de estas.

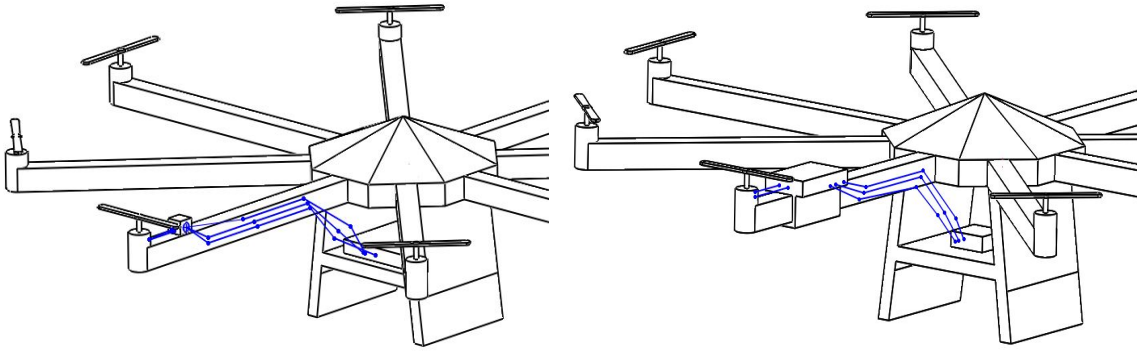
Tabla 4.3: Generador de conceptos en las características principales

Distribución	Algoritmo de detección	Corrección de fallos	Comunicación
Un solo PCB	Detecta la superación de umbrales preestablecidos	Señal que indique al programa de vuelo que debe detener el motor	A través de miniPC existente
Dos PCB donde solo uno va en el brazo	Detecta patrones analizando agrupaciones de datos	Interrumpir directamente las líneas de alimentación del motor	Nuevo componente exclusivo para comunicar el fallo

A continuación, se describe cada un de las posibles características propuestas en la Tabla 4.3, para la generación de conceptos, de manera que se puede visualizar mejor en que consiste cada, para luego proceder a la conformación y evaluación de conceptos teniendo todas las consideraciones presentes.

- **Distribución:** El rasgo de distribución considera las especificaciones de tamaño y forma del diseño, así como del montaje de este. En la Figura 4.2 se pueden observar los bosquejos de las dos variaciones propuesta para este concepto. El primero consiste en el desarrollo de un solo PCB el cual iría colocado sobre la carcasa central del UAV, en esta se encontrarían todos los componentes y algunos pines que servirían como entradas de los sensores que deban ir sobre el brazo o cercanos al motor. Por otra parte, el segundo concepto contempla la confección de dos PCB, en este caso se colocaría sobre el brazo todo el conjunto de sensores que recogen información del motor y se comunicaría por medio de una faja de cables a la segunda placa, donde estaría el microcontrolador encargado del procesamiento de los datos.

- **Algoritmo de detección:** Para el algoritmo de detección de fallos se proponen dos conceptos que a grandes rasgos funcionarían como se observa en el diagrama de flujo presentado en la Figura 4.3, donde le primer concepto se presenta al



(a) Una placa **(b)** Dos placas

Figura 4.2: Conceptos para la distribución de componentes.

lado izquierdo y el otro al lado derecho. Para el primer concepto se plantea un algoritmo para el que se debe hacer todo un estudio previo y análisis de los datos que permita establecer umbrales para cada una de las mediciones, de manera que si estos son superados se inicie un protocolo de recolección. Por otra parte, se plante un segundo algoritmo que no se enfoque en valores umbrales, si no que entrene con secuencias datos y aprenda patrones para cada tipo de fallo y una vez que reconozca uno de igual forma inicie el protocolo de recolección.

- **Corrección de fallos:** La corrección de fallos hace referencia a la forma en la que se activa la corrección, se contempla un concepto mucho más complejo en el que se altera el programa de vuelo del UAV, de manera que cuando se active la alerta de fallo se eleven las velocidades del motor que simultáneamente se apagaría, de esta manera la transición no sería abrupta. Por otra parte, se considera un segundo concepto que desconectaría directamente la alimentación del motor con algún tipo de relé. Estos conceptos se resumen en la Figura 6.5, uno a cada lado del diagrama de flujo.

- **Comunicación:** Una vez que se detecta un fallo, otra de las cualidades importantes del sistema a diseñar es la capacidad de comunicar este fallo al usuario, para esto se proponen dos conceptos diferentes, el primero plantea utilizar la comunicación

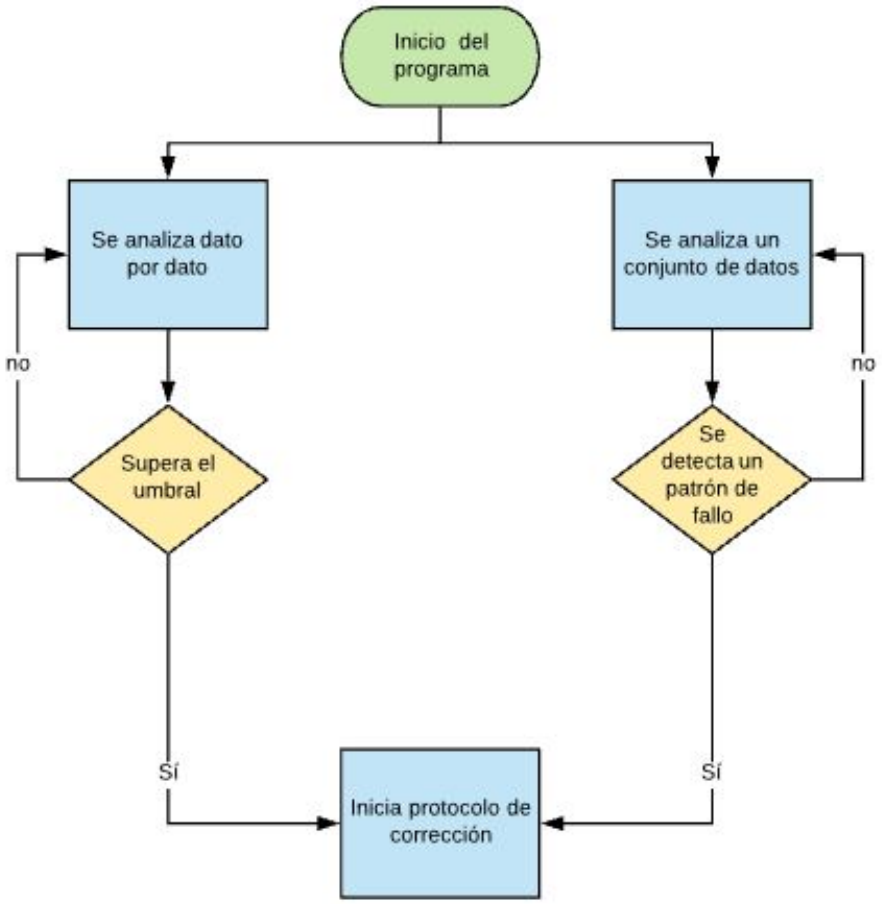


Figura 4.3: Conceptos para el algoritmo de detección.

ya existente entre el mini PC a bordo y una computadora tierra, los cuales se comunican por medio de radiofrecuencia. Por otra parte, también se propone como solución la implementación de un sistema de comunicación independiente, que se conecte directamente a la salida del microcontrolador que realiza la detección y envié el mensaje al operador, se considera la posibilidad de que se realice por medio de un servidor de manera que el usuario pueda ver el estado hasta en su teléfono. En la Figura 4.5 se muestra un breve diagrama que describe ambos conceptos.

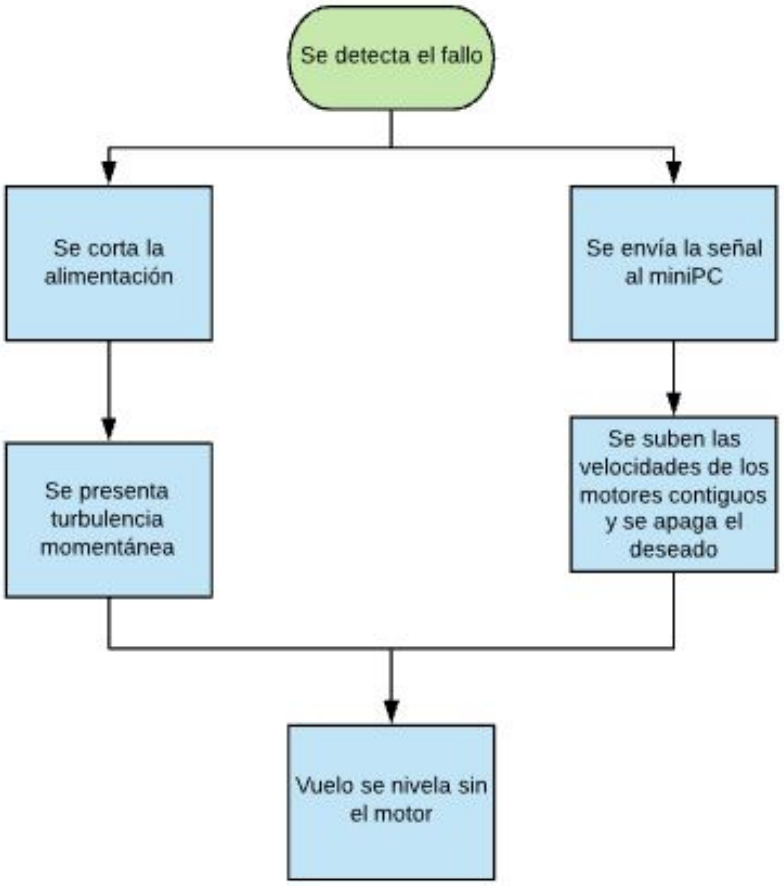


Figura 4.4: Conceptos para la activación de la corrección.

4.3 Evaluación de conceptos

Una vez que se han presentado los conceptos se procede a la evaluación de estos, esto se realiza evaluando cada uno de acuerdo con ciertos criterios de interés de manera que se pueda seleccionar el que mejor calificación obtenga al final. En este caso se analizaron cuatro características distintas y se decidió generar una tabla de evaluación para cada una de ella, debido a que son muy diferentes entre sí como para evaluarlas bajo los mismos criterios.

En la Tabla 4.4 se presenta la evaluación de los conceptos de distribución. Si bien condensar todo en una sola placa puede facilitar la manufactura y hasta reducir los costos por el ahorro de material, no resulta ser el mejor concepto. Este concepto puede

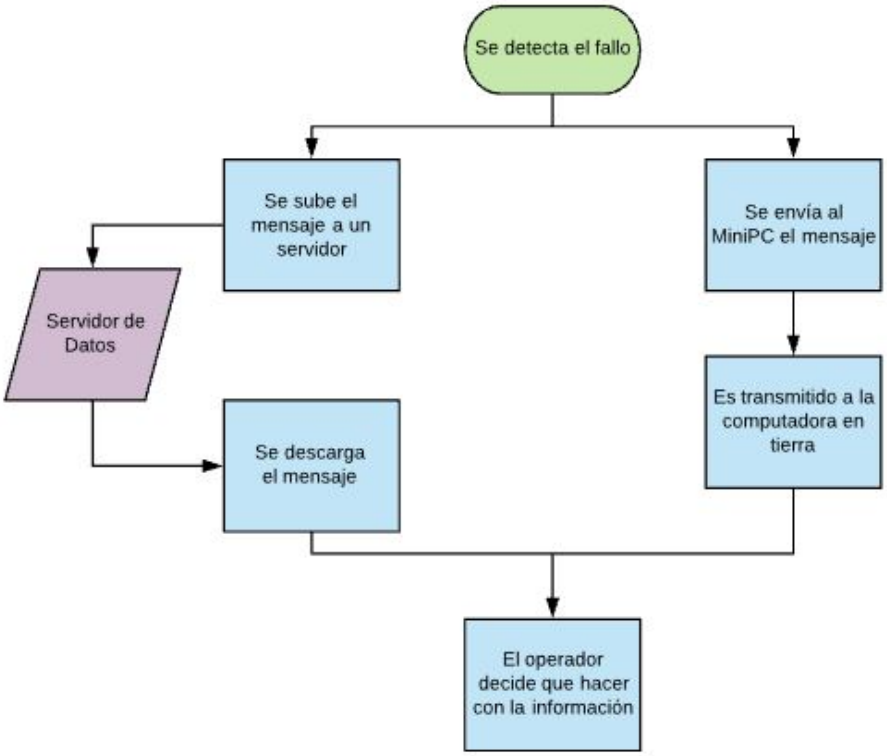


Figura 4.5: Conceptos para la comunicación del fallo.

Tabla 4.4: Evaluación del concepto de distribución

Criterio de Selección	Una sola placa central	Dos placas distribuidas
Facilidad de manufactura	+1	-1
Costo de manufactura	+1	-1
Mejor distribución de peso	-1	+1
Menos cantidad de cables	-1	+1
Menor número de soportes	0	0
Facilidad de conexión	-1	+1
Mejor prevista a múltiples motores	-1	+1
Suma Neta	-2	+2

terminar utilizando más cableado, ya que todos los sensores que deban ir cerca del motor tendrán que trasladarse allí, lo cual puede dificultar las conexiones. Además, se destaca que el concepto de dos placas considera mejor la posibilidad de implementarse luego a todos los demás motores, lo cual sería más complicado con una sola placa en el centro.

En la Tabla 4.5 se puede observar la evaluación de los conceptos para el algoritmo de detección, donde el que obtiene un mayor puntaje es el que se enfoca en la detección de

Tabla 4.5: Evaluación del concepto de algoritmo de detección

Criterio de Selección	Superación de umbrales	Detección de patrones
Menor de trabajo de fondo	-1	+1
Facilidad de programación	0	0
Lidiar con errores en lectura	-1	+1
Velocidad de procesamiento	+1	-1
Requiere mayor recolección datos	0	0
Permite analizar dependencias temporales	-1	+1
Suma Neta	-2	+2

patrones. Este concepto resulta mejor ya que requiere menos trabajo de fondo, ya que no se deben estudiar los datos para establecer valores umbrales, además puede trabajar mejor con señales ruidosas, ya que analiza series de datos y no datos individuales, por lo que, en caso de realizarse una lectura de un valor erróneo, no se calificaría como fallo. La detección de patrones permite trabajar mejor con datos con dependencias temporales, propio de los que se espera recolectar en este proyecto.

Tabla 4.6: Evaluación del concepto de corrección de fallos

Criterio de Selección	Interrupción directa	Interrupción mediante sistema
Menor costo de componentes	-1	+1
Facilidad de programación	+1	-1
Menor turbulencia	-1	+1
Mayor velocidad de actuación	+1	-1
Mayor independencia del sistema de control	+1	-1
Suma Neta	+1	-1

Otro par de conceptos evaluados fueron los de la corrección de fallos, la evaluación de estos se presenta en la Tabla 4.6. Como se observa se calificó como mejor concepto para la corrección de fallos a la interrupción directa. Este se destaca principalmente por ser el método con una implementación más fácil, a costa de pagar un poco más en componentes, además su mejor característica es que permite una actuación casi instantánea una vez detectado el fallo. Si bien este tipo de desconexión del motor

provocaría una turbulencia, esta sería corregida rápidamente por el sistema de control del vehículo y no representaría un mayor riesgo.

Tabla 4.7: Evaluación del concepto de comunicación de fallos.

Criterio de Selección	Transmisión por canal existente	Transmisión por otra vía
Menor costo de componentes	+1	-1
Facilidad de programación	-1	+1
Versatilidad para recepción	-1	+1
Mayores distancias	+1	-1
Velocidad de transmisión	+1	-1
Suma Neta	+1	-1

Finalmente, en la Tabla 4.7 se presenta la evaluación del último par de conceptos, los cuales estaban relacionado con la comunicación de fallos al operador. En este caso el concepto ganador fue el de la transmisión por el canal existente, sobre todo por el bajo costo en componentes, debido a que ya se cuenta con ellos, además de que es un sistema que logra enviar información a largas, lo cual es sumamente necesario cuando se vuela un UAV. Además, debido a tenerse una comunicación por radio frecuencia esta es sumamente rápida ya que solo se debe enviar el mensaje, no depende de factores de conexión, como si se estuviera trabajando con redes de internet. Si bien su programación puede ser más compleja ya que hay que hacerlo directamente sobre la Mini PC, no se destaca como un factor decisivo.

Una vez que se ha seleccionado el mejor concepto para cada característica se unen para formar el concepto final, que es el conjunto de todos, el cual pasará a la etapa de desarrollo en capítulos posteriores. En este caso se decidió desarrollar un sistema con dos placas PCB la cual tendrá un algoritmo de detección que será capaz de la detección de patrones y contará con una interrupción directa a la alimentación del motor, la cual se activará una vez que se haya detectado un fallo crítico. Finalmente comunicará la presencia de un fallo al operador por medio de la vía red existente entre la Mini PC a bordo y una computadora en tierra.

Capítulo 5: Descripción detallada de la solución

En este capítulo se describen la solución implementada para el problema, se divide en tres secciones principales. Primeramente, se describe como se desarrollo el sistema de adquisición de datos, desde la selección de las variables de medición hasta la confección de los PCB y algoritmo de recolección de datos. En la segunda parte se describe la red neuronal implementada para la predicción y detección de fallos, así como una descripción de los datos que utiliza para su entrenamiento. Finalmente se presenta una sección sobre el sistema que se encargaría de la corrección de fallos, así como los diagramas de flujo que permiten visualizar mejor como será esta respuesta.

5.1 Sistema de recolección de datos

Para llevar a cabo el diseño del sistema de adquisición de datos, se ha llevado a cabo una categorización y conceptualización de todos los fallos que podrían presentarse en el multirrotor(sobre todo en el sistema de propulsión), así como las variables en que dichos fallos se ven reflejados, lo que permite detectarlos. Por medio de la información condensada en la Tabla 2.1 y la proporcionada por [29], se construyó el esquema observado en la Figura 5.1. En la parte superior de esta figura se presentan los posibles

fallos a detectar, mientras que en la parte inferior se encuentran las variables o formas en las que estos se evidencia y finalmente en las celdas rojas se presentan los peores escenarios, donde la presencia de estos fallos desencadena la pérdida de algún elemento del UAV.

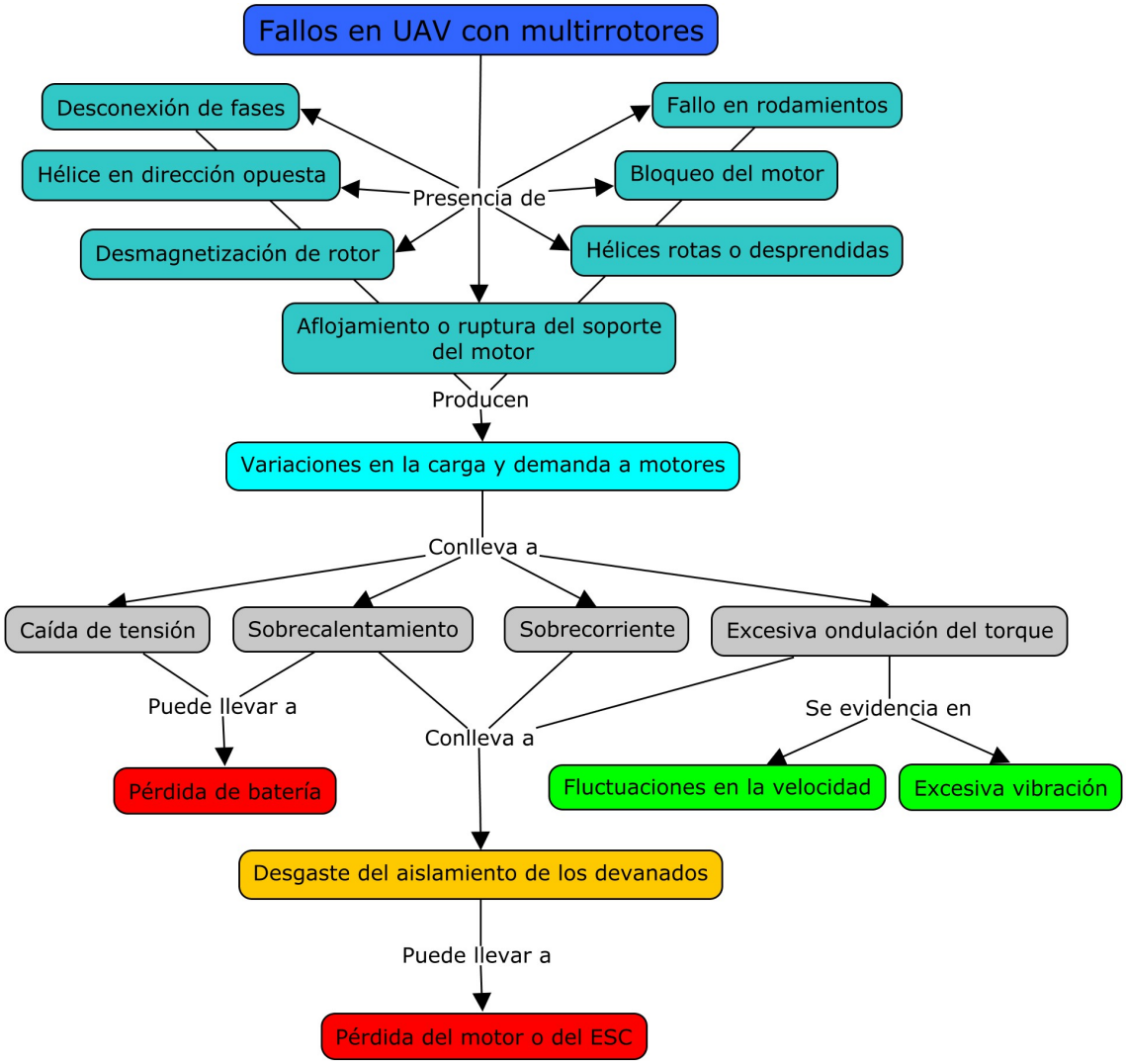


Figura 5.1: Mapa de causas y fallos.

Basado en la Figura 5.1, se define que las variables de medición que puede aportar datos relevantes en el desarrollo de la herramienta de detección y predicción son los siguientes:

- **Tensión de la batería**: Esta variable permite conocer el estado de la batería

además de que es un dato necesario para conocer la potencia que demanda el motor.

- **Temperatura**: Se debe medir tanto la temperatura de la batería como la del motor, ya que en ambos componentes las altas temperaturas pueden ser factores críticos en la aparición de fallos.

- **Corriente**: Es una variable muy importante, que proporciona información sobre la carga y la potencia eléctrica, además muchos componentes tienen especificaciones de diseño que giran en torno a limites de corriente.

- **Velocidad**: Medir las fluctuaciones en la velocidad puede proporcionar información sobre la presencia de alguna irregularidad, además junto al torque pueden dar datos de la potencia mecánica.

- **Vibración**: La vibración es una de las variables de medición más importante, debido a que todos los fallos o irregularidades que se presentan en vuelos de UAV, afectan directamente la estabilidad de la nave, por lo que monitorear la vibración en los tres ejes es indispensable.

Se contempló medir directamente los parámetros de torque y fuerza de empuje, sin embargo, el costo de estos sensores puede llegar a ser elevado, por lo que se descartó esta medición, contando con que las mediciones de tensión, corriente y velocidad puedan brindar información suficiente en cuanto a potencia y torque.

5.1.1 Especificaciones y sensores utilizados

Para el diseño de un circuito encargado de la recolección de datos se deben conocer las especificaciones de los componentes del UAV sobre el que se van a montar, sobre todo del motor, del ESC y la batería, ya que serán los que directamente se interrumpirán para la recolección de datos. En cuanto a la batería, se utilizan dos en paralelo de

5000mAh y tres celdas, lo que quiere decir que se tiene una tensión nominal de 11.1V. Las especificaciones del motor y del ESC se pueden observar en las Tablas 5.1 y 5.2 respectivamente.

Tabla 5.1: Especificaciones del motor utilizado[30].

Motor BLDC NTM Prop Drive 28-26	
KV	1000 rpm/V
Número de polos	6
Corriente máxima	15A
Empuje máximo	632g
Potencia máxima	176 W @ 11.1V (3S) / 235W @ 15V (4S)
Tensión máxima	15V

Tabla 5.2: Especificaciones del ESC utilizado [31].

XRotor 40A	
Corriente máxima	40A
Pico de corriente máxima (10 s)	60A
Tensión mínima	7.4V
Tensión máxima	22.2V

Una vez definidas las variables de medición y tomando en cuenta las especificaciones de los elementos del motor, se procedió a la selección del microcontrolador y los sensores de medición. Para la elección de estos componentes se buscó aquellos sensores que cumplieran con los requerimientos, pero sin dejar de contemplar el costo de estos. A continuación, se detallan los elementos seleccionados:

- **Microcontrolador:** Para cumplir con la función de microcontrolador se plantearon por parte del cliente dos sistemas disponibles en su momento,un Arduino UNO y una Rasperry PI. Una vez realizado un análisis decidido utilizar la placa Arduino UNO, fundamentalmente por dos motivos, busca un sistema de control, el Arduino es un controlador, ya que sus características de software y hardware están orientadas hacia el control. Independientemente de que la Raspberry se pueda utilizar como controlador, realmente es una computadora, lo que conlleva a una serie

de diferencias en su forma de operación. Además, para el uso de una Raspberry se tendrían que adquirir convertidores analógico digital, que el Arduino si posee en su placa. Adicional a esto para la selección del Arduino se considero que por parte del cliente este proyecto se contempla como una prueba de concepto, que posteriormente puede ser mejorada con otras soluciones comerciales.

A continuación se detallan algunas de las características principales de esta placa de microcontrolador, tomados de [32]:

- La placa cuenta con el microcontrolador ATmega328P.
- Su voltaje de operación es de 5V.
- Cuenta con 14 pines digitales y 6 analógicos.
- La memoria flash es de 32 KB
- Su frecuencia de reloj es de 16 MHz.

- **Sensor de corriente:** Para la medición de corriente se adquirió el transductor de corriente HXS 50-NP. Con respecto a la información de su hoja de datos [33], entre su datos más importantes se destaca que su corriente nominal es de 50A sin embargo su rango máximo es de ±150A. Se alimenta con 5V y el valor de tensión medido puede ser convertido a unidades de corriente con la ecuación 4.1. En la Figura 5.2 se puede observar este sensor.

$$I_{Medida} = \frac{|2.5 - V_{medida}| * I_{Nominal}}{0.625} \tag{5.1}$$

- **Sensor de velocidad:** Se optó por utilizar el Brushless Motor RPM Sensor V2, de la compañía Eagle Tree, debido a que ya se contaba con uno en el laboratorio del GII. Como se menciona en [34] este sensor se alimenta con 5V [34] y su funcionamiento consiste en dos cables que interrumpen dos de las fases del motor

Figura 5.2: Transductor de corriente HXS 50-NP [33].

sin escobillas, de manera que lee la frecuencia con las que estas conmutan, y de esta manera se puede calcular la velocidad en RPM con la ecuación 4.2. En la Figura 5.3, se observa un diagrama de como se monta este sensor.

$$V_{Medida} = \frac{Frecuencia * 60}{Numero\,de\,Polos} \tag{5.2}$$

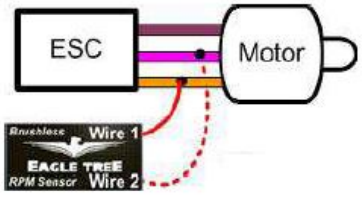


Figura 5.3: Sensor de velocidad EagleTree RPM sensor [34].

- **Sensor de temperatura:** Para esta medición se optó por conseguir los sensores LM35DZ, que según su hoja de datos [35], puede medir en un rango de 0 a 100 °C y su tensión de alimentación puede ir de 4 a 30V. La conversión de voltios a grados centígrados esta dada por la relación 10 mV/°C.

- **Sensor de tensión:** Dado que el Arduino UNO puede realizar lecturas de tensión que estén en el rango de 0 a 5, se puede leer la tensión de la batería con un divisor de tensión de resistencias de valores conocidos, de manera que se acondicione la escala. En este caso, dado que la tensión máxima que podrían presentar las baterías de tres celdas es de 12.6V, se optó por utilizar una resistencia de 4.7kΩ y

una de 10kΩ, de manera que la lectura de 12.6V sería vista por el Arduino como un valor de 4V.

- **Sensor de vibración:** Los sensores más utilizados en la medición de vibración son los acelerómetros, en este caso se optó por el MMA8451,que como dice su hoja de datos [36], su alimentación puede ser con 3 o 5 V y opera en un rango desde ±2g hasta ±8g. La unidad de un g hace referencia a la aceleración de $9.81\,m/s^2$. La comunicación con este sensor se da por un protocolo I2C. En la Figura 5.4 se puede observar el acelerómetro adquirido para medir vibración.

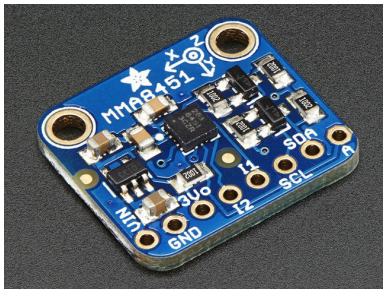


Figura 5.4: Acelerómetro para medir vibración MMA8451 [36].

Adicional a estos sensores se adquirió un modulo de tarjeta microSD para Arduino, el cual permite almacenar los datos recolectados en un archivo de texto, de manera que luego se pueda trabajar con ellos con mucha más facilidad, al menos para la etapa del entrenamiento.

5.1.2 Algoritmo de recolección de datos

Una vez seleccionados todos los elementos del sistema de recolección de datos, se confeccionó el programa para ser cargado al Arduino UNO. Este algoritmo es el encargado de inicializar cada uno de los sensores, además de recolectar y procesar cada uno de los datos provenientes de estos, para finalmente almacenarlos en las tarjeta microSD.

En la Figura 5.5 se puede observar el diagrama de flujo que sigue el ciclo del algoritmo de recolección. Fuera de este ciclo existen dos funciones independientes,

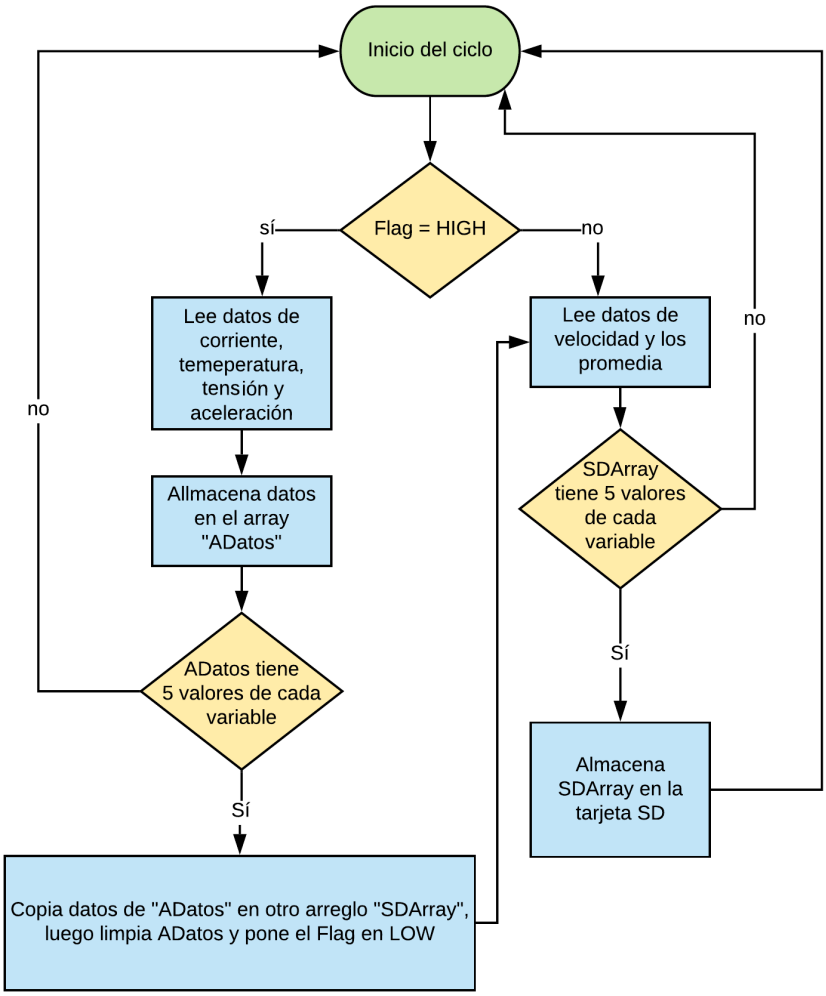


Figura 5.5: Diagrama de flujo del algoritmo de recolección de datos.

una es aquella en la que se inicializan librerías y variables para el funcionamiento del programa. La otra es una función de interrupción, en la cual se pone en HIGH a la variable Flag, que como se observa en el diagrama, es la primera variable consultada al inicio del ciclo. La interrupción de este programa se lleva a cabo con el TIMER2 del Arduino UNO, la misma se activa cada vez que este temporizador llega a una cuenta de 33 milisegundos, esto se realiza con el fin de que la recolección de datos se de con una frecuencia conocida.

De manera general el programa lee las señales de tensión provenientes de cada uno de los sensores, los procesa con alguna ecuación de ser necesario, para que tengan el

formato esperado, y los almacena en una variable tipo array, con el formato en el que serán almacenados en la tarjeta microSD. Como se observa en el diagrama, la medición de la velocidad se realiza de manera independiente, esto se debe a que esta medición utiliza el TIMER1 del Arduino, el cual entorpece la medición del TIMER2, ya que cuando se atiende una interrupción los timer se detienen y esto dañaba el dato de velocidad. La velocidad es medida con un timer ya que lo que se debe medir es la frecuencia de una señal cuadrada, esto se realiza contando la cantidad de pulsos de reloj que caben en una onda completa de la señal recibida desde el sensor de velocidad. Dado que la medición de frecuencia no suele ser muy precisa, se toman 5 datos y luego se promedian.

Abrir, escribir y guardar archivos en la microSD suele ser un proceso que toma bastante tiempo y en ocasiones variable. Es debido a esto que se tomó la decisión de almacenar los datos en un primer arreglo y una vez que se hubieran recolectado 5 datos se procediera a generar una copia de este arreglo, el cual sería guardado en la tarjeta microSD. De esta manera el array que se carga de datos en cada interrupción es independiente del que se guarda en la tarjeta, esto evita que se generen problemas de datos repetidos o de que no se tomen suficientes datos debido a la duración de el proceso de guardado. Para más información sobre el funcionamiento de este algoritmo se puede consultar el Anexo 1, donde se adjunta el código completo.

5.1.3 Diseño del PCB y montaje

Una vez se contaba con un sistema de recolección de datos funcionando en una placa de pruebas, se debía diseñar un circuito impreso, donde se pudieran colocar todos los sensores y demás componentes, de manera que se tuviera un sistema más compacto y con una mejor presentación para su instalación en el dispositivo multirrotor.

En el diseño tuvieron peso dos factores principalmente, se debía confeccionar un PCB pequeño, de manera que no se convirtiera en un estorbo o en un peso innecesario

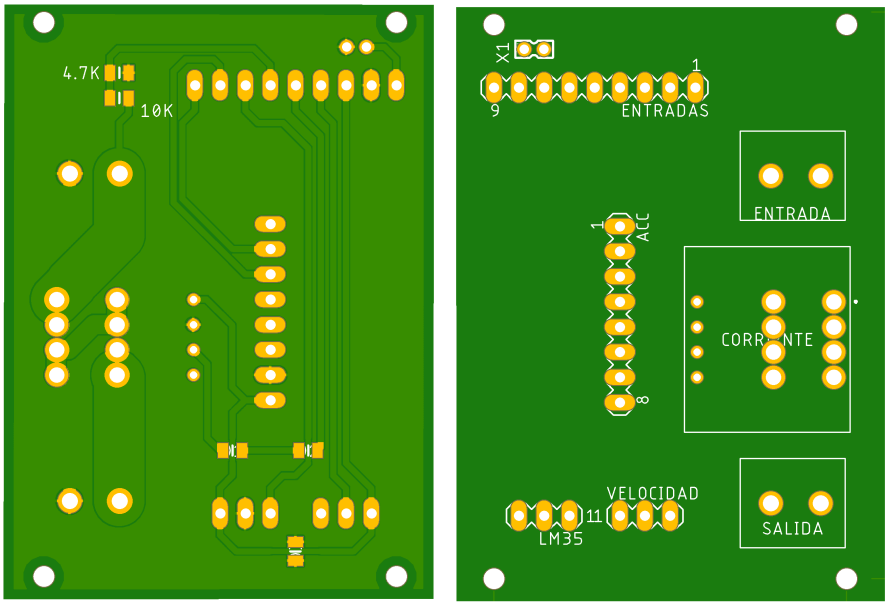


Figura 5.6: Diseño del PCB para montaje de los sensores de 43.4x60 mm.

en el vehículo y la cercanía que la mayoría de los sensores debían tener con el motor. Bajo esta premisa se intentó colocar todos los elementos en un solo PCB, sin embargo, se convertían un circuito muy grande, que debía ir sobre uno de los brazos del UAV, además en esta placa se contaba con elementos que no tenían ningún papel cerca del sistema de propulsión, como lo eran el microcontrolador, el módulo SD y el sensor de temperatura de la batería. Contemplando todo se optó por diseñar dos circuitos impresos como solución final, lo que permitiría generar placas pequeñas con solo lo estrictamente necesario en cada una.

Para el diseño del PCB se consideró la norma [37], para la selección del ancho de las pistas. Las señales generadas por los sensores y leídas por el Arduino no superan los 40mA máximos soportados por los pines analógicos de este microcontrolador. Para una corriente de este valor el ancho de una pista debería ser de al menos $1\mu m$, sin embargo, este tamaño es muy pequeño para la CNC con el que se confecciona la placa, por lo que se utilizó un ancho de 0.6mm, el cual manejaría sin problemas estas señales. En esta placa también existe una pista que si debe conducir altas corrientes, que es la que entra y sale del sensor de corriente, por acá pueden pasar hasta 15A, aunque el

promedio ronda los 10A. Con estas consideraciones y utilizando la norma se calculó que para una placa con un espesor de cobre de $70\mu m$, el ancho de la puesta debía ser de al menos 4.92mm según la norma, por lo que el diseño final se hizo con un ancho de 5mm, considerando que un mayor tamaño podría causar interferencia entre algunas pistas.

En la Figura 5.6 se observa el diseño del primer circuito impreso, este abarca la mayor parte los sensores y va instalado sobre el brazo del UAV como se puede observar en la Figura 5.7. Este PCB interrumpe las dos líneas de alimentación que salen de la batería hacia el ESC, esto permite medir la corriente y la tensión, cada uno con su respectivo método de medición. Sobre la placa esta instado directamente el acelerómetro que permite medir las vibraciones, este sensor es una de las razones de más importantes de instalar el PCB cerca del motor, ya que ahí las vibraciones o irregularidades se presentan con mayor intensidad. Finalmente se tienen tres pines para el sensor de temperatura y el de corriente. El sensor de temperatura va instalado directamente sobre el motor, por lo que lo que saldrán del PCB serán cables que se conectarán directamente a las pines del sensor. El sensor de velocidad tiene sus propias tres terminales que van directo al PCB, mientras las otras dos interrumpen las fases del motor. Al otro extremo del circuito se instalaron nueve pines que conectan, por medio de un fajo de cables, este PCB con el otro.

El segundo PCB se puede observar en la Figura 5.8, este diseño así como su montaje que se observa en la Figura 5.9, son más sencillos. Esta placa se instala sobre la carcasa que resguarda todo el sistema de control del UAV, por lo que solo hubo que rediseñar esta con unos taladrados y unos relieves en la parte superior para la instalación del PCB. El circuito contiene al Arduino UNO, el módulo SD y tres pines para el sensor de temperatura de la batería, que de igual forma va instalado directamente sobre la batería, por lo que a estos pines solo van tres cables que se conectan con el sensor. Además, tiene los mismos nueve pines que el otro PCB, permitiendo así la conexión entre ambos. En estos pines se incluyó uno que permite alimentar al Arduino con la

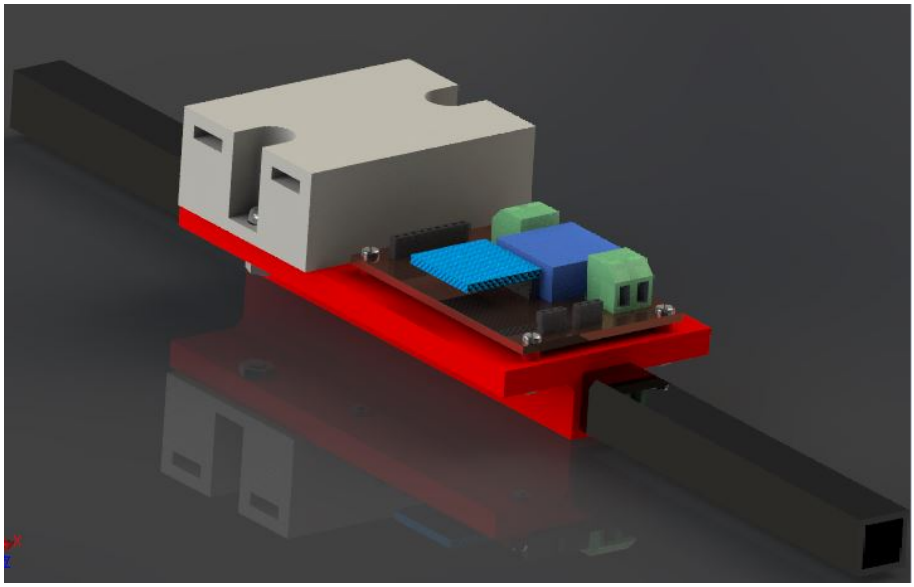


Figura 5.7: Diseño del montaje del PCB de los sensores sobre el brazo del UAV.

batería LiPo, sin embargo, se puede dejar sin conectar y alimentarlo por el puerto USB o el Jack de alimentación. En el montaje se decidió utilizar los taladrados que trae ya incorporados la placa de Arduino, esto por sus dimensiones estándar y porque da más seguridad al Arduino durante el vuelo, ya que es el componente de mayor valor en la placa. Además para la confección del segundo PCB se utilizó el mismo ancho de pista de 0.6mm, ya que en esta ninguna señal supera ese valor.

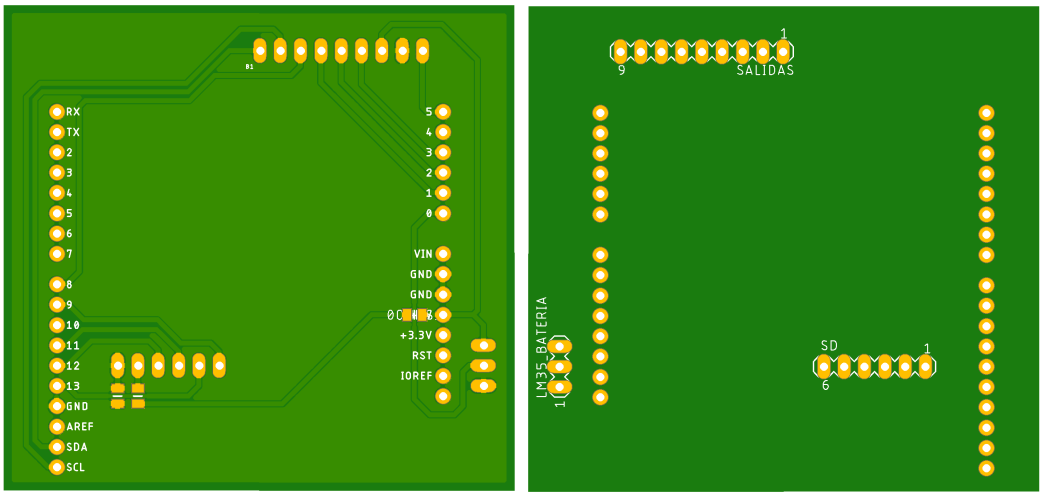


Figura 5.8: Diseño del PCB para montaje del Arduino y microSD 63.9x60.7 mm.

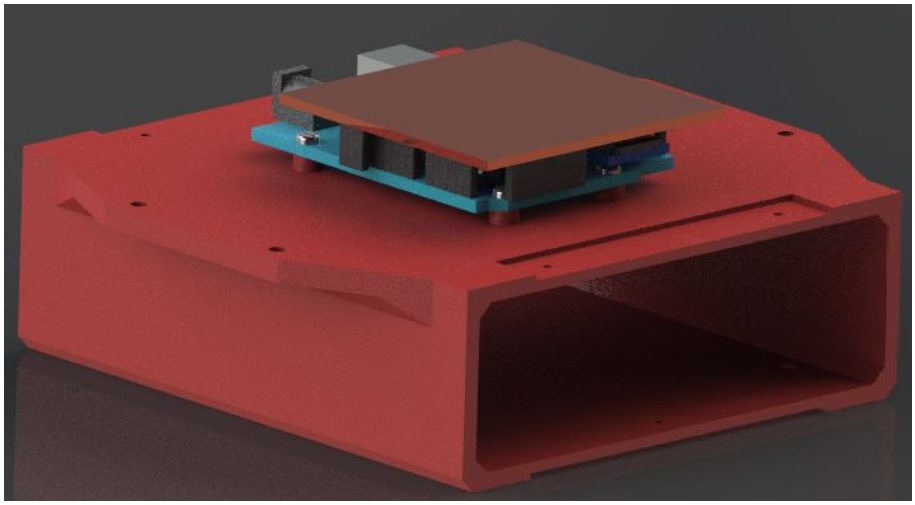


Figura 5.9: Diseño del montaje del PCB del Arduino y el módulo SD sobre la carcasa central.

En el Anexo 2 se pueden ver los circuitos que dieron origen a los PCB seleccionados y en el Anexo 3 se presentan los planos de las piezas de sujeción utilizadas en la instalación del PCB de los sensores, así como el de las modificaciones realizadas a la carcasa para la instalación de la segunda placa.

5.1.3.1 Manufactura de los elementos de sujeción

En cuanto al diseño del soporte se optó por buscar una opción rápida, de buena calidad y con materiales ligeros. Considerando que el GII contaba con una impresora 3D y que la carcasa actual del UAV fue manufactura utilizando esta técnica, se optó por desarrollar un diseño del nuevo elemento de sujeción de la misma manera. En cuanto al soporte para el PCB que va sobre la unidad central, solo represento una modificación pequeña al modelo anterior, por lo que de igual forma se planteó para su manufactura en impresión 3D.

Para la manufactura se consideraron dos materiales comúnmente utilizados en impresión 3D, ambos polímeros, el acrilonitrilo butadieno estireno (ABS) y el ácido poliláctico (PLA). En el trabajo de [38], se comparan las propiedades mecánicas de ambos materiales. Según su investigación la resistencia de ambos materiales puede variar de acuerdo con

densidad de relleno de la impresión o a la configuración que se le dé al material. Como conclusión de su trabajo se encontró la configuración panal con una densidad del 10% es la que ofrece peores propiedades mecánicas en el caso de ambos materiales, llegando a un esfuerzo de fluencia de 9.08MPa para el ABS y de 9.45MPa para el PLA. Aunque bajo condiciones no tan críticas por lo general el PLA tiene valores mucho más altos de resistencia a la deformación, como se puede ver en [39], donde el promedio para el esfuerzo de fluencia del ABS es de 28.5MPa y de 56.6MPa para el PLA.

A pesar de que el PLA ofrece un mejor esfuerzo a la fluencia, como se menciona en [40], utilizar este material no es adecuado para cualquier aplicación. El PLA destaca para aplicaciones donde se mantienen cargas estáticas, sin embargo, es considerado un material muy frágil, que tiene poca resistencia al impacto y las altas o bajas temperaturas. Cualidades en la que, si destaca el ABS, que a pesar de no ser un material muy flexible si soporta mejor los golpes debido a su mayor dureza a que su composición de polibutadieno, caucho sintético con alta resistencia al desgaste.

Considerando que los elementos diseñados se montarán sobre un vehículo volador, expuesto directamente a los rayos del sol, que traen consigo altas temperaturas, que además esta siempre expuesto a caer y recibir golpes, se considera que el material más adecuado para su confección es el ABS. Ya que además como también lo menciona en [40], este material ofrece la posibilidad de dar mejores acabados visuales también, debido a que permite ser tratado con acetona, sustancia en la que es soluble. En cuanto al costo , ambos materiales manejan un valor prácticamente igual, por lo que no se considera como un factor importante en este diseño.

Como se ve en los planos del Anexo 3, las piezas de sujeción llevan uniones mecánicas como las observadas en la Figura 5.10, de un tornillo con tuerca. En total se utilizaron 8 tornillos ISO 1580 M2.5x12, 8 tuercas ISO 4035 M2.5, 2 tornillos ISO 1580 M4x12 y 2 tuercas ISO 4035 M4, todos hechos de acero de grado 4.8 que según la norma ASTM F568M [41], tienen un esfuerzo a la fluencia mínimo de 340 MPa.

Figura 5.10: Uniones mecánicas utilizadas en los elementos de sujeción.

5.2 Algoritmo para la predicción y detección

Se diseñaron dos redes recurrentes para la detección y clasificación de fallos, la LSTM y la GRU, ambas muy similares, pero que pueden ofrecer distintos resultados que vale la pena comparar. La clasificación se realiza entre diez categorías, por lo que para cada uno de estas se generó un conjunto de datos, de manera que para el entrenamiento se cuenta con datos de cada una de las posibilidades de fallo.

5.2.1 Generación del conjunto de datos

Para la generación de datos sintéticos se partió de datos de vuelos pasados, los cuales habían sido recolectados con el controlador de vuelo Pixhawk 2. Este controlador tiene tres unidades de medición inercial, por lo que permite obtener tres conjuntos de datos para la aceleración, que será la medida de vibración utilizada. Además, proporciona datos de velocidad y tensión en la batería. Los datos obtenidos de este controlador se tomaron en un vuelo que no tuvo ningún problema, por lo que son considerados datos de un funcionamiento normal. Los datos generados para las demás categorías, se logran por medio de modificaciones al funcionamiento normal.

5.2.1.1 Caracterización de los datos en funcionamiento normal

Los datos en funcionamiento normal fueron tomados, en su mayoría, de un archivo de un vuelo pasado. De este se tomó la velocidad de tres motores, obteniendo un total de 2000 datos, de los cuales se utilizan 1500 para el entrenamiento y 500 se utilizarán posteriormente en la validación de la herramienta. En la Figura 5.11 se observa la forma que tiene los datos de velocidad, los cuales oscilan aproximadamente entre 3500 y 1500 rpm durante un vuelo normal.

Figura 5.11: Datos de velocidad en vuelo utilizados como referencia para la generación de datos sintéticos.

La Figura 5.12 muestra el comportamiento de la aceleración(vibración) en los ejes X y Z, se omite el eje Y ya que su comportamiento es muy similar al del eje X. Los tramos de la vibración son coincidentes con los de la velocidad, como se contaba con los datos de tres unidades de medición inercial, se tomaron tramos de datos de los tres, por lo que se cuenta con los mismos 2000 datos. En un vuelo normal la medida de vibración suele variar en rangos bajos, ya que la estructura absorbe gran cantidad de las vibraciones.

Para la generación de los datos de corrientes se escalaron los valores de velocidad, como se observa en la Figura 5.13, la forma de la gráfica de corriente y la de velocidad es la misma, lo que varían son los valores y las unidades. Por una consulta a [29], se decidió que la corriente máxima en vuelo normal rondara los 12A. Los datos de corriente

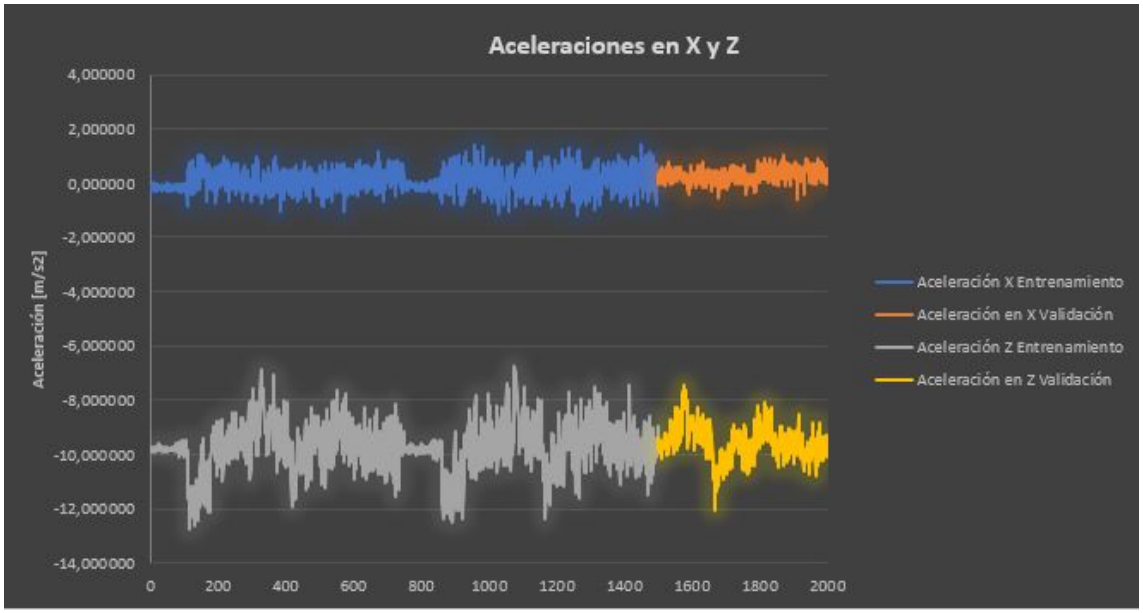


Figura 5.12: Datos de aceleración en vuelo alrededor de los ejes X y Z utilizados como referencia para la generación de datos sintéticos.

se pueden escalar desde los de velocidad en el caso de un vuelo sin ninguna anomalía, ya que no existen variaciones en la carga o en la tensión de la batería, cumpliendo con la ecuación 4.3, donde si se omiten todas las pérdidas, la potencia de entrada debe ser igual a la de salida, la corriente solo debe compensar los cambios en la velocidad, resultando en una curva con una tendencia similar. En la ecuación 4.3, **V** representa la tensión, **I** la corriente, τ el torque de motor y ω la velocidad angular.

$$V * I = \tau * \omega \tag{5.3}$$

La tensión de la batería ronda los 12V cuando no ha entrado en operación como se ve al inicio de los datos en la Figura 5.14, sin embargo una vez que comienza a volar, estabiliza su nivel de tensión alrededor de los 11.1V que es la esperada en una batería de LiPo de tres celdas. Estos datos también fueron tomados del archivo de un vuelo pasado.

Finalmente en las Figuras 5.15 y 5.16 se pueden observar los datos de temperatura en el motor y en la batería respectivamente. Por lo general la temperatura tiende a un

Figura 5.13: Datos de corriente en vuelo utilizados como referencia para la generación de datos sintéticos.

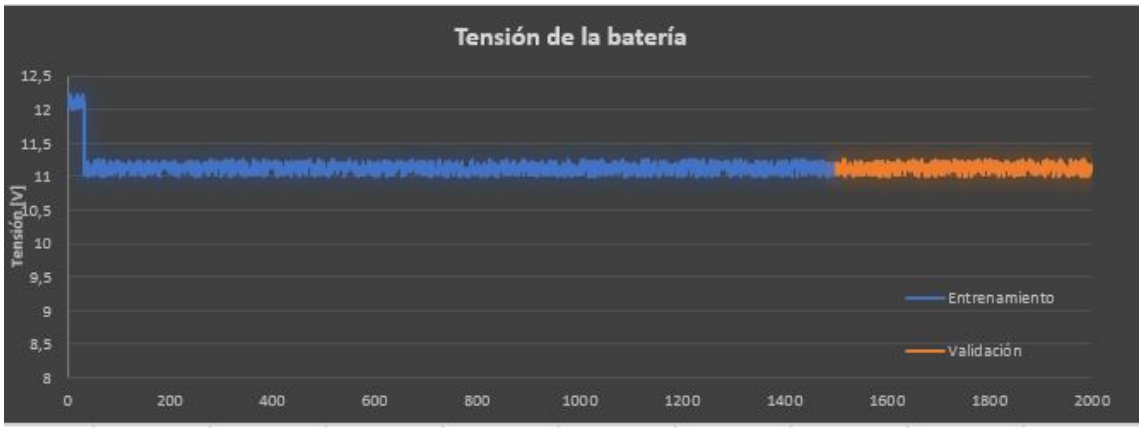


Figura 5.14: Datos de tensión en la batería en vuelo utilizados como referencia para la generación de datos sintéticos.

aumento en más componentes, aunque bajo un funcionamiento normal esta elevación no es muy pronunciada. El aumento de temperatura en el motor se puede deber tanto a aspectos mecánicos como eléctricos, aunque la mayoría de las veces la variable que más peso tiene en el aumento de la temperatura es la corriente. Considerando esto se generó los datos con una relación directa a la corriente, donde cada vez que hubiera un aumento en esta, se elevaría un poco la temperatura y si por el contrario disminuía, la corriente iba a bajar, aunque en una proporción menor que en la que sube. Se utilizo el mismo método para la temperatura de la batería, la cual también se ve afectada por la demanda de corriente.

Figura 5.15: Datos de velocidad en vuelo utilizados como referencia para la generación de datos sintéticos.

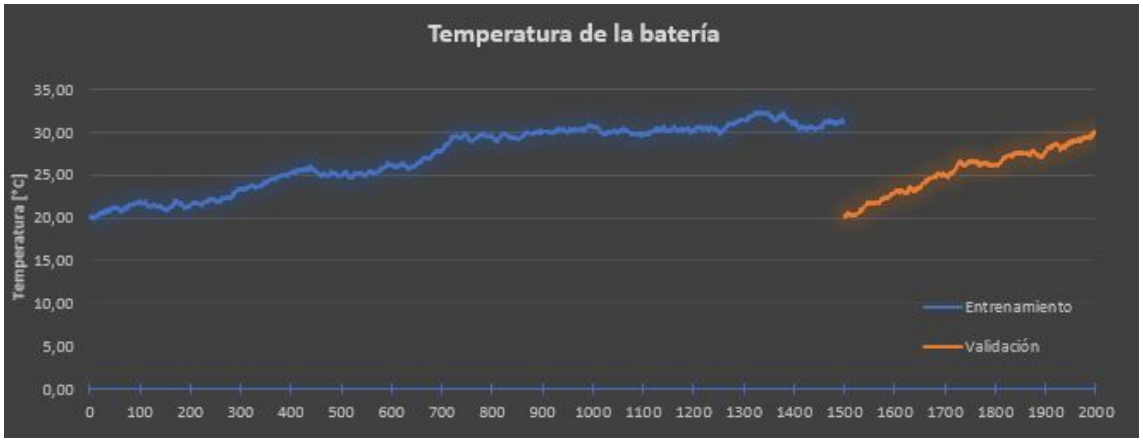


Figura 5.16: Datos de velocidad en vuelo utilizados como referencia para la generación de datos sintéticos.

5.2.1.2 Consideraciones sobre los datos para los diferentes tipos de error identificados

A continuación se describen las modificaciones o consideraciones con respecto a los datos en funcionamiento normal, realizadas para la generación de los datos correspondientes a cada uno de los fallos.

- **Motor o ESC dañado:** Cuando el motor o el ESC se dañan, se espera que este se detenga el movimiento abruptamente, por cualquiera que haya sido la razón del fallo, aunque por lo general la causa es un corto circuito que quema alguno de

los dos elementos. Por esta razón es que los datos de corriente y velocidad caen a valores cercanos a cero, no es correcto asumir que son cero ya que la medición se puede ver afectada por diversas fuentes de ruido, lo que provocaría que existan lecturas de valores diferentes de 0. En caso de hacerse dañado producto de un corto circuito, se espera que la temperatura, tanto de la batería como del motor, se hayan elevado, por lo que se espera que una vez dañado, este empiece a bajar lentamente. En cuanto a las vibraciones se espera que en el momento del fallo existan picos súbitos de vibraciones, ya que el vehículo de desestabilizará de golpe, sin embargo, rápidamente comenzará a estabilizarse, demandando más potencia a otros motores. En la Tabla 5.3 se puede observar un resumen de estas variaciones.

Tabla 5.3: Variaciones realizadas para generar los datos de un Motor o ESC dañado

Motor o ESC dañado	
Tensión	Misma que en funcionamiento normal
Corriente	Alrededor de 0A
Temperatura del motor	Desciende desde 60°C hasta llegar a temperatura ambiente
Temperatura de la batería	Desciende desde 40°C y luego sigo la misma tendencia que en funcionamiento normal
Vibraciones	Picos de aceleración de hasta 3 m/s^2 más que lo normal, que luego se estabilizan
Velocidad	Alrededor de 0 RPM

- **Batería dañada:** Cuando se presenta un fallo en la batería este se ve reflejado en una pequeña caída en la tensión total, aunque las dos baterías que van abordo están en paralelo, al perder una la demanda de corriente sigue siendo la misma y al no poder cumplir con esta demanda la batería baja su tensión. La corriente se eleva un poco para equilibrar la potencia eléctrica. Al igual que cuando daña el motor, se espera que la temperatura de la batería haya aumentado, ya que los daños internos o cortos circuitos que pueden dañar una batería, habrían subido la temperatura. En la Tabla 5.4 se observa un resumen de estos cambios.

- **Hélice rota:** En la Tabla 5.5 se pueden observar las variaciones aplicadas al

Tabla 5.4: Variaciones para generar datos cuando la batería se daña

Batería dañada	
Tensión	Alrededor de 9.5V
Corriente	Aumenta ligeramente
Temperatura del motor	Misma que funcionamiento normal
Temperatura de la batería	Desciende desde 40°C y hasta una temeperatura ambiente
Vibraciones	Misma que en funcionamiento normal
Velocidad	Misma que en funcionamiento normal

conjunto de datos para simular el falla de una rotura de hélice. Una rotura de hélice hace referencia a una perdida de un fragmento de la hélice, esto se puede ver como una disminución de la carga por lo que la corriente necesaria debe disminuir un poco, para efectos prácticos se disminuyó en un 10%, con respecto al funcionamiento normal. Al perder parte de la hélice también aumenta levemente la velocidad, debido a que al perder parte del área que cubre la hélice para realizar la propulsión, debe solventar esto con más velocidad angular. Finalmente hay un aumento en la magnitud de las vibraciones, que se mantiene, debido a la inestabilidad que genera la falta de uniformidad en la hélice.

Tabla 5.5: Variaciones para generar datos cuando la hélice se rompe un poco

Hélice Rota	
Tensión	Misma que en funcionamiento normal
Corriente	Disminuye en cierta proporción
Temperatura del motor	Misma que en funcionamiento normal
Temperatura de la batería	Misma que en funcionamiento normal
Vibraciones	Aumento de la magnitud en $3m/s^2$ más de la normal
Velocidad	Aumenta ligeramente su magnitud

- **Hélice desprendida:** Como se observa en la Tabla 5.6, el fallo de una hélice desprendida es una versión más extrema de cuando se rompe parte de la hélice, ya que existe de igual forma una disminución de en la carga, lo cual lleva a la corriente a valores muy bajos, en este caso se generaron datos por debajo de los 2A. Al no tener hélice el motor trata de compensar con velocidad, elevándola al máximo, sin embargo, al no existir hélice y tener un área de empuje nula, no se

da la propulsión. Finalmente, las vibraciones tienen el mismo comportamiento que cuando se daña el motor, ya que es como si este componente dejara de existir repentinamente, por lo que se dará un pico en las vibraciones y luego el sistema de control tenderá a estabilizar el UAV apoyándose en otros motores.

Tabla 5.6: Variaciones para generar datos cuando la hélice se desprende

Hélice Desprendida	
Tensión	Misma que en funcionamiento normal
Corriente	Disminuye a valores muy bajos
Temperatura del motor	Misma que en funcionamiento normal
Temperatura de la batería	Misma que en funcionamiento normal
Vibraciones	Picos de aceleración de hasta 3 m/s^2 más que lo normal, que luego se estabilizan
Velocidad	Se eleva abruptamente

- **Hélice en dirección opuesta y aflojamiento del soporte:** El fallo de instalar una hélice en dirección se puede presentar por un descuido a la hora de instalar la hélice, el problema es que este fallo provoca que la fuerza de empuje se uno de los motores vaya en dirección opuesta, generando la aparición de un poco de vibraciones y aumentado la demanda de otros motores para contrarrestar esta fuerza. En general el motor con la hélice al revés tiene un funcionamiento normal, por lo que la única variable en la que se refleja un poco este fallo es en las vibraciones, para efectos de la simulación se limito a un aumento en la magnitud de las vibraciones de 2 m/s^2. Este fallo se agrupo con el del aflojamiento del soporte por el parecido que pueden tener a la hora de presentarse, debido a que afectan únicamente con un aumento en las vibraciones. Por aflojamiento se entiende cuando se desajustan los tornillos que mantienen unido el motor al vehículo.

- **Desconexión de fases y Desmagnetización:** La desmagnetización y la desconexión de fases son fenómenos muy similares, ambos implican una reducción en la fuerza magnética entre las bobinas y los imanes permanentes. Al disminuir esta atracción el sistema trata de compensarlo elevando la corriente, de manera que aumente

Tabla 5.7: Variaciones para generar datos cuando la hélice se instala en dirección opuesta o se afloja el soporte del motor.

Hélice en Dirección Opuesta y Aflojamiento del soporte	
Tensión	Misma que en funcionamiento normal
Corriente	Misma que en funcionamiento normal
Temperatura del motor	Misma que en funcionamiento normal
Temperatura de la batería	Misma que en funcionamiento normal
Vibraciones	Aumento de $2m/s^2$ en las magnitudes de la señal normal
Velocidad	Misma que en funcionamiento normal

la potencia entregada al motor, para efectos de la simulación este aumento se estableció en un 25%. Además, se da la aparición de vibraciones, debido a las fuerzas de atracción no uniformes, presentes en el motor, lo cual induce inestabilidad en su giro. En la Tabla 5.8 se observa el resumen de estas variaciones.

Tabla 5.8: Variaciones para generar datos cuando se da una desmagnetización o una desconexión de fases.

Desconexión de fases y Desmagnetización	
Tensión	Misma que en funcionamiento normal
Corriente	Aumento de la en cierta proporción
Temperatura del motor	Mayor pendiente que funcionamiento normal
Temperatura de la batería	Misma que en funcionamiento normal
Vibraciones	Aumento en la magnitud de $2m/s^2$ más de lo normal
Velocidad	Misma que en funcionamiento normal

- **Rodamientos dañados:** Un rodamiento puede fallar por diversas razones, sin embargo, para efectos de la generación de datos se contemplará como una sola categoría. Por general el desgaste mecánico de los rodamientos es algo que se va presentando gradualmente con el tiempo y suele verse reflejado en vibraciones que tienen una frecuencia característica y va aumentado su amplitud con el tiempo. Considerando esto, para la generación de los datos de vibración, se optó por hacer que los datos de vibración fueran incrementando poco a poco llegando a sumar hasta $3m/s^2$, más que la medida de los datos en funcionamiento normal. Como se describe en la Tabla 5.9, también se elevo ligeramente el valor de corriente, en un 10%, debido a que un desgaste en el rodamiento agrega una carga extra al motor,

lo cual se convierte en una mayor demanda de corriente.

Tabla 5.9: Variaciones para generar datos cuando hay fallos en los rodamientos

Rodamientos dañados	
Tensión	Misma que en funcionamiento normal
Corriente	Se eleva levemente
Temperatura del motor	Mayor pendiente que funcioanmiento normal
Temperatura de la batería	Misma que en funcionamiento normal
Vibraciones	Su magnitud va creciendo lentamente
Velocidad	Misma que en funcionamiento normal

- **Motor bloqueado:** Como se observa en la Tabla 5.10, cuando se presenta el estado de motor bloqueado, se evidencia en casi todas las mediciones. El motor bloqueado aparece como una carga infinita lo cual dispara el valor de la corriente y se refleja como un incremento acelerado. Las temperaturas también se elevan debido a las altas corrientes, por lo que también se elevan rápidamente. La velocidad pasa a estar alrededor de 0 RPM y la tensión cae, debido a la alta demanda de corriente, aplicada a la batería. Finalmente, las vibraciones tienen el mismo comportamiento que cuando se daña el motor, se da un pico de vibraciones altas que luego se va normalizando poco a poco.

Tabla 5.10: Variaciones para generar datos cuando hay motor bloqueado

Motor bloqueado	
Tensión	Alrededor de 9.5V
Corriente	Se eleva rápidamente
Temperatura del motor	Se eleva rápidamente
Temperatura de la batería	Se eleva rápidamente
Vibraciones	Picos de aceleración de hasta 3 m/s^2 más que lo normal, que luego se estabilizan
Velocidad	Alrededor de 0 RPM

- **Ruptura del soporte:** El ultimo fallo hace referencia a cuando se presenta una ruptura en el soporte que mantiene unido a los motores al vehículo, en este caso las vibraciones aumentan abruptamente, ya que el motor no deja de

funcionar como en la mayoría de los casos, si no que puede quedar colgando de las líneas de alimentación, tendiendo a enrollarse alrededor del brazo, resultado en un aumento momentáneo de esta variable. Además al perder un motor, pero que sigue conectado, existirá una demanda de mayor velocidad, con una corriente asociada, debido a que el control tratará de estabilizar el UAV primero actuando sobre ese motor.

Tabla 5.11: Variaciones para generar datos cuando hay ruptura del soporte

Ruptura del soporte	
Tensión	Misma que en funcionamiento normal
Corriente	Aumento abrupto que luego se normaliza
Temperatura del motor	Misma que en comportamiento normal
Temperatura de la batería	Misma que en comportamiento normal
Vibraciones	Aumento de magnitud en 5 m/s^2 momentáneamente
Velocidad	Aumento abrupto que luego se normaliza

5.2.2 Diseño del subsistema para el procesamiento de datos

La herramienta computacional encargada de la parte de predicción, detección y clasificación de fallos fue diseñada como una red neuronal recurrente, por su similitud se optó por diseñar dos redes, una utilizando capas LSTM y la otra utilizando capas GRU, de manera que se pudieran comparar los resultados de ambas y definir cual es la más adecuada para utilizar en este problema específico. Ambas redes fueron implementadas en el lenguaje de programación Python, Ya que es un lenguaje realmente útil para el diseño rápido de sistemas, que además cuenta con múltiples bibliotecas que ayudan en este proceso, como lo son TensorFlow y Keras, utilizadas en este proyecto.

5.2.2.1 Modelo inicial

La arquitectura del modelo inicial propuesto se observa en la Figura 5.17, como se observa las entradas son las datos de los sensores detallados en la sección 5.1. Luego de las entradas se pasa a las capas recurrentes (GRU o LSTM), para el modelo inicial

se propone utilizar una capa con diez neuronas, como se indica en la Tabla 5.12. Finalmente se establecieron diez salidas para la red, ya que cada una representa una categoría de posible fallo, de los establecidos anteriormente.

Tabla 5.12: Configuración inicial de la red neuronal diseñada para la predicción, detección y clasificación de fallos

Parámetro	Valor o configuración
Optimizador	Descenso de gradiente estocástico (SGD)
Función de error	Entropía cruzada categórica
Función de activación de la salida	ReLU
Número de capas ocultas	1
Número de neuronas de las capas	10
Número de épocas	50
Tamaño de lote	12

En la Tabla 5.12 se pueden observar otros de los parámetros importantes escogidos en el diseño de la red neuronal. Dado que el planteamiento de puntos de partida en redes neuronales no siempre esta establecido o formalizado, se establecieron estos basados en la experiencia personal en el desarrollo de otros modelos similares, y se considero que podían ser una buena primera aproximación.

El primer parámetro es el optimizador, que en este caso se diseñó con el descenso de gradiente estocástico (SGD), estes un optimizador clásico, que permite la actualización de los pesos (Representados por cada uno de las flechas en la Figura 5.17), por medio de las derivadas parciales de la función de error. Para la función de error se utilizó la entropía cruzada categórica, la cual esta descrita por la ecuación 2.1 y es utilizada en problemas de clasificación, donde puede haber dos o más etiquetas en la salida.

Otro de los parámetros iniciales escogido para la red es la función de activación de la salida, en la capa recurrente esta función ya está preestablecida por la arquitectura, sin embargo se debía establecer una para la capa de salida, por lo que se optó por la función ReLU, la cual es una de las más utilizadas en la actualidad en el diseño de redes neuronales, una de sus principales ventajas es que mientras que los valores que llegan a la función no sean negativos, se evita el desvanecimiento de gradientes, debido a que

es una función lineal con elevada pendiente, lo cual hace muy difícil que los valores tiendan a cero.

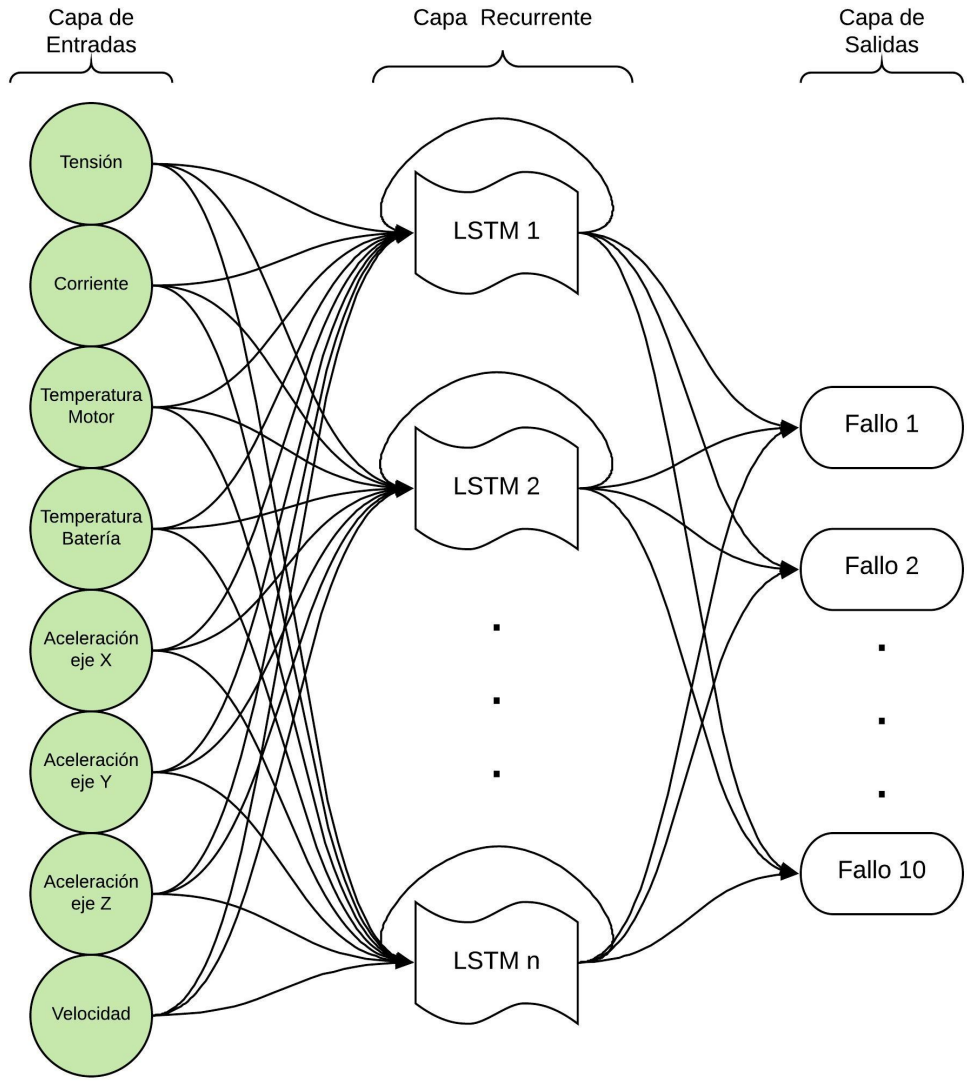
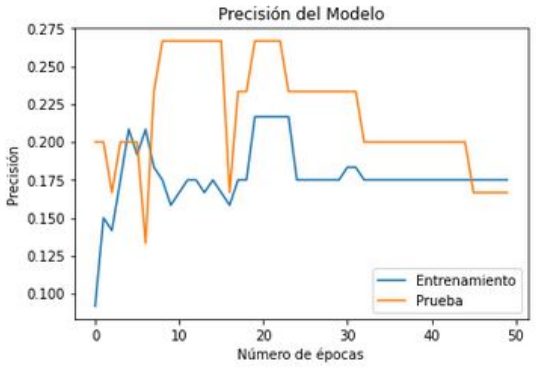


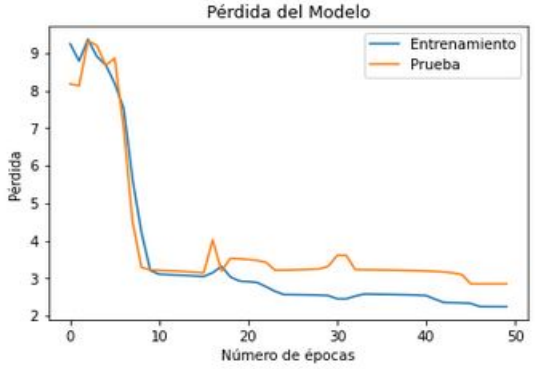
Figura 5.17: Estructura de la red neuronal LSTM utilizada.

Finalmente se tienen las variables numéricas de la red. La cantidad de capas ocultas y el número de neuronas se relacionan mucho entre sí, una mayor cantidad representa directamente una mayor complejidad en la arquitectura, lo que puede representar una mayor demanda computacional, que en ocasiones es innecesaria, por eso se parte de un

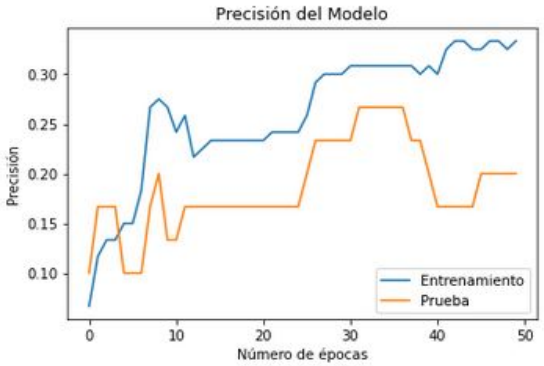
modelo con pocas unidades, con una sola capa y diez neuronas, para luego valorar si es importante aumentar o incluso disminuir estas cantidades. Los últimos dos valores también están relacionados entre sí, el número de épocas se refiere a la cantidad de veces que la red neuronal procesará el conjunto de datos completos, mientras que el tamaño del lote es una subdivisión que se crea en los datos, para que sean procesados en grupo, es decir en este caso se estableció un lote de doce datos, eso quiere decir que la red procesa doce datos y luego actualiza sus pesos, una vez hecho esto pasa a procesar otros doce y sigue realizando esto hasta haber procesado todo el conjunto de datos, concluyendo así una época. Como ya se mencionó estos valores fueron escogidos arbitrariamente, sin embargo, se analizará su impacto en la red neuronal y que pasa si se varían.



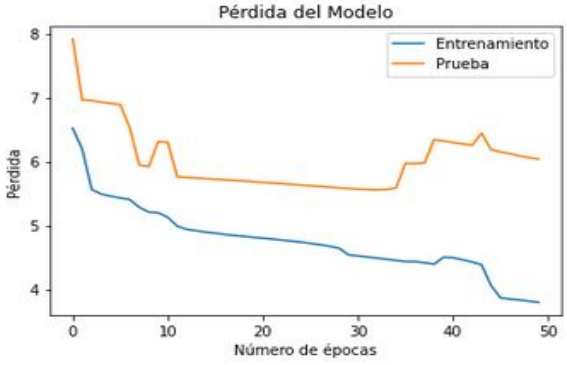
(a) Precisión LSTM



(b) Error LSTM.



(c) Precisión GRU



(d) Error GRU.

Figura 5.18: Gráficas de precisión y error para el primer modelo diseñado según la Tabla 5.12 utilizando capas LSTM y GRU.

Una vez diseñada la red se realizó un primer entrenamiento, para verificar el comportamiento de esta primera configuración, esto se llevó a cabo tanto para la red LSTM como para la GRU. En la Figura 5.18 se presentan dos tipos de gráfica, la de precisión, que permite saber el porcentaje de éxito que alcanza la red, como se observa en este caso para la LSTM la gráfica tanto en los datos de entrenamiento como los de prueba logro entre 15 y 18%, lo cual demuestra que esta primer configuración de LSTM no es capaz de detectar errores correctamente. Por otra parte, para la GRU si bien se obtienen valores más altos, igual se encuentran por debajo de un 35%, lo cual aun no se considera un resultado satisfactorio, ya que se busca que este porcentaje sea al menos de un 60%. Las gráficas de error, mostradas también en esta figura, funcionan de manera opuesta a la de precisión, se debe considerar que se espera que las salidas sean 0 o 1 y esta función de pérdida muestra la diferencia entre el valor esperado y el obtenido, en esta primer configuración de ambas redes se observan valores de 3 o 4, lo cual es una diferencia muy alta si se espera que los valores sean de 1 cuando mucho, ya que se estaría teniendo en la salida hasta un 300% más del valor esperado.

5.3 Mecanismo para la corrección de fallos

La corrección se compone por dos funciones principales, la corrección propiamente, que implica la interrupción de la alimentación para el motor en el que se este presentando el fallo y la comunicación al operador. En la Figura 5.19 se presenta un diagrama de flujo que explica como aplican estas dos funciones a los diferentes fallos detectados por la red. Los fallos se subdividieron en tres categorías de acuerdo con la relevancia de este, en rojo se observan los fallos críticos, son los que requieren una respuesta inmediata, la alimentación debe ser cortada al instante de haberse detectado, luego están los fallos graves en naranja, de igual forma se aplica la acción de cortar la alimentación, sin embargo, en primera instancia no representa un riesgo tan grande

como los otros. Finalmente, en amarillo se muestran los fallos medianamente graves, los cuales no requieren de la desconexión del motor, ya que su alteración al sistema no es tan relevante.

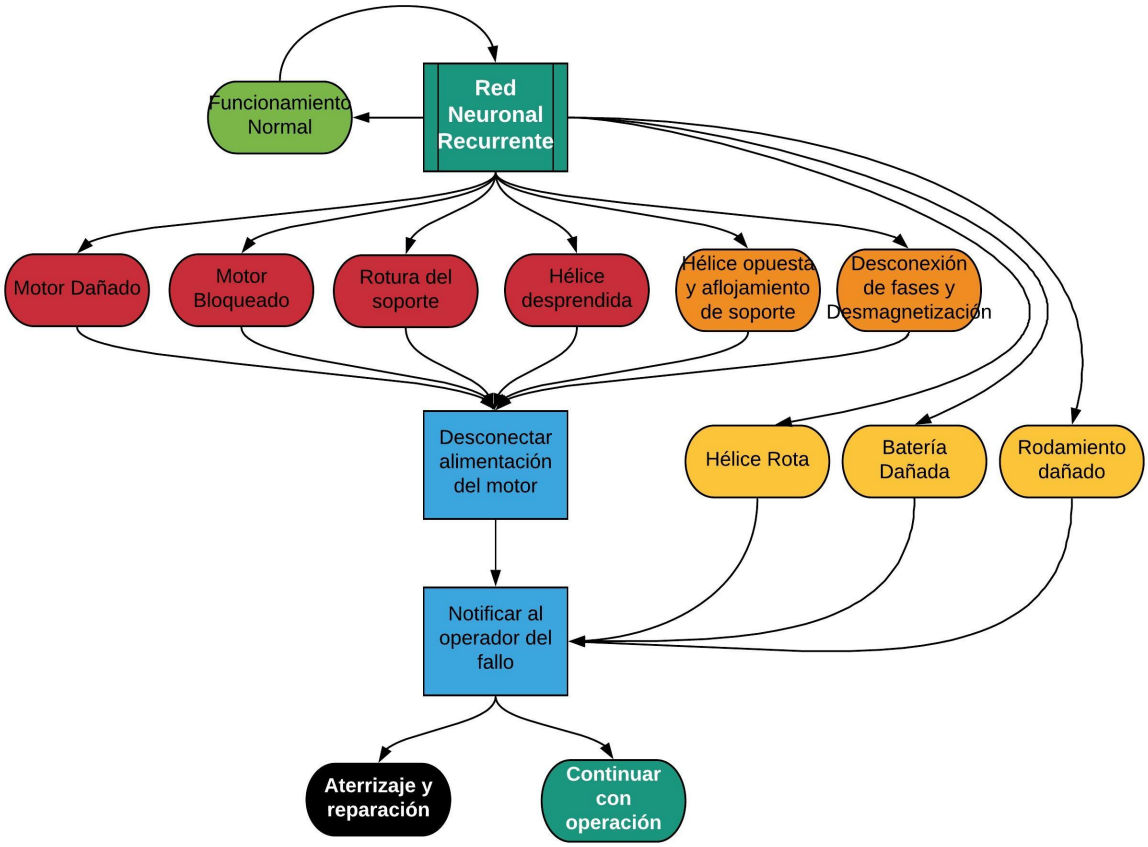


Figura 5.19: Diagrama de flujo que describe la respuesta a los fallos.

Una vez que el operador de vuelo es informado sobre la presencia del fallo, tiene dos opciones, puede iniciar con un aterrizaje inmediato o puede seguir volando el vehículo. La corrección de fallo no considera tomar acciones automáticas para el aterrizaje y lo deja a criterio del operador ya que en distintas ocasiones el aterrizaje puede no ser la mejor opción, dependiendo de la misión que se esté realizando o del área que se encuentre sobrevolando el vehículo.

5.3.1 Interrupción de corriente

En el concepto planteado anteriormente se definió que la interrupción de la alimentación del motor se haría de manera directa, esto quiere decir que se interrumpirán las líneas que van de la batería al controlador de velocidad. Para la implementación de este esta función se propone la utilización de un relé, el cual funciona como un interruptor, que al recibir una señal de control corta o abre el paso de la corriente para la línea que pase a través de este. Previamente en la Figura 5.7 se podía apreciar el montaje de este elemento justo detrás del PCB con los sensores, esto se realizó debido que el sensor de corriente también debe interrumpir las líneas de alimentación, por lo que se pretende aprovechar e instalar el relé justo antes de este sensor, de la manera que se ve en la Figura 5.20.

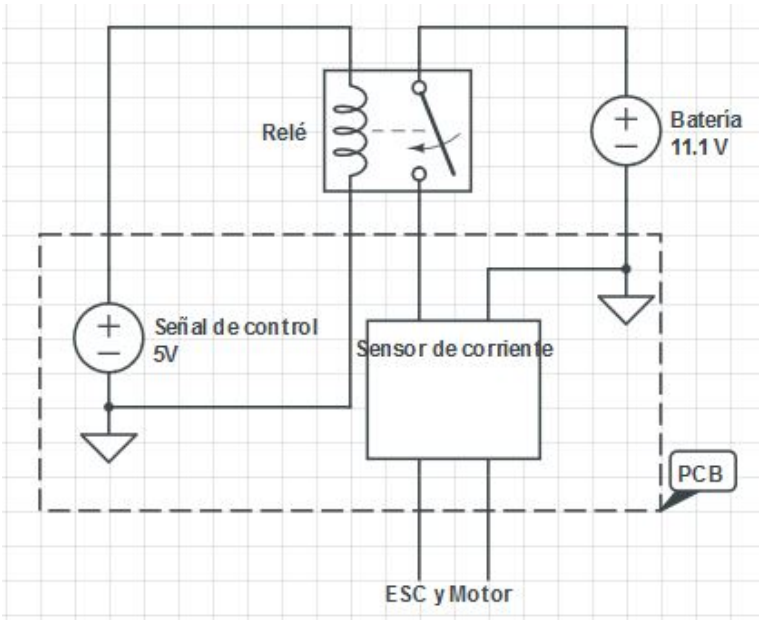


Figura 5.20: Conexión del PCB con el relé.

Para la selección del relé se consideraron las especificaciones del sistema que ya habían sido descritas, como la corriente máxima soportada, ya que este relé tiene que trabajar directamente con esta corriente. Por lo que se debía tratar de conseguir uno que fuera capaz de soportar al menos los 40 amperios, que es la corriente máxima

soportada por el ESC, además del relé debía ser apto para manejar corrientes directas tanto en la entrada de control como en la de potencia. Tomando en cuenta todas estas consideraciones se optó por el relé mostrado en la Figura 5.21.

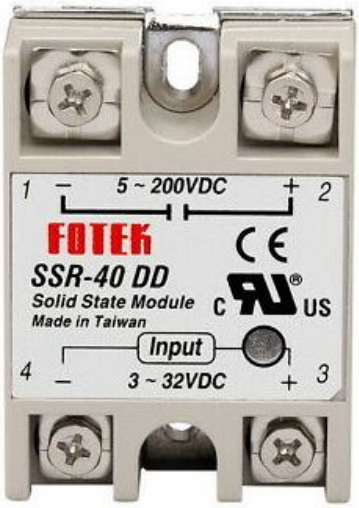


Figura 5.21: Relé SRR-4- DD utilizado para la interrupción de la corriente [42].

Este relé esta diseñado para soportar corrientes nominales de hasta 40A y picos máximos de hasta 400A, su señal de control puede ir de 5 a 200 V, con una corriente de activación de 75mA. Pude interrumpir líneas de hasta 32V de tensión, lo cual resulta perfecto en este caso que apenas se manejan 11.1V. Opera sin problemas en temperaturas entre los -20 y los 80°C, por lo que operará perfectamente con la temperatura ambiente. Finalmente, la última característica de interés es su rápida respuesta, la cual es menor a 10 milisegundos. Por lo que una vez enviada la señal de control la respuesta es casi inmediata. Toda esta información se encuentra en su hoja de datos[42].

5.3.2 Integración del diseño y comunicación con el operador

En la Figura 5.22 se muestra un diagrama de los principales elementos que interactúan con el sistema implementado. Este proyecto incorporo los PCB y el relé, los demás eran elementos ya presentes desde antes en el vehículo, pero que se pretenden utilizar a la hora de integrar todo el proyecto.

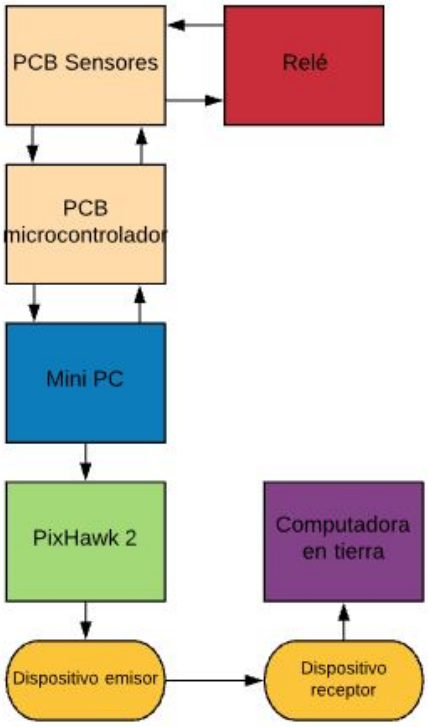


Figura 5.22: Relación de los elementos que conforman parte del sistema de detección y corrección de fallos.

El flujo de información es bidireccional como se puede observar según las flechas, en este caso el inicio del flujo de datos comienza en el PCB de los sensores, esta placa se comunica directamente por una faja de cables al PCB del microcontrolador, por lo que se puede decir que trabajan como uno solo. Los PCB son los encargados de la recolección y procesamiento de los datos leídos por los sensores, luego se conectan mediante un cable USB al Mini PC donde se estará ejecutando el algoritmo de detección de fallos, por lo que el programa establecerá una comunicación serial con el microcontrolador y solicitará los datos directamente, de manera que una vez que cuente con una secuencia del tamaño establecido para la entrada de la red, esta ejecutará el algoritmo.

Una vez que el algoritmo detecta un fallo, primero se procede a enviar una señal al PCB de manera que el microcontrolador cambie la señal de control del relé y que por consiguiente se interrumpa la alimentación del motor de ser necesaria esta acción. Por otra parte, la Mini PC, se comunica vía serial con el PixHawk, el cual finalmente,

mediante el protocolo MAVLink, es capaz de enviar mensajes por radio frecuencia a la computadora en tierra, en la cual se pueden observar parámetros del vehículo por medio de aplicaciones como Mission Planner, además de mensajes enviados por PixHawk.

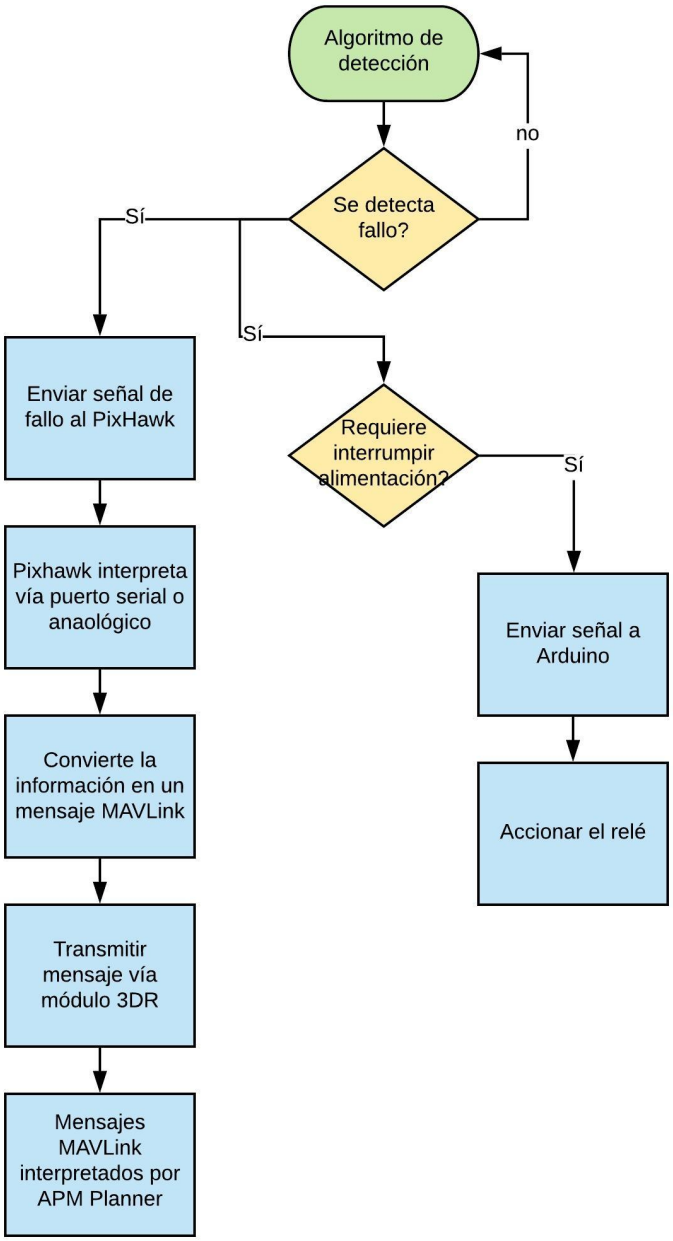


Figura 5.23: Diagrama de flujo sobre la comunicación de fallos al operador.

En la Figura 5.23, se detalla un poco más el procedimiento para la comunicación del fallo. Antes de explicar un poco más este procedimiento se detallan a continuación

los componentes involucrados:

- **Mini PC Intel® NUC7i5BNK:** Es una mini computadora que va a bordo del UAV, se encarga de procesar algunos de los cálculos necesarios en el vuelo del vehículo, sobre todo aquellos relacionados con las funciones de comportamiento autónomo. Cuenta con un sistema operativo Windows y permite una memoria de hasta 32GB [43]. Sobre esta mini PC es que se correrá la red neuronal.

- **PixHawk 2:** Es un sistema de autopiloto de software abierto, el cual consta de 2 placas principales, la IMU y la FMU(Unidad de manejo de vuelo), ambas trabajan en conjunto para controlar el vuelo de la aeronave. El software de PixHawk esta respaldado por Ardupilot y utiliza un protocolo de comunicación MAVLink reconocido por este software[44]. En la Figura 5.24 se observa el PixHawk que utiliza el octocóptero del GII.

Figura 5.24: PixHawk 2.1 también conocido como Cube flight controller [44].

- **Módulo 3DR:** Este es el transmisor que comunica la computadora en tierra en el PixHawk sobre el vehículo, es un modulo que trabaja 433MHZ y con una potencia de transmisión de 100mW [45]. La comunicación se establece con el protocolo de MAVLink ya mencionado. Además, según lo descrito en el trabajo de [46] la distancia máxima conseguida con esta comunicación ronda los 200m, el cual era el mínimo establecido en las especificaciones.

El protocolo MAVLink utilizado en esta comunicación, es comúnmente utilizado para la transmisión de datos entre vehículos aéreos y estaciones en tierra. Tiene un

largo conjunto de mensajes predefinidos en un documento se extensión XML, y sus menajes pueden ser enviados por casi cualquier tipo de conexión serial, sin depender del tipo de tecnología. Los mensajes no pueden superar un tamaño de 263 bytes y la estructura clásica de los mismos se observa en la Figura 5.25. A continuación, se da una breve descripción de cada uno de estos bytes:

- Inicio(STX): Este byte indica el comienzo del mensaje.
- Longitud (LEN): Indica el tamaño del mensaje (PAYLOAD), y puede tener un valor entre 0 y 255.
- Secuencia(SEQ): Indica la posición en la secuencia de mensajes enviados, esto permite reordenarlos o detectar paquetes perdidos, de igual forma puede tomar un valor entre 0 y 255.
- Sistema(SYS): Permite diferenciar el sistema del que proviene le mensaje, en caso de existir más de uno en la red. Puede tener un valor entre 1 y 255.
- Componente(COMP): Permite identificar el componente del que proviene el mensaje, como por ejemplo la IMU o la FMU. Puede tomar valores entre 0 y 255.
- Identificación del mensaje(MSG): Permite reconocer el tipo de mensaje que se recibió, para facilitar el proceso de decodificación.
- Mensaje(PAYLOAD): Este el el contenido del mensaje, puede tener un tamaño entre 0 y 255 bytes.
- Suma de Comprobación(CKA-CKB): Este último se compone de dos bytes y permite la comprobación de errores en el mensaje.

Los mensajes son recibidos y decodificados por el programa de Mission Planner de ArduPilot, en una interfaz como la que se observa en la Figura 5.26, donde en la esquina inferior izquierda se muestra un panel con las variables que se desean observar

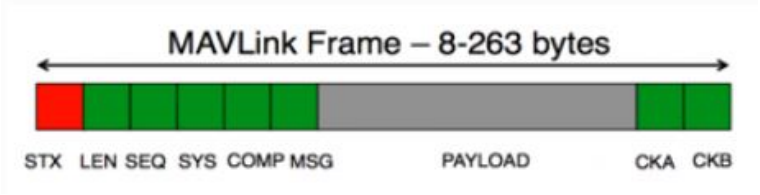


Figura 5.25: Bytes del protocolo MAVLink [47].

del vehículo, en esta misma sección existen diferentes pestañas, una para observar el estado de cada una de las variables y otra para los mensajes recibidos, ambas permitirían observar loes mensajes de fallo reportados por el PixHawk.

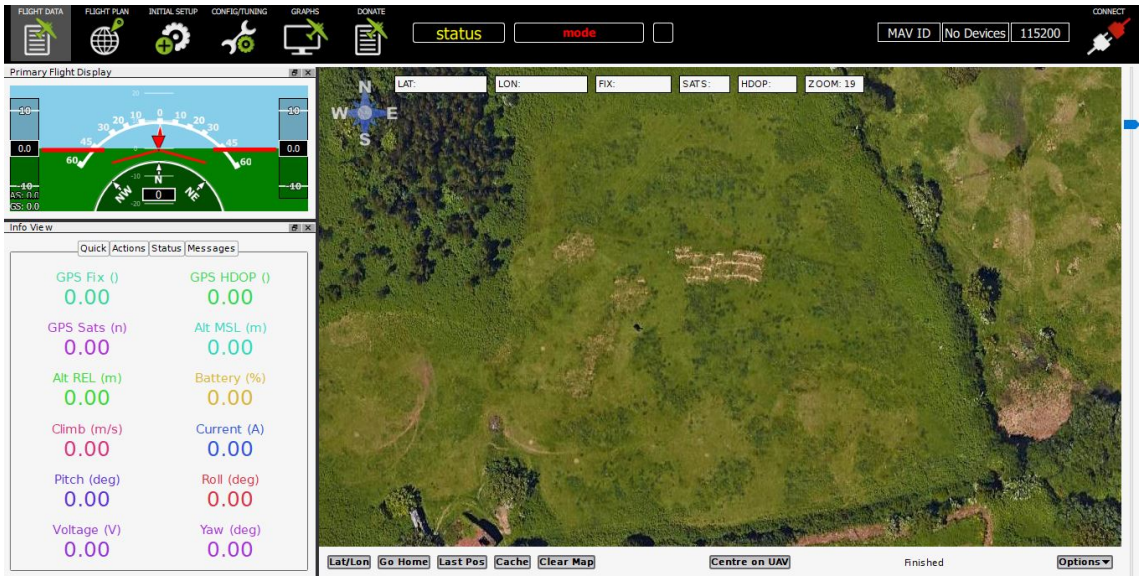


Figura 5.26: Pantalla principal del software Mission Planner de Ardupilot [48].

Capítulo 6: Análisis de Resultados

En este capítulo se presentan los principales resultados del diseño implementado como solución del presente proyecto. En el capítulo anterior se detallaba como estaba compuesta la solución implementada en cada uno de sus subsistemas, por lo que para este apartado los dos primeros subsistemas fueron probados, de manera que se muestran resultados tanto del sistema de recolección de datos como del algoritmo de detección y predicción implementados, además se muestra el proceso de como cada uno de estos subsistemas fue depurado.

6.1 Depuración del sistema de recolección de datos

Aunque el sistema de recolección de datos no fue probado en vuelos reales, su correcto funcionamiento fue verificado con ayuda de un motor y un controlador de velocidad sujetado, de manera que fue posible probar el funcionamiento de cada uno de los sensores y su funcionamiento en conjunto. El microcontrolador utilizado en la etapa de pruebas fue un Arduino UNO como se mencionaba anteriormente el programa utilizado se encuentra en el Anexo 1. Por lo que el desempeño obtenido en cada uno de los sensores se logró con trabajando con esta placa.

Se realizaron varias pruebas de medición de tensión, las cuales el Arduino fue capaz de leer con ayuda del convertidor analógico digital de 10 bits integrado en la placa, en la Figura 6.1 se muestra una gráfica generada con los datos tomados en una de las pruebas, como se observa es una muestra de aproximadamente 8000 datos los cuales

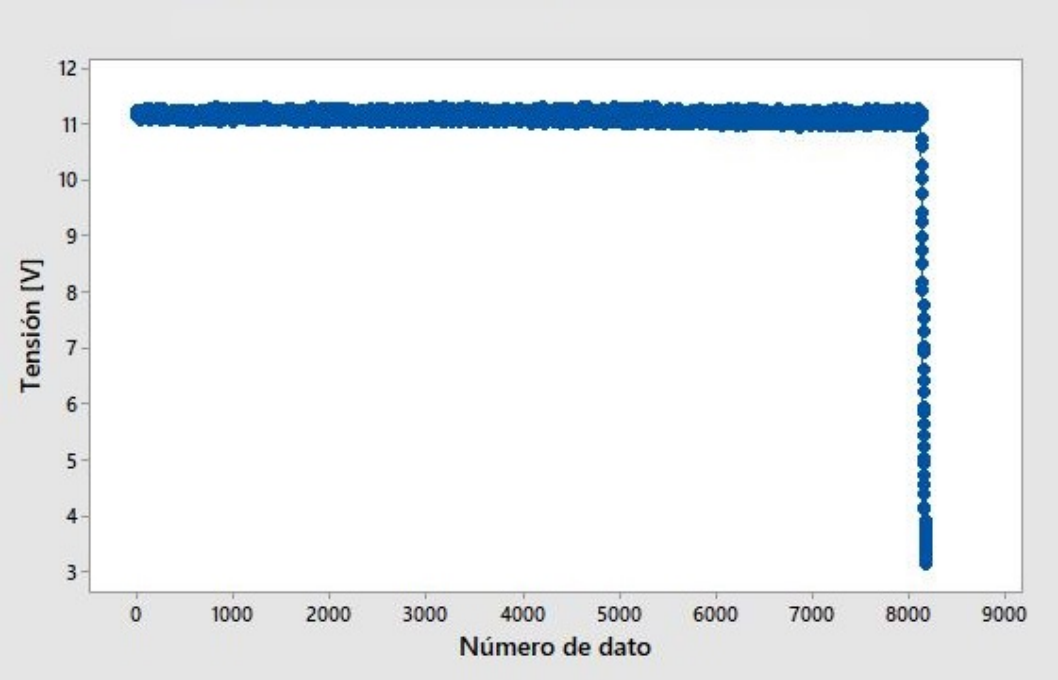


Figura 6.1: Gráfica de tensión medida en la alimentación del motor con ayuda del ADC de 10 bits de la placa Arduino UNO.

fueron tomados a una razón de 30Hz. La alimentación era fija a 11.1V y como se puede observar la lectura se mantiene muy cercana a este valor, excepto por el tramo final donde se cortó la alimentación, llevando la medición a 0V súbitamente. Dado que la lectura de no dio ningún problema con este tipo de medición, se optó por conservar esta forma de tomar los datos de tensión.

Las pruebas de temperaturas se limitaron a probar que los sensores funcionaran y dieran valores acordes a la realidad, por lo general su estudio se hicieron respecto a la temperatura ambiente, lo que permitió pruebas de entre 17 y 23 grados. En este caso se presenta en la Figura 6.2 los datos de temperatura tomados a 23 grados centígrados, como se observa los datos varían entre 22 y 24 grados aproximadamente. La lectura no se aleja de los esperado de este sensor el cual es una precisión de ± 0.5 grados centígrados. Para el desarrollo de este proyecto la diferencia de un grado en la lectura de la temperatura no es considerado crítico, ya que por lo general lo que importa es observar la tendencia creciente o decreciente de la misma, pero en valores

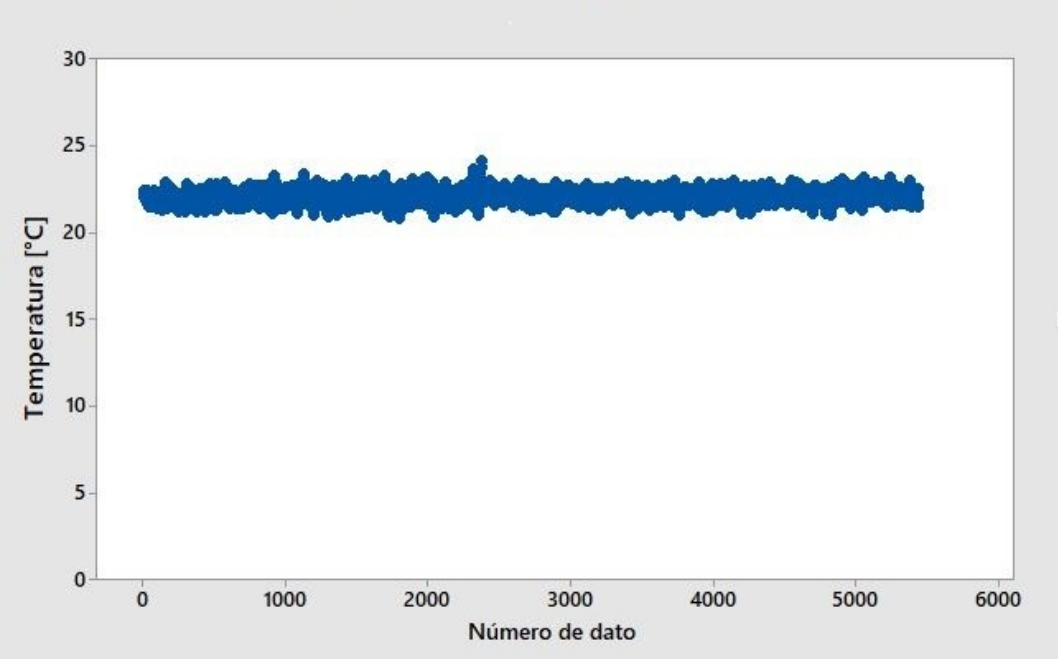
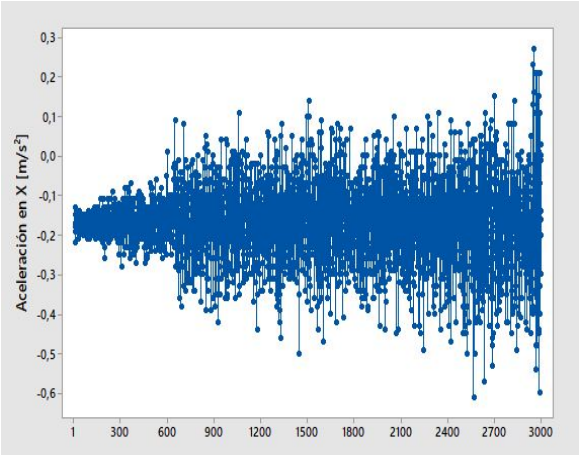


Figura 6.2: Gráfica de temperatura medida con ayuda del sensor LM35

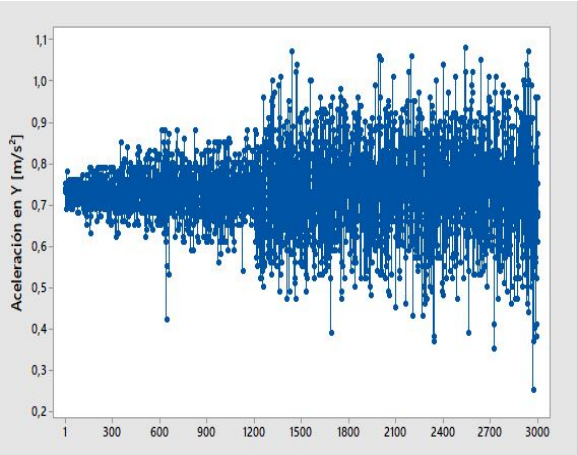
más significativos.

El último sensor que fue probado con éxito y sin necesidad de modificaciones fue el sensor de vibraciones, en este caso utilizando el acelerómetro MMA84, que permite medir las aceleraciones alrededor de cada uno de los ejes. Este sensor valor de referencia de sus variaciones dependiendo de la posición en la que se coloque el sensor, pero en su posición horizontal de montaje se espera que las mediciones del eje X y Y tengan como referencia el cero y que las aceleraciones en Z estén alrededor de -9.8 m/s^2, valor de la aceleración producto de la gravedad, que ya se presenta como constante en ese eje. En la Figura 6.3 se pueden observar los gráficos de vibración, para cada uno de los ejes, estos datos fueron tomados mientras se subía la velocidad del motor y se colocaba el sensor cerca de este, como se puede observar las vibraciones tienen una tendencia ascendente en su magnitud, en las tres gráficas, lo cual cumple con lo esperado, ya que al aumentar la velocidad los motores aumenta los niveles de vibración. Otro detalle importante de estas gráficas es que se puede observar que las aceleraciones no presentan variaciones de valores muy altos, se mantienen en rangos de hasta $\pm 2 m/s^2$ e incluso

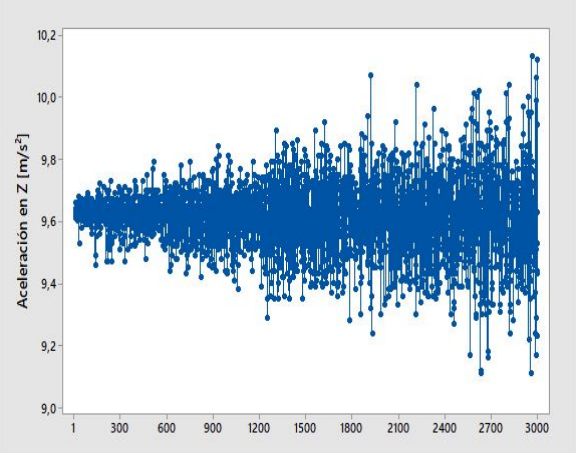
menor a eso. Esto permite confirmar que la mejor configuración para este sensor es la de la escala de $\pm 19.6 m/s^2$, que es la más pequeña disponible en este sensor.



(a) Aceleraciones en el eje X



(b) Aceleraciones en el eje Y



(c) Aceleraciones en el eje Z

Figura 6.3: Gráficas de aceleración, utilizadas como medida de vibración, tomadas con el sensor el MMA84.

Una de las razones de mayor relevancia a la hora de seleccionar el sensor de corriente fue su capacidad para soportar altas corrientes, ya que, aunque no se espera que en el sistema se lleguen a presentar más de 15A, el cual es el límite permitido por el motor, en presencia de un corto circuito esta corriente podría alcanzar valores mucho más altos. El HXS 50-NP soporta una corriente nominal de hasta 50A y picos de 150A, sin embargo esto es solo alcanzado en una de las conexiones disponibles en el sensor, con ose muestra en la Figura 6.4, este sensor se puede conectar de tres maneras distintas

permitiendo las cantidades de corriente que soporta y variando la escala de las lecturas, lo que varía la precisión de las mismas.

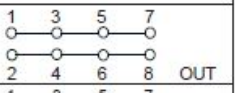
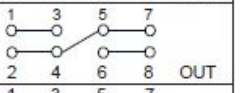
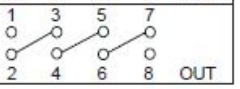
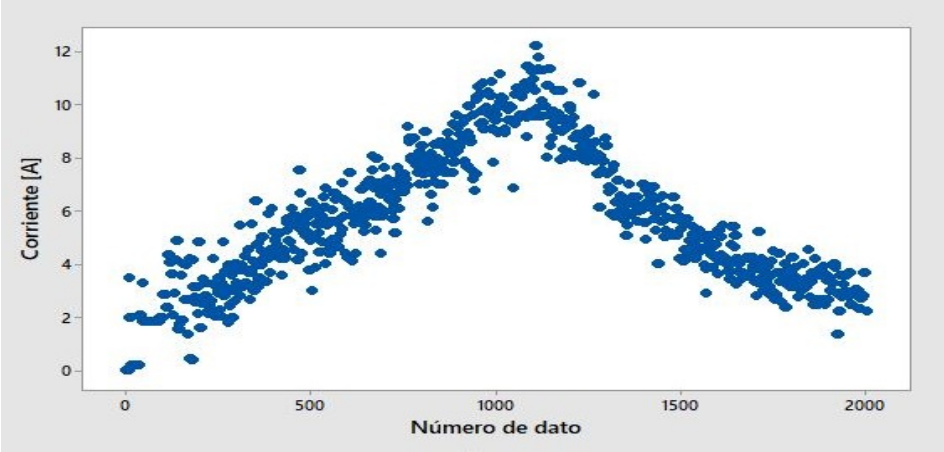
Number of primary turns	Primary current		Primary resistance R_p [mΩ]	Primary insertion inductance L_p [µH]	Recommended PCB connections
	Nominal I_{PN} [A]	Maximum I_p [A]			
1	50	150	0.05	0.025	
2	25	75	0.2	0.1	
4	12.5	37.5	1	0.4	

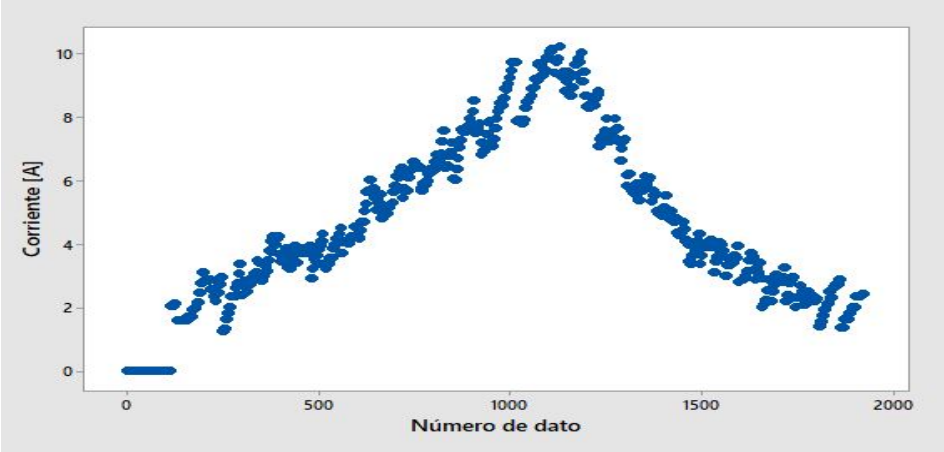
Figura 6.4: Configuraciones disponibles en el sensor HXS 50-NP [32].

Las pruebas del sensor de corriente se realizaron explorando todo el rango posible de corrientes con el motor, de manera que se elevó lentamente la velocidad hasta el máximo y luego se disminuye, como se observa la tendencia en las Figuras 6.5.a y 6.5.b, esta corriente fue tomada en pruebas de aproximadamente un minuto, por lo que se obtienen muestras de unos 2000 datos como se observa en las gráficas. Entrando en detalle con los resultados obtenidos, en la Figura 6.5.a, se puede observar el funcionamiento del sensor de corriente bajo la primer configuración y sin ningún tipo de ajuste, los datos recolectados reflejan la tendencia de la corriente, sin embargo, se obtiene una señal muy ruidosa, donde la corriente puede oscilar en rangos de hasta 4A como se puede observar.

Como manera de disminuir el ruido en la señal se optó por hacer dos correcciones a la manera de obtener los datos, la primera a nivel del hardware donde se cambió la configuración del sensor, se conectó de manera que la corriente nominal sea de 25 amperios y soporte picos de hasta 75 amperios, esto debería duplicar la precisión de la lectura ya que la escala se reduce la mitad. A nivel de software se optó por promediar la señal, de manera que antes de almacenar un valor se realiza la lectura de 5 valores y se guarda el promedio. El resultado de estas modificaciones se observa en la Figura 6.5.b, donde se puede notar una gran mejora con respecto a los datos recolectado



(a) Sin modificaciones



(b) Aplicando modificaciones

Figura 6.5: Gráficas de corriente medidas con el sensor HXS 50-NP.

anteriormente, si bien aun existe presencia de ruido, se puede ver como los rangos de oscilación entre valores es de aproximadamente 2A e incluso menores. Algo a destacar de ambas gráficas es que existe una mayor presencia de ruido en las lecturas de corriente pequeñas, lo cual era de esperar ya que el sensor está diseñado para medir valores muy altos de corriente, sin embargo, tener alta precisión en los valores bajos no es tan relevante, ya que por lo general los fallos o cortos circuitos se evidencia por corrientes elevadas.

El sensor de velocidad fue el que más complicaciones presentó en las pruebas, debido a que en un tramo específico la señal se distorsionaba de la manera que se observa en

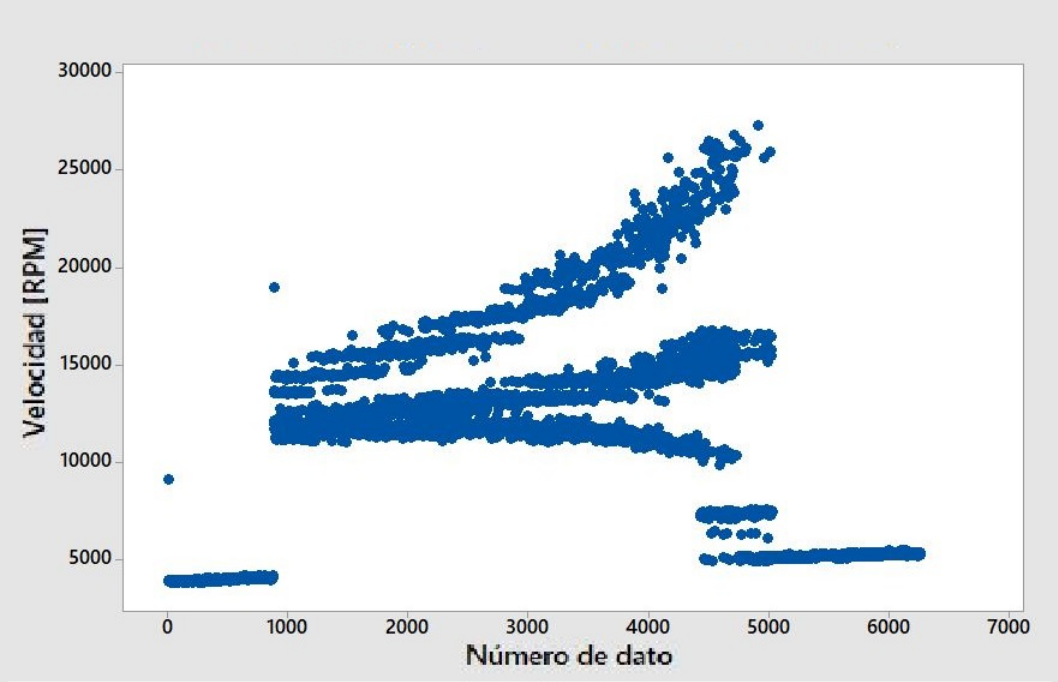
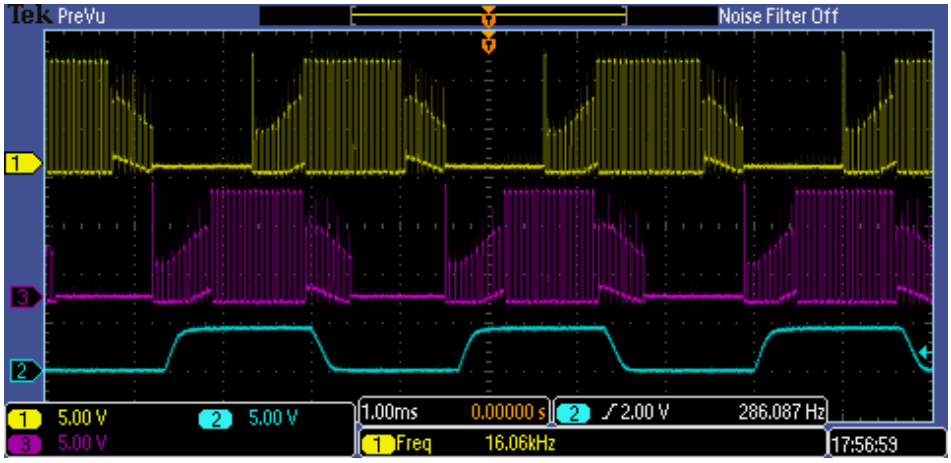


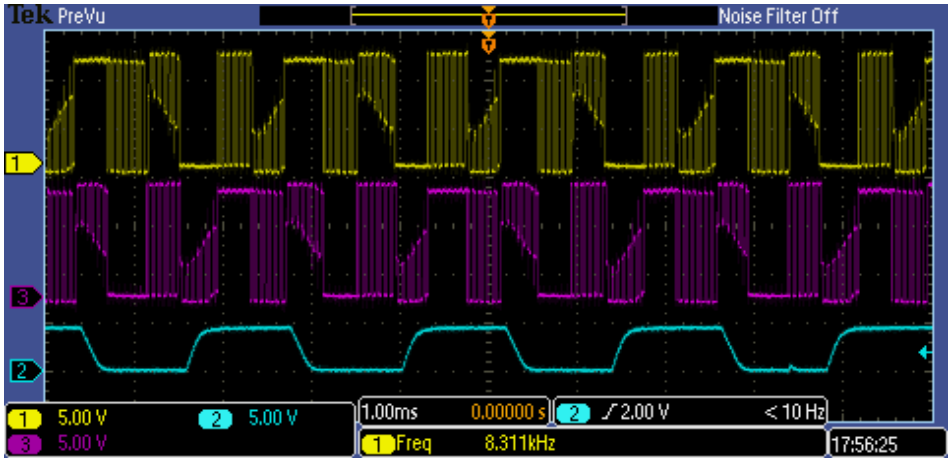
Figura 6.6: Error presente en los datos de velocidad recolectados con el sensor EagleTree Brushless RPM sensor.

la Figura 6.6. Este fallo se presenta específicamente cuando el ciclo de trabajo del ESC está entre el 29% y 40%. En esta región se puede observar como las lecturas de velocidad se disparan a valores muchísimo mas altos que los que estaban justo antes de que apareciera el fallo, además no tienen una tendencia definida. Dado que la medición de velocidad se considera una de las más importantes, en conjunto con la de corriente y vibraciones, este no resultaba un error que se pudiera ignorar, por lo que se procedió a realizar diferentes pruebas para poder encontrar el origen del fallo.

Primero se consideró que el sensor pudiera estar dañado, por lo que se probó con otro que se contaba en el laboratorio, sin embargo, el fallo se seguía presentado de la misma manera. Se probó con otros ESC, del mismo tipo y uno de otra marca, pero se obtuvo el mismo resultado. Las últimas pruebas se realizaron probando otros motores, primero probó con uno nuevo de exactamente el mismo tipo, lo cual de nuevo no ofreció mejora y finalmente a manera de descarte se utilizó un motor de otra marca y de mucho más grande, el cual no puede ser utilizado en el UAV sobre el que se realizan las pruebas,



(a) Zona donde el sensor opera sin problemas



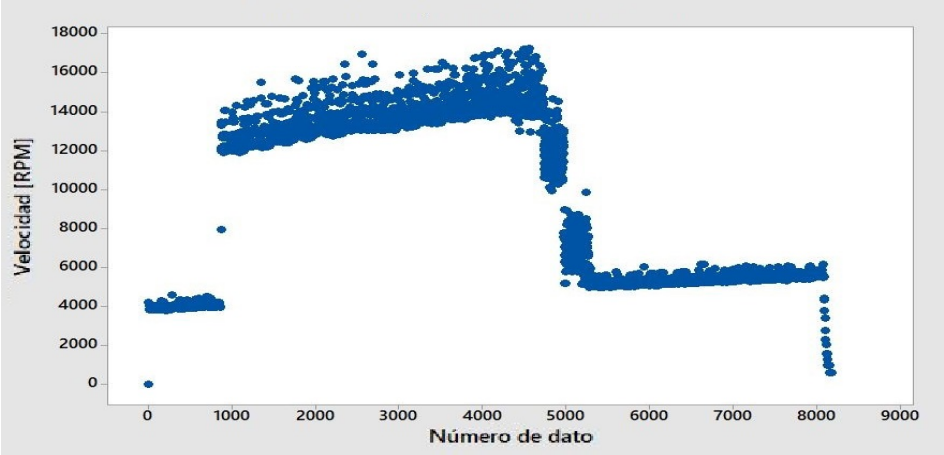
(b) Zona donde se distorsiona la señal

Figura 6.7: Lecturas de 2 de las fases del motor y de la señal de PWM generada a la salida del sensor de velocidad.

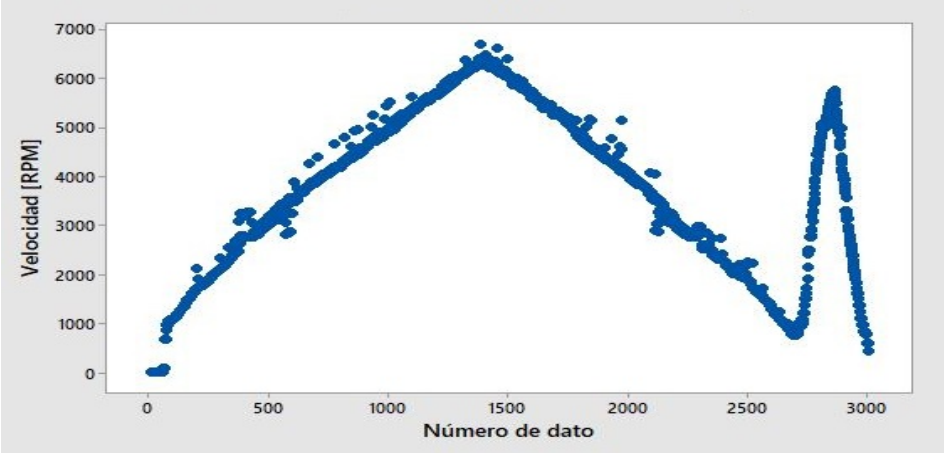
sin embargo, en este caso se obtuvieron, por primera vez, datos sin presencia de esta perturbación en la señal.

Una vez se detectó que el problema parecía tener una relación directa con el motor, se procedió a analizar en detalle la señal que lee el sensor de velocidad, esto se llevó a con un osciloscopio, con el cual se leyeron las dos señales de las fases del motor que lee el sensor de velocidad, adicional a esto se midió la señal de PWM generada a la salida del sensor de velocidad. En la Figura 6.7 se pueden observar estas señales. Como se indica en la figura, la primera imagen es de la visualización de la señal durante la operación normal, es decir cuando no se esta dentro de la zona donde se distorsiona la

lectura. En estas imágenes es claro que se presenta un fenómeno muy peculiar, donde el ESC cambia completamente su forma de operar en esta región, se puede apreciar que la señal se satura dando la impresión de que existe una señal cuadrada que es portadora de la antigua señal. Esta variación en la forma de operar de ESC no provoca ningún problema en el motor, ya que la velocidad sigue subiendo de manera normal, sin embargo, para el sensor esta distorsión no es normal, por lo que no puede realizar una lectura acorde a lo que realmente sucede, en su lugar hace lecturas de casi el triple del valor que se espera.



(a) Zona de distorsión promediando los valores



(b) Señal de velocidad con todas las modificaciones

Figura 6.8: Lecturas de 2 de las fases del motor y de la señal de PWM generada a la salida del sensor de velocidad.

Con el objetivo de corregir este fallo se optó por intentar soluciones de software

primero, antes de tomar la decisión de cambiar el hardware o buscar otro sensor. La primera modificación que se realizó al programa fue muy similar a la que se hizo con el sensor de corriente y consistió en hacer varias lecturas y luego promediarlas, de igual manera que en el caso de la corriente se tomaron cinco datos de velocidad para este cálculo. El resultado de promediar la señal se puede observar en la Figura 6.8.a, donde se puede observar una gran mejora con respecto a la gráfica anterior, ya que aunque los datos siguen presentándose en valores muy altos que no corresponden a los reales, ahora tienen todos una tendencia similar, que es la de ir en aumento casi con la misma pendiente que la señal cuando no está en la zona de distorsión. Finalmente se solucionó el problema de que los valores fueran tan altos, escalándolos entre un factor 3, para definir este valor se tomaron varias muestras y se promedio que este factor era de 2.95, sin embargo, se terminó por redondearlo a 3. Una vez que se han aplicado ambas modificaciones se puede observar que la gráfica de velocidad tiene la forma que se observa en la Figura 6.8.b, donde se presenta la curva a lo largo de todo el rango de velocidades posibles, primero variándola lentamente y luego un tramo en el que se subió y bajo la velocidad rápidamente, con el objetivo de verificar el comportamiento del sensor ante variaciones abruptas en la velocidad. Se pueden visualizar un poco de ruido presente en la lectura, sin embargo son bastante pocos e insignificantes, de manera general este sensor ofrece una buena precisión en sus lecturas y una vez aplicadas las correcciones al software no parece haber necesidad de realizar ninguna modificación inmediata al hardware.

6.2 Validación de los elementos de sujeción

Las piezas de sujeción diseñadas no pudieron someterse a pruebas reales por lo que se desarrolló una simulación de los esfuerzos a los que estos elementos podrían verse sometidos en el programa [49]. Esta simulación considera un aproximado de los pesos

presentes en el sistema los cuales se presentan en la Tabla 6.1 y fueron obtenidos del las hojas de datos respectivas o de los proporcionados por el software.

Tabla 6.1: Propiedades de los componentes del sistema de sujeción.

Pieza	Peso(N)	Esfuerzo de fluencia (MPa)
Soportes en ABS	0.5	28.5
4 Tornillos M2.5	0.03	340
2 Tornillos M4	0.04	340
2 Tuercas M4	0.01	340
4 Tuercas M2.5	0.006	340
Brazo de Aluminio	0.5	268
PCB	0.35	55
Relé	1	—
PCB Central	0.5	55

Como se observa en la Figura 6.9 se colocaron 2 fuerzas en este sistema, la primera es la gravedad, que afecta en su totalidad a toda la simulación con una aceleración de $9.81 m/s^2$, en la dirección hacia abajo como lo indica la flecha. Por otra parte, también se colocó una fuerza hacia arriba en el extremo del brazo, la cual representaría la fuerza de empuje ejercida por el motor durante el vuelo. Para el cálculo de esta fuerza se utilizó el empuje máximo ofrecido por la hoja de datos del motor, que equivale a 632 gramos y se redondeó a una fuerza de 6N. Aparte de estas fuerzas se fijo el otro extremo del brazo, ya que este es el que va fijo a la sección central del vehículo. Finalmente con el objetivo de simular las cargas del relé y el PCB se optó por colocar dos laminas de una alta densidad de manera que se pudiera igualar la masa y el peso al que equivaldrían estos componentes. La simulación fue realizada colocando el extremo de la pieza a una distancia de 120mm del punto de empotramiento.

Las simulaciones realizadas con [49], permiten obtener el esfuerzo de Von Mises, que a su vez permiten calcular el factor de seguridad del diseño con ayuda de la ecuación 6.1, la cual se ha obtenido de [50]. Conocer el factor de seguridad permite determinar la relación entre la capacidad del sistema diseñado con respeto a los requerimientos del mismo.

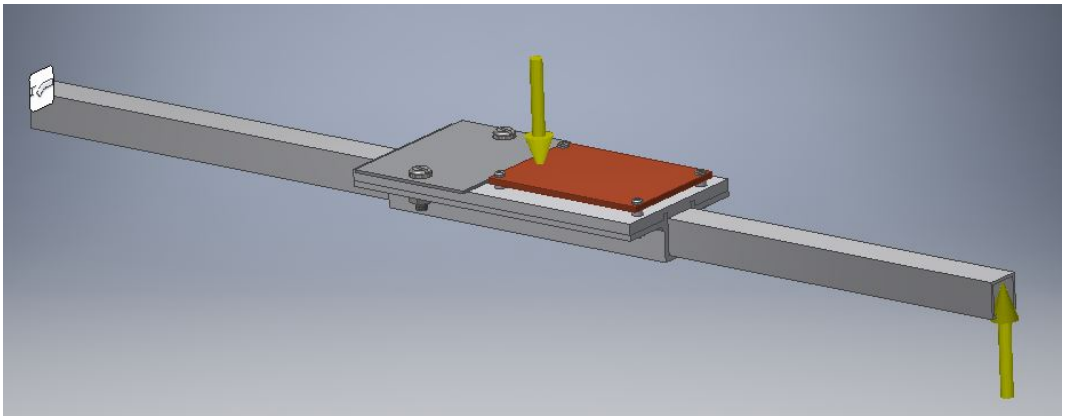


Figura 6.9: Simulación diseñada para el soporte que va sobre el brazo del UAV.

$$FS = \frac{\sigma_{fluencia}}{\sigma_{VM}} \tag{6.1}$$

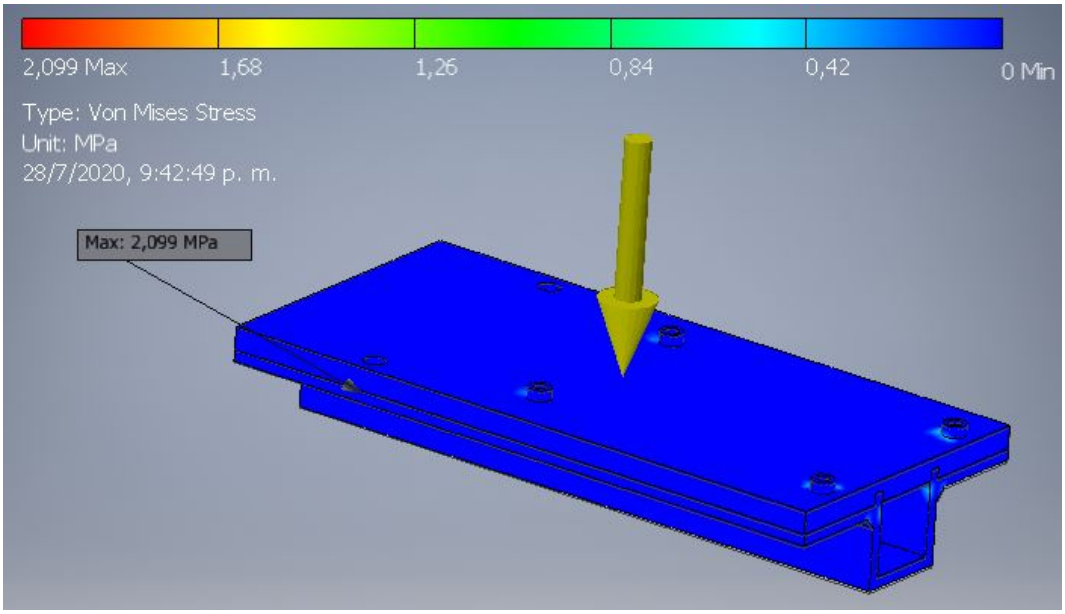


Figura 6.10: Esfuerzo de Von Mises en soporte de ABS.

En la Figura 6.10 se presenta el análisis de esfuerzos realizado al soporte de ABS, donde el mayor esfuerzo se presenta entre el contacto de este con una de las tuercas, produciendo un esfuerzo máximo de 2.099 MPa, con el cual se puede calcular el factor de seguridad en la ecuación 6.2. El resultado fue de 13.58, lo cual indica que el diseño tiene u buen margen con respecto a sus esfuerzos críticos y se encuentra lejos de fallar.

Para este cálculo se utilizó el esfuerzo de 28.5 MPa, sin embargo, como se mencionó en la Sección 5.1.3.1, existen configuraciones para las que este esfuerzo puede llegar a ser de hasta 9.08 MPa, aunque en este caso ni utilizando esa configuración el diseño fallaría. Finalmente se reafirma que no existe problema en utilizar ABS como material para los elementos de sujeción, no hay necesidad de un material de mayor esfuerzo de fluencia como el PLA, ya que con el esfuerzo actual se obtiene un alto factor de seguridad.

$$FS = \frac{28.5MPa}{2.099MPa} = 13.58 \tag{6.2}$$

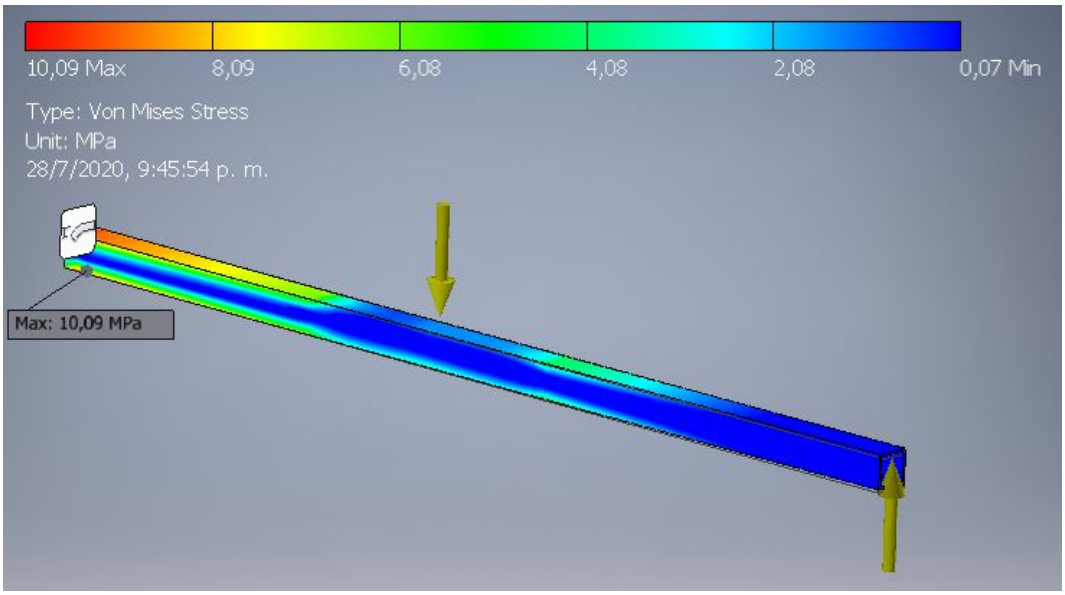


Figura 6.11: Esfuerzo de Von Mises en el brazo de Aluminio.

Se decidió simular el brazo para verificar que el peso extra en la mitad no representará un gran problema, sin embargo este peso no es suficiente para contrarrestar la fuerza de empuje del motor, como se observa en la Figura 6.11, el mayor esfuerzo se presentaría en la región empotrada en la carcasa central, en donde habría un esfuerzo de aproximadamente 10.09MPa, la cual considerando el esfuerzo máximo de 268MPa para este material, le da un factor de seguridad de hasta 26.56. Esta no es una pieza que se pueda modificar, ni que fuera parte del diseño, sin embargo, se comprueba que conserva un alto factor de seguridad a pesar llevar la placa de sensorización.

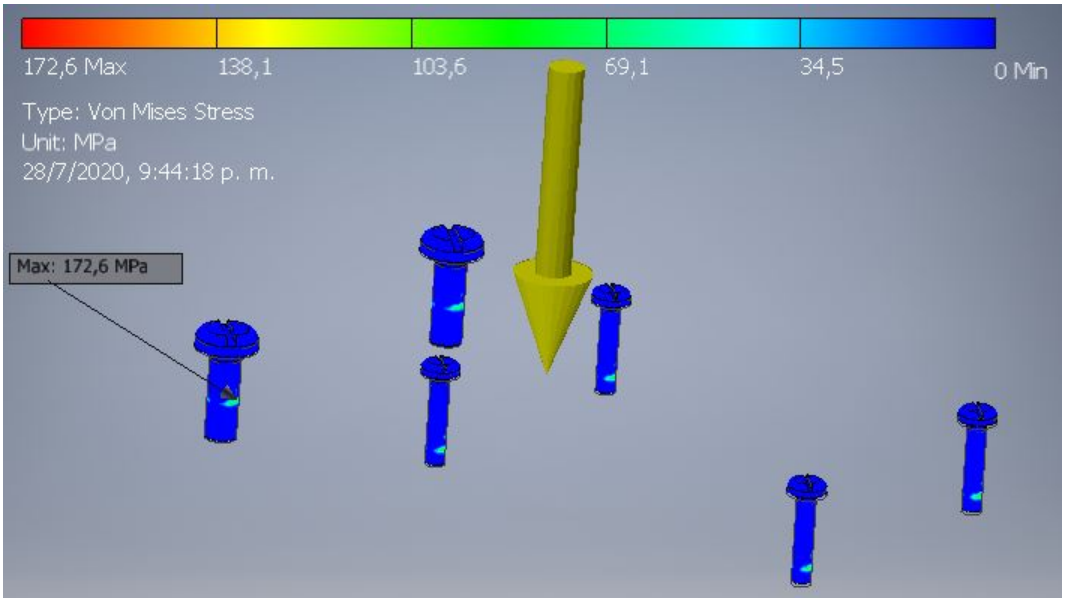


Figura 6.12: Esfuerzo de Von Mises en los tornillos de grado 4.8.

El siguiente elemento que se sometió a análisis fueron los tornillos, que como se observa en la Figura 6.12, se presenta un esfuerzo máximo de Von Mises de 172.6MPa en uno de los tornillos M4. En cuanto al factor de seguridad se calculó como se observa en ecuación 6.3, obteniéndose un valor de 1.97 . Este resultado no es tan alentador como los otros, sin embargo, todavía se encuentra dentro de lo aceptable, donde bajo condiciones similares no deberían fallar. Este elemento es capaz de soportar hasta 167.4 MPa más de lo esperado antes de fallar.

$$FS = \frac{340MPa}{172.6MPa} = 1.97 \tag{6.3}$$

Se realizó en análisis en la tuercas, en las cuales como se puede observar en la Figura 6.13, se presenta el esfuerzo más elevado de todo el sistema, con un valor de 159.7 MPa, en una de las tuercas M2.5, que a su vez da como factor de seguridad un 2.13, como se ve en el cálculo de la ecuación 6.4. Se considera aceptable todo factor de seguridad por encima de 1 y aunque en este caso se presente un valor de 2.13 y parezca muy cercano a 1, la diferencia entre el esfuerzo de fluencia y el que se presenta es de 180.3MPa, diferencia bastante alta, por lo que se puede aceptar este diseño.

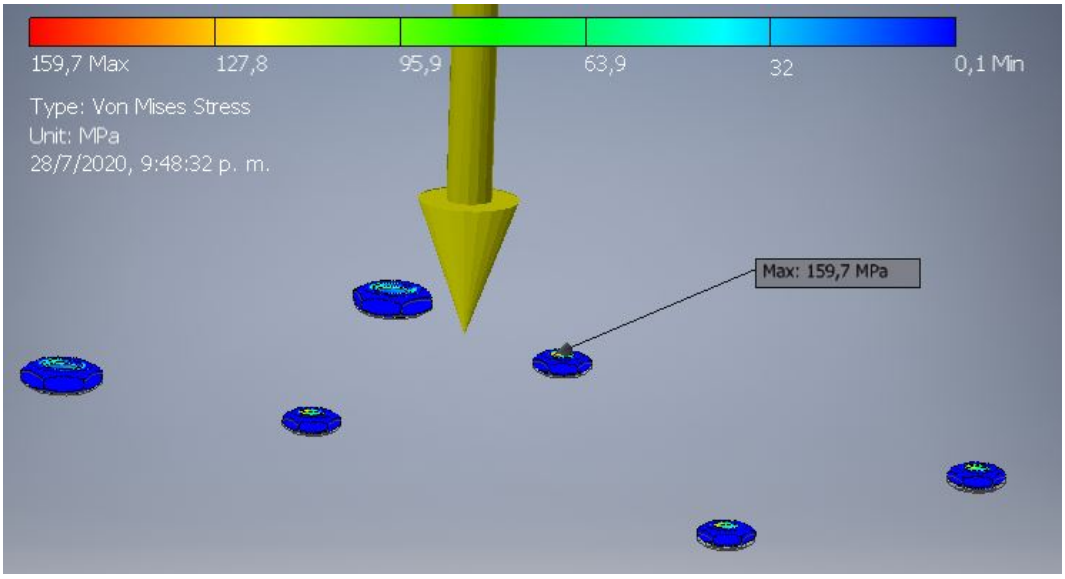


Figura 6.13: Esfuerzo de Von Mises en las tuercas de Acero grado 4.8.

$$FS = \frac{340MPa}{159.7MPa} = 2.13 \tag{6.4}$$

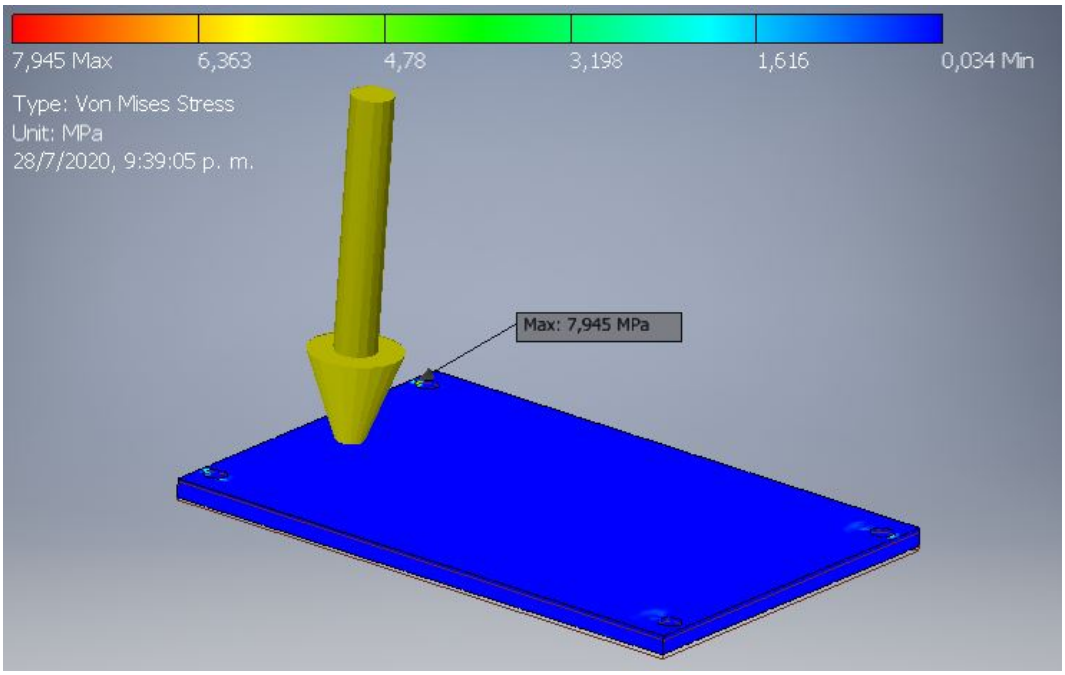


Figura 6.14: Esfuerzo de Von Mises en la placa del PCB.

Se realizó otro análisis más a la placa del PCB, la cual se compone principalmente de baquelita, aunque posee una capas de cobre. El esfuerzo máximo soportado por la

baquelita es de 55MPa y como se observa en la 6.14, esta sufre esfuerzos de 7.945MPa, lo cual le da un factor de seguridad de 7.33, por lo que es poco probable que esta placa sufra algún fallo debido a fuerzas externas.

Finalmente se realizó una prueba para analizar el desplazamiento del soporte de ABS, producto de los pesos que van sobre este, para realizar esta prueba se fijo el brazo como un elemento rígido, de manera que se pudiera apreciar solo el desplazamiento de las partes del soporte, y ver si podría originar algún problema como un pandeo del PCB. En la Figura 6.15 se muestra que el mayor desplazamiento se presenta en las aletas del soporte, como es de esperarse, sin embargo, alcanza un valor de 1.183e-04 milímetros como máximo, lo cual es altamente despreciable en una estructura que tiene todas sus en valores de hasta 4 ordenes mayor, por lo que no se espera que el desplazamiento sea un problema en este soporte.

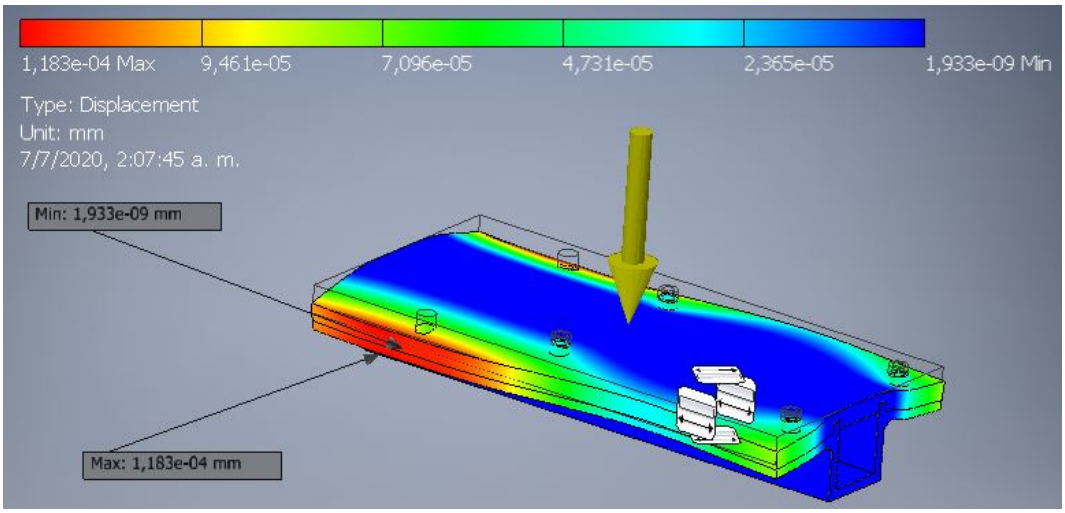


Figura 6.15: Desplazamiento de la estructura de ABS.

Para el montaje del segundo PCB se prescindió de realizar una simulación, ya que en este caso solo se realizaron unas pequeñas modificaciones, agregando 4 agujeros de 2.5mm para atornillar el PCB a la carcasa. Si bien un agujero puede representar un concentrador de esfuerzo, como se observa en el trabajo de [46], en esta carcasa los esfuerzos se concentran en la parte inferior de la carcasa ya que se lleva un peso de

1.21 kilogramos dentro de la unidad central, por lo que agregar 50 gramos de un PCB no haría una gran diferencia. Sin embargo en este trabajo se demuestra que el factor de seguridad más bajo se encuentra en la zona de la carcasa que tiene contacto con los pernos que lo conectan con la pieza de la Figura 6.16, por lo que se procederá a recalcular este factor con el peso del PCB agregado.

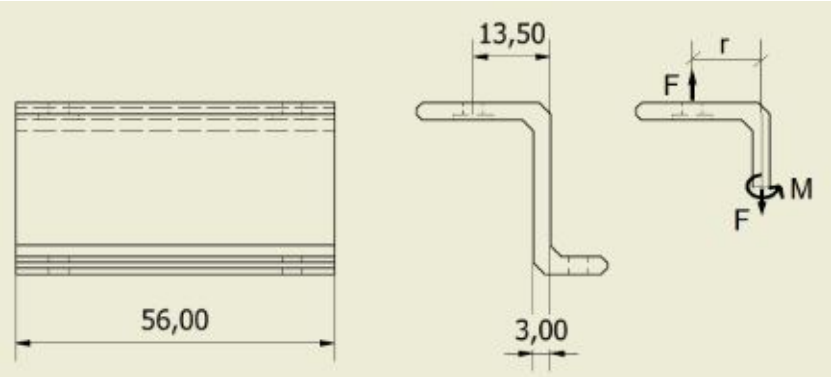


Figura 6.16: Unión entre la carcasa central y el UAV [46].

Utilizando el esfuerzo de 28MPa para el ABS en el trabajo de [46], el factor de seguridad para esta región critica seria de 13.87, sin embargo al agregar el peso de 0.5N del PCB se recalcula este factor como se observa en la ecuación 6.5, obteniéndose un nuevo factor de 13.29. El resultado representa tan solo una variación de 0.58 en el factor de seguridad y aun se mantiene bastante alto, por lo que en esta sección el agregar este peso no representa un problema.

$$FS = \frac{\sigma_{fluencia}}{\frac{F}{A_{contacto}}} = \frac{28[MPa]}{\frac{18.6[N]}{8.83e{-}6[m]}} = 13.29 \tag{6.5}$$

Finalmente se analizo para este diseño, la fuerza de aplastamiento que sufre la estructura directamente por el peso del PCB, sobre los relieves en los cuales van los pernos que lo sujetan, como se ven en la Figura 6.17. Esta sección circular tiene la forma de un anillo de 3mm de radio exterior y 1.25mm de radio interior. Para el cálculo del esfuerzo se utilizó la carga total de 0.5N, asumiendo un caso crítico, sin embargo esta fuerza debería distribuirse por entre los cuatro soportes. En la ecuación 6.6 se puede

observar el cálculo de del factor de seguridad de este segmento.

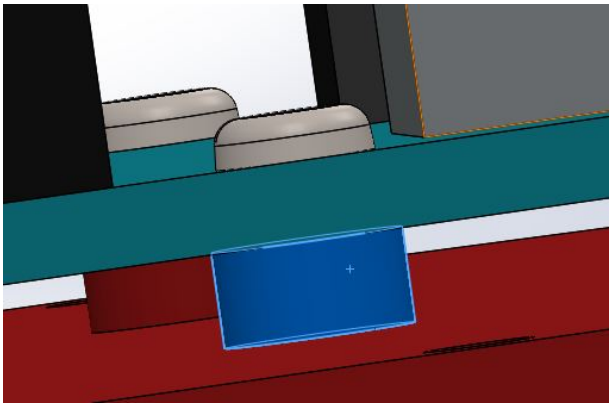


Figura 6.17: Soporte que sostiene el PCB para evitar el contacto directo con la carcasa.

$$FS = \frac{28[MPA]}{\frac{0.5[N]}{(0.003^2 - 0.00125^2) * \pi[m]}} = 416.5 \tag{6.6}$$

Con un resultado final de 416.5 como factor de seguridad para este elemento, se descarta que al agregar el peso del PCB y los cuatro agujeros de 2.5mm a la carcasa central, se alterar el diseño al punto de hacerlo peligroso. Agregar este PCB resulta en un cambio prácticamente nulo, de manera que se puede conservar este rediseño propuesto.

6.3 Depuración del modelo de predicción, detección y clasificación de fallos

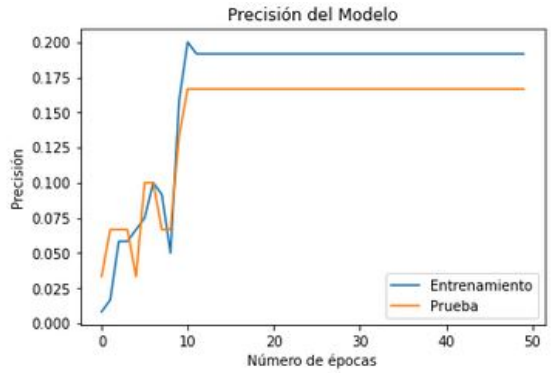
En el capítulo anterior se describió el modelo inicial propuesto para la red neuronal recurrente que se pretende utilizar para la predicción y detección de fallos, sin embargo, los resultados obtenidos estuvieron muy lejos de ser considerados aceptables, obteniendo precisiones de un máximo de 30%. Es por eso por lo que esta sección se analizará el peso de cada una de las variables de la red y se buscará la manera de rediseñar hasta obtener mejores resultados. Las mejoras se harán en base a la precisión por lo que se

buscará que conservar la variación que tenga el valor más alto de esta.

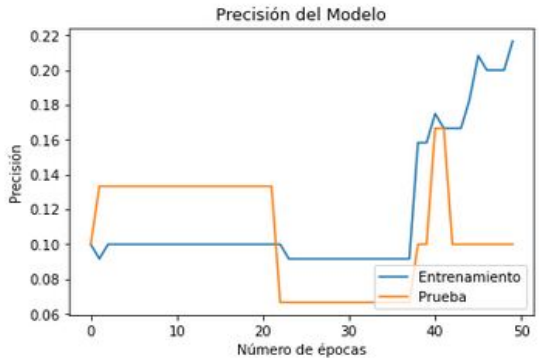
6.3.1 Optimizador

El optimizador, como su nombre lo dice, es el encargado de optimizar la red neuronal, cuando se optimiza una función matemática se hace por lo general buscando mínimos o máximos, en el caso de las redes neuronales lo que se busca es buscar el mínimo de la función de error. De manera general los optimizadores usan como base las derivadas parciales, utilizadas en los métodos de optimización comunes, sin embargo, existen variaciones que se aplicaron en este problema para seleccionar el mejor optimizador para el diseño de esta red.

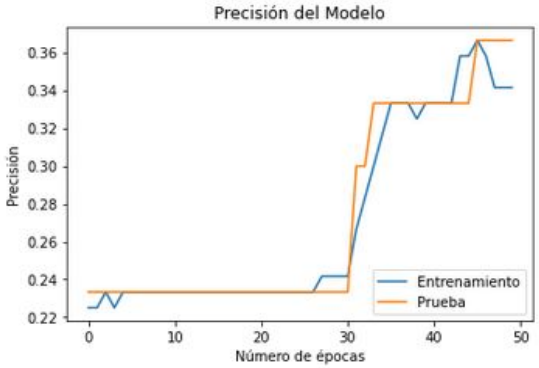
El primer modelo se planteó con el optimizador llamado descenso de gradiente estocástico (SGD), el cual es uno de los optimizadores más sencillos, de manera general este optimizador tiene un valor prefijado en su tasa de aprendizaje (la cual actúa como un factor que regula el gradiente de los pesos) y actualiza los pesos cada vez que procesa un dato o un lote en el caso de utilizarse. Como se puede observar en la Figura 6.18 los resultados obtenidos con esta función de optimización no son para nada buenos, ya que este primer modelo solo permite obtener precisiones del 20%.



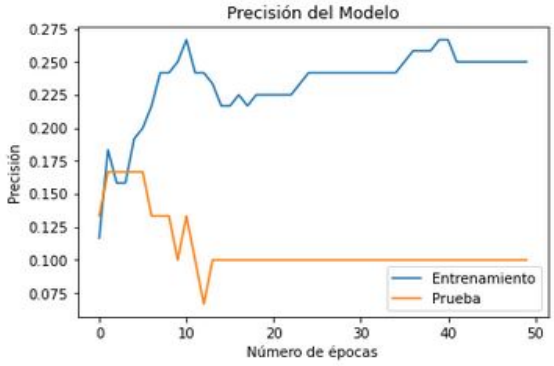
(a) SGD en LSTM



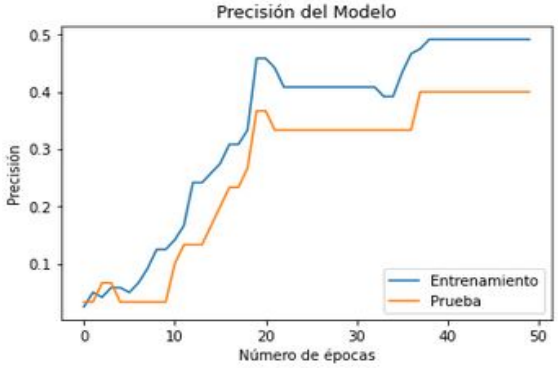
(b) SGD en GRU



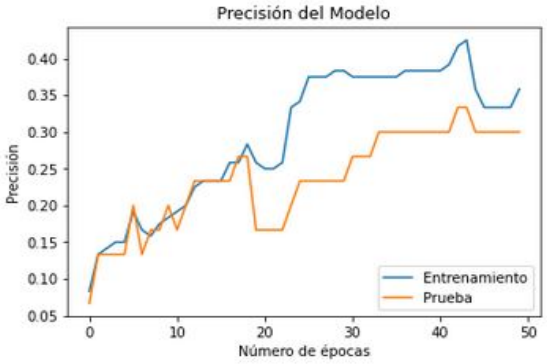
(c) Adagrad en LSTM



(d) Adagrad en GRU



(e) Adam en LSTM



(f) Adam en GRU

Figura 6.18: Gráficas de precisión variando la función de optimización en redes LSTM y GRU.

La segunda prueba se llevo a cabo con un optimizador denominado Adagrad, la principal diferencia entre esta función y la SGD es que posee una tasa de aprendizaje adaptativa, lo cual le permite variar el tamaño de los pasos que da en busca de acercarse a un mínimo local. En las gráficas se puede ver que este optimizador ofrece mejores

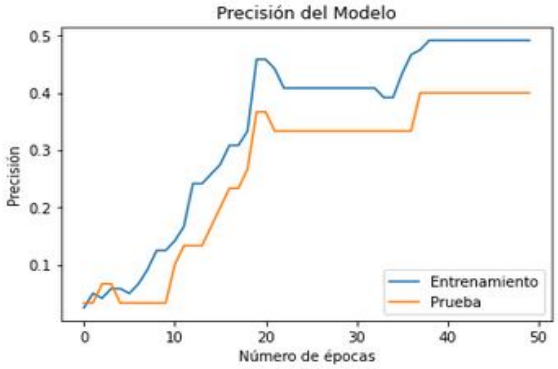
resultados, al menos en la red LSTM logra alcanzar precisiones de hasta 37%, por lo que si ofrece una mejora con respecto al SGD clásico. Finalmente, se probó el optimizador conocido como Adam, el cual es considerado uno de los más robustos y utilizados en la actualidad, aparte de poseer una tasa de aprendizaje adaptativa como Adagrad, integra un concepto de momentum adaptativo, donde momentum hace referencia a una constante que considera los gradientes pasados y define que tanta importancia tienen estos en el cálculo del gradiente actual. Las gráficas obtenidas al aplicar este optimizador son sin duda las mejores, tanto para a la red LSTM como para la GRU permitiéndoles alcanzar precisiones de 50% y 40% respectivamente, por lo que se selecciona este como nuevo optimizador.

6.3.2 Función de error

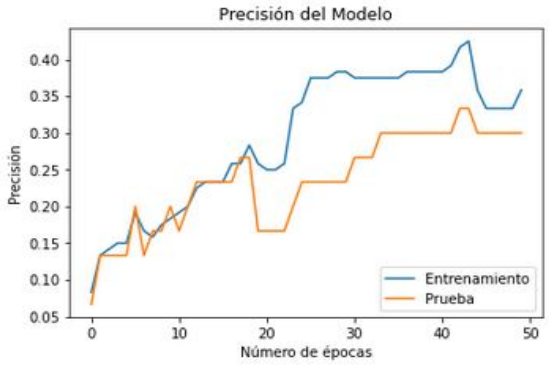
La función de error es aquella que permite a la red neuronal evaluar los valores predichos y compararlos con los esperados. Como se mencionó anteriormente, esta función es la que es optimizada en busca del valor mínimo de error posible. Dado que para la solución de este proyecto se optó por una red de clasificación, existen funciones de error específicas para utilizar en estos casos, por lo que se probaron algunas de ellas con el objetivo de ver cual ofrecía mejores resultados en busca de mejorar la precisión de la red.

En la Figura 6.19 se pueden observar las gráficas obtenidas al probar tres funciones de error diferentes, la primera es la ya seleccionada para el modelo inicial, que es la entropía cruzada categórica, que es utilizada en problema de clasificaciones con múltiples clases y es muy común en los problemas de clasificación, sin embargo, en este caso no parece muy favorable en la obtención de altas precisiones por parte la red neuronal. Por otra parte, a segunda función utilizada fue la entropía cruzada binaria, esta es una variación de la categórica, que igual esta regida por una función logarítmica, sin embargo esta se especializa en problemas binarios donde las salidas solo pueden ser

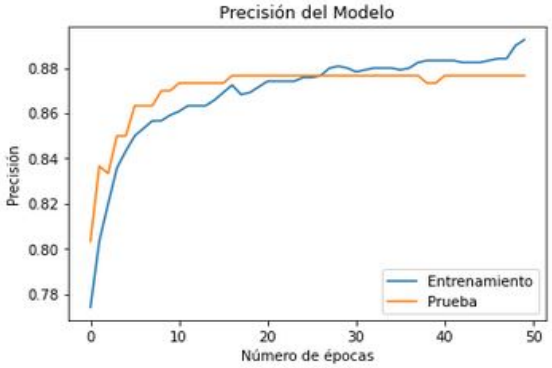
dos, como es el caso de este problema, donde se asignaron categorías con probabilidad de 0 o 1, sin posibilidad de valores intermedios, esta función de error fue la que ofreció mejores resultados haciendo que la precisión pasara del 50% hasta un 90%.



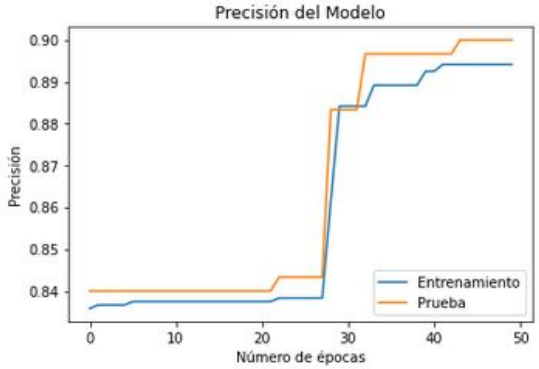
(a) Categorical Crossentropy en LSTM



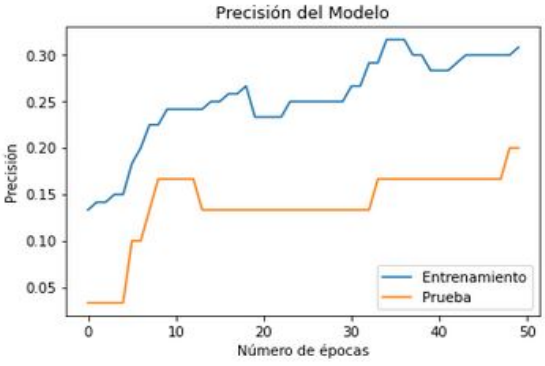
(b) Categorical Crossentropy en GRU



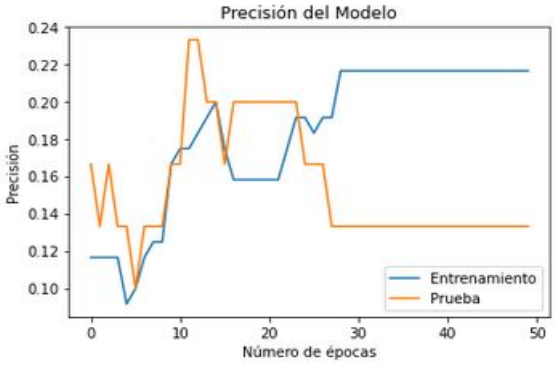
(c) Binary crossentropy en LSTM



(d) Binary crossentropy en GRU



(e) Hinge en LSTM



(f) Hinge en GRU

Figura 6.19: Gráficas de precisión variando la función de error en redes LSTM y GRU.

Finalmente, se probó con una función más, conocida como pérdida de Hinge, la

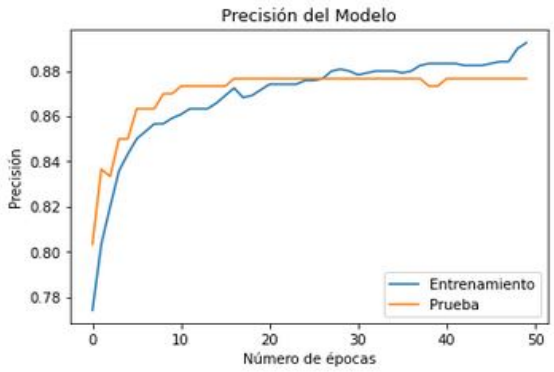
cual también se utiliza principalmente en problemas binarios, como la entropía cruzada binaria, sin embargo, la función de Hinge está regida por una recta lineal, en lugar de una logarítmica, además de ser mas estricta a la hora de castigar valores de error. Como se puede apreciar, esta función es la que ofrece peores resultados, implica incluso una reducción en los valores de precisión que ya se tenían, llegando a valores del 30% como máximo. Considerando las tres funciones es claro que la más conveniente para el diseño de la red es la entropía cruzada binaria.

6.3.3 Función de activación de la salida

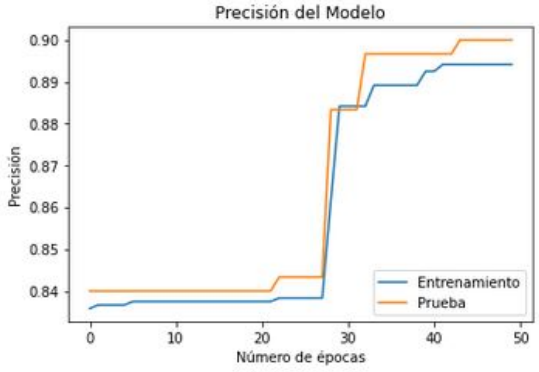
La función de activación de la capa de salida es aquella recibe los valores que salen de todas las neuronas de la ultima capa oculta y los procesa de manera que se genere un valor a la salida. Por lo general se definen funciones de activación para cada una de las capas de la red neuronal, pero en el caso de las redes LSTM y la GRU ya tiene dentro de sus estructuras sus propias funciones de activación. Para demostrar el efecto de variar la función de activación se probaron tres funciones comúnmente utilizadas, pero diferentes entre, de manera que se pudiera seleccionar la que mejor se desempeño mostrara.

En la Figura 6.20 se pueden observar las gráficas para las tres funciones de activación probadas, el modelo inicial se planteo con la función ReLU, la cual es una función sin acotar que designa cero a la salida para valores negativos en la entrada. Los resultados con esta función ya eran buenos, lográndose un 90% de precisión en la red GRU y un 88% en la LSTM. Por otra parte, se probó también la función sigmoide, la cual esta acotada entre 0 y 1 lo cual resulta idea a la hora de trabajar con probabilidades, como sucede en redes de clasificación, ya que también se manejan entre estos valores. Como se puede ver en las curvas obtenidas, esta función logró que ambas redes convergieran a un 90% de precisión lo que implica una mejora, además de que la transición se da de manera natural y no abruptamente como ocurre en la curva de GRU con la función

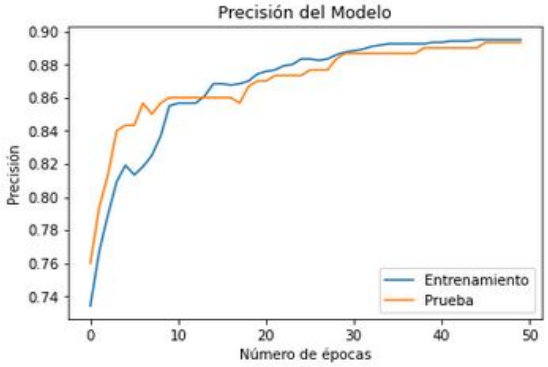
ReLU.



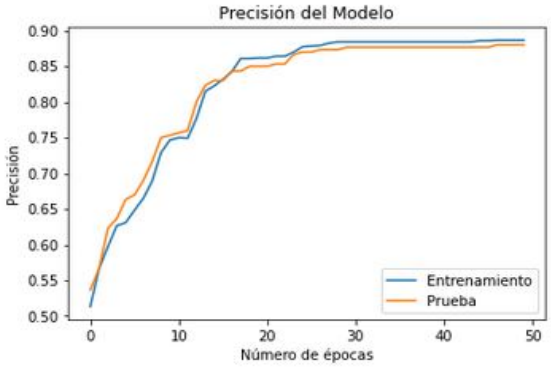
(a) ReLU en LSTM



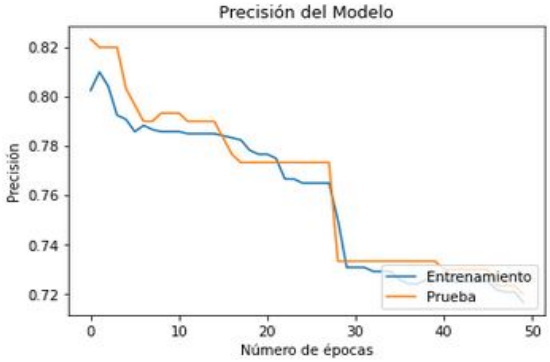
(b) ReLU en GRU



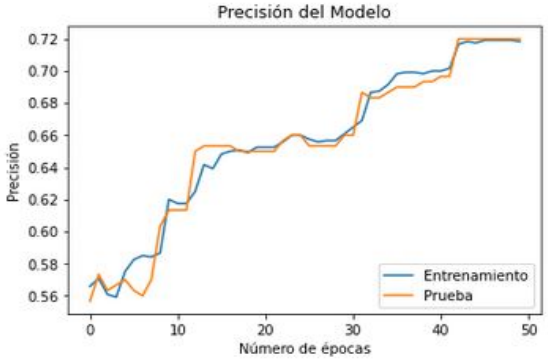
(c) Sigmoide en LSTM



(d) Sigmoide en GRU



(e) Tangente hiperbólica en LSTM



(f) Tangente hiperbólica en GRU

Figura 6.20: Gráficas de precisión variando la función de activación en la capa de salida en redes LSTM y GRU.

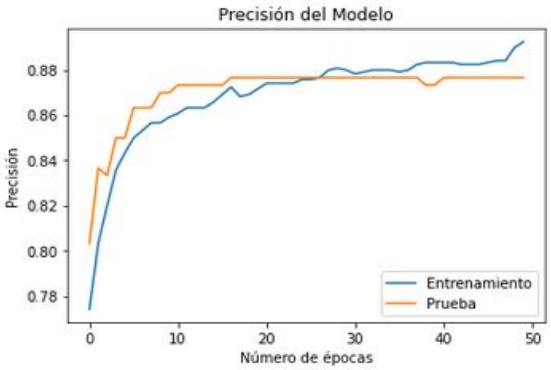
La última función probada fue la tangente hiperbólica, la cual es muy similar a la sigmoide pero esta acotada entre -1 y 1, por lo que permite valores negativos a la salida, es posible que esta sea la razón por la que su desempeño sea tan malo, ya que como

se observa en las gráficas, a diferencia de las otras, esta curva de precisión tiende a bajar en lugar de subir terminando luego de las 100 épocas en un valor de precisión de aproximadamente 70 %. Considerando todos los resultados, se cambió la función ReLU propuesta inicialmente, por una función sigmoide, ya que aunque fuera por poco, demostró tener un mejor desempeño.

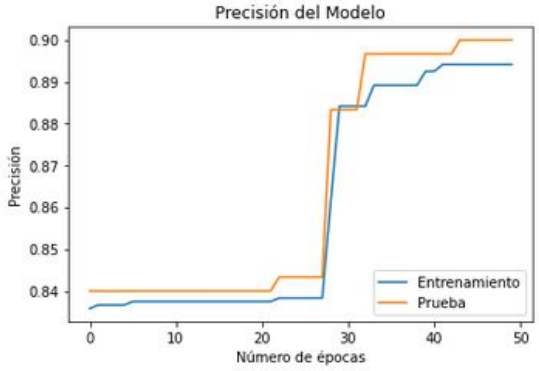
6.3.4 Número de capas ocultas

Agregar más capas o neuronas no implica directamente que se contará con mejores resultados. Aumentar el número de capas ocultas aumenta la complejidad de la red, por lo que el procesamiento de los datos toma más tiempo, agregar una capa equivale a agregar otro conjunto de variables a la ecuación matemática que representa una red neuronal, esto puede permitirle a la red explorar regiones o soluciones que no se han explorado, sin que esto implique que sean mejores, lo que si se puede decir es que aporta una mayor variabilidad al sistema.

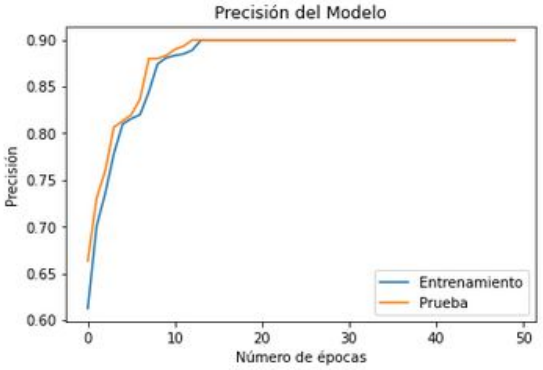
En la Figura 6.21 se pueden observar las gráficas de precisión al variar el número de capas ocultas, tanto para una red LSTM como para una GRU. Se probaron las configuraciones de 1, 2 y 3 capas ocultas, donde es claro que al colocar dos o tres capas se logran mejores resultados que cuando solo se utiliza una. No solo por la rapidez con la que la curva logra converger a valores altos de precisión si no también porque alcanza mejores valores. Ahora entre colocar dos y tres capas no existe tanta diferencia, ambas convergen muy rápido y lo hacen al mismo valor de precisión, es indiscutible que con tres capas se converge un poco antes, sin embargo, la diferencia es mínima, por lo que en busca de reducir la complejidad de la red se selecciona el modelo de dos capas para ambas redes neuronales.



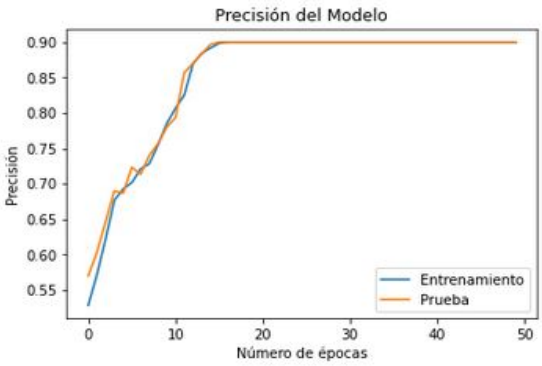
(a) 1 capa oculta LSTM



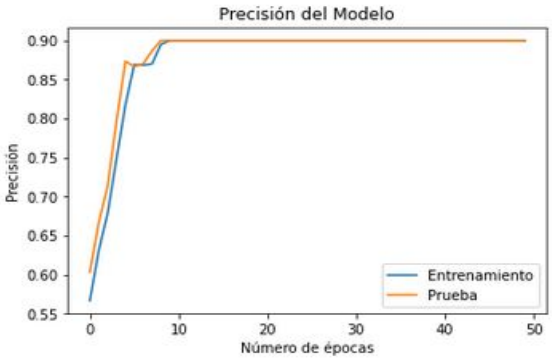
(b) 1 capa oculta GRU



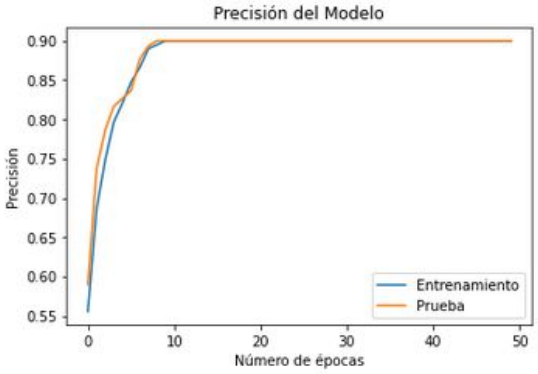
(c) 2 capas ocultas LSTM



(d) 2 capas ocultas GRU



(e) 3 capas ocultas LSTM



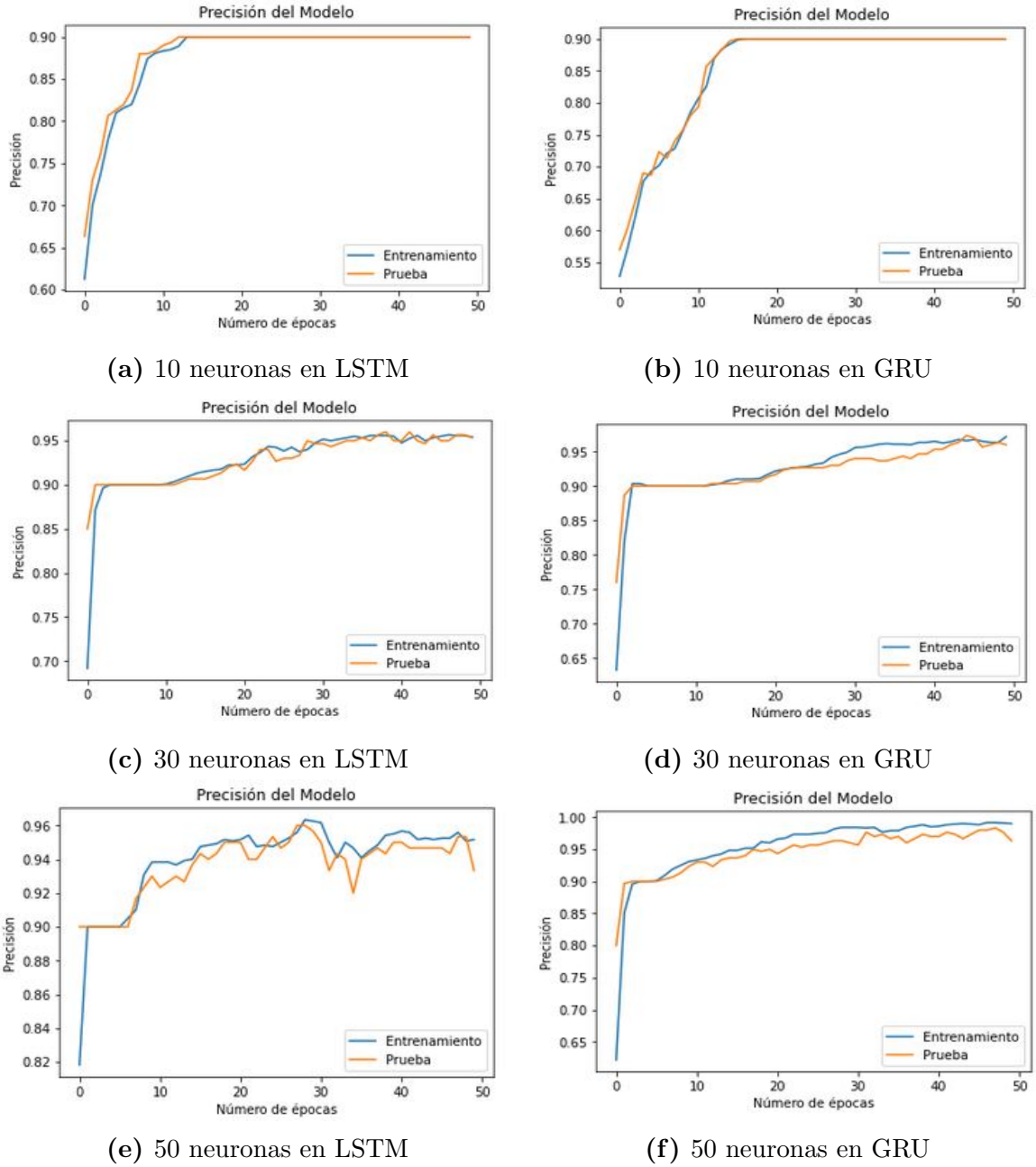
(f) 3 capas ocultas GRU

Figura 6.21: Gráficas de precisión variando la cantidad de capas ocultas en redes LSTM y GRU.

6.3.5 Número de neuronas de las capas ocultas

Cuando se varía el número de neuronas aplica la misma lógica que con el número de capas, una mayor cantidad de neuronas aumenta el orden del sistema, lo que le permite explorar más opciones, aunque esto no significa que se vaya a lograr converger a una

mejor solución y en ocasiones solo se estaría demandando mas recurso computacional al elevar la complejidad de la red neuronal.



(a) 10 neuronas en LSTM

(b) 10 neuronas en GRU

(c) 30 neuronas en LSTM

(d) 30 neuronas en GRU

(e) 50 neuronas en LSTM

(f) 50 neuronas en GRU

Figura 6.22: Gráficas de precisión variando la cantidad de neuronas de las capas ocultas en redes LSTM y GRU.

En la Figura 6.22 se encuentran las gráficas de precisión obtenidas al probar diferentes números de neuronas. Se tenía un modelo inicial con 10 neuronas que lograba converger

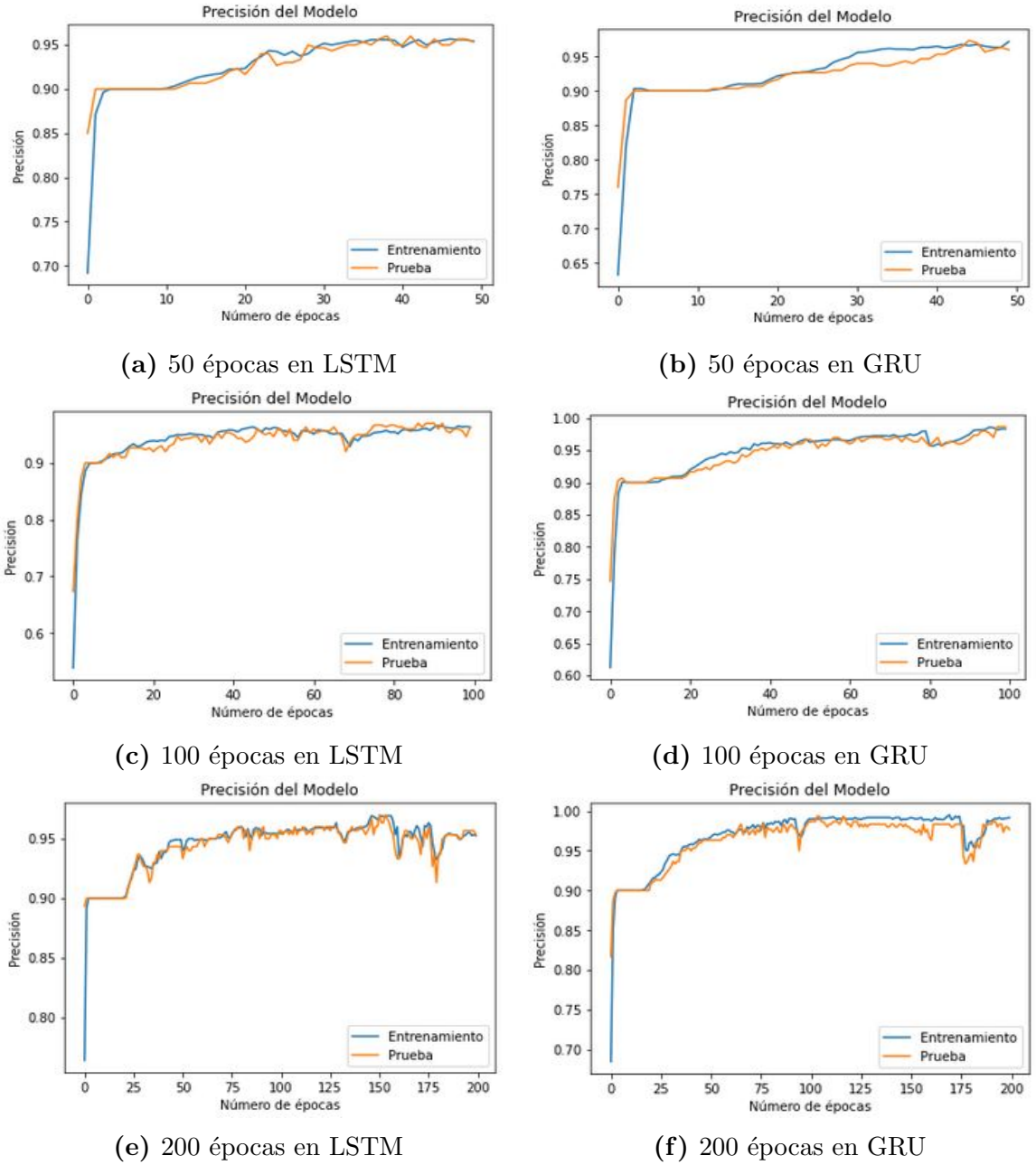
a un 90% de precisión, por lo que se optó por probar aumentando la cantidad de neuronas a 30 y 50, obteniendo en valores de aproximadamente 95% de precisión, por lo que es claro que el aumento de neuronas si aportó una mejora en este tipo de problema. Entre utilizar 30 o 50 neuronas es preferible utilizar la configuración de la 30, ya que ambas convergen a prácticamente el mismo valor, sin embargo, el modelo con 30 lo logra con una menor complejidad en la arquitectura de la red neuronal.

6.3.6 Número de épocas

El número de épocas hace referencia a la cantidad de veces que se pasa por la red neuronal todo el conjunto de datos, en el modelo inicial se proponía utilizar 50 épocas, eso quiere decir que la red recorría en su totalidad los datos 50 veces, actualizando los pesos por cada vez que terminaba con un lote. Aumentar el número de épocas permite dar más oportunidades a la red de converger a otros valores, ya que se podría decir que tiene un mayor número de oportunidades de detectar los patrones presentes en los datos. Sin embargo, en ocasiones aumentar el número de épocas también puede llevar a la red a una reducción de su precisión, ya que algunas actualizaciones de pesos pueden intentar mejorar el desempeño de la red con ciertos datos, pero al costo de alterar la forma en la que ya procesaba los demás, además entrenar una red en exceso no es apropiado, ya que se puede incurrir en un sobre entrenamiento, esto se puede entender como forzar a la red a aprenderse los datos de entrenamiento y no los patrones que reflejan los fallos, de manera que si luego se enfrenta a un conjunto de datos nuevo puede no brindar buenos resultados.

En la Figura 6.23 se puede observar el efecto entrenar la red durante diferente número de épocas. Inicialmente se entrenaba a lo largo de 50 épocas, como se puede ver en las primeras dos gráficas, consiguiendo precisiones de un 95%, sin embargo, al aumentar el número a 100 épocas se puede ver una pequeña mejora que ha llevado al sistema a una precisión del 98% para a red GRU, por otra parte al seguir subiendo

utilizar 200 épocas, los resultados son muy similares a los obtenidos con 100, con el defecto de que la señal comienza a oscilar un poco más entre épocas. Considerando todo esto se seleccionó el modelo de 100 épocas, debido a que son suficiente cantidad para ofrecer mejora y subir más esta cantidad podría provocar que el entrenamiento acabe con una precisión mucho más baja por tratar de buscar mejores soluciones.



(a) 50 épocas en LSTM

(b) 50 épocas en GRU

(c) 100 épocas en LSTM

(d) 100 épocas en GRU

(e) 200 épocas en LSTM

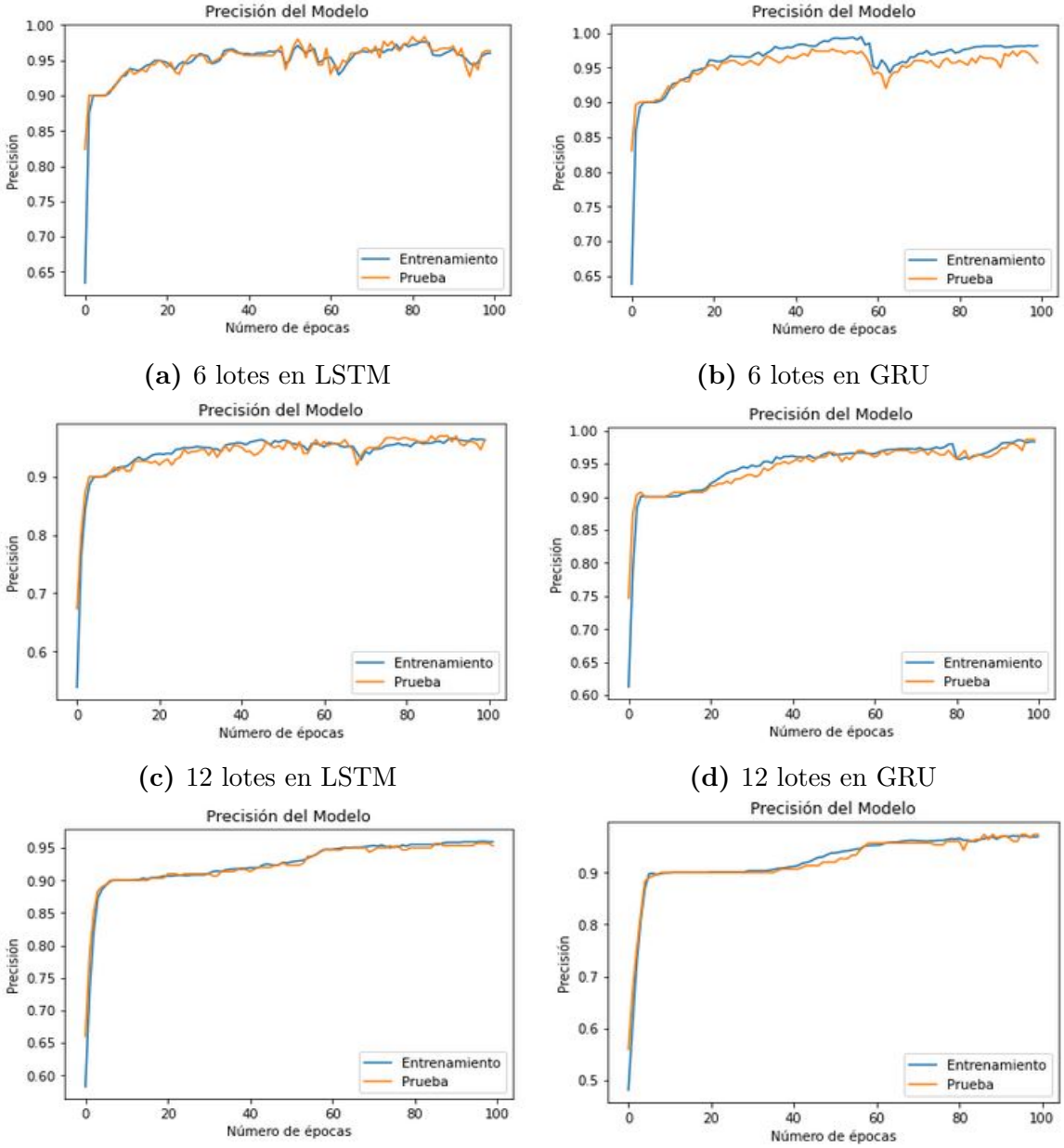
(f) 200 épocas en GRU

Figura 6.23: Gráficas de precisión variando la cantidad de épocas en redes LSTM y GRU.

6.3.7 Tamaño del lote

El último parámetro de la red que se varió fue el tamaño del lote, como se mencionaba anteriormente, una época representa una pasada del conjunto de datos completo por la red neuronal, el concepto de lote por otra parte se utiliza cuando el conjunto de datos agrupado en subgrupos a lotes para el entrenamiento. El tamaño del lote representa a la cantidad de datos que conforman cada uno de estos subgrupos, a nivel de funcionamiento, una red neuronal actualiza sus pesos cada vez que termina de procesar un lote, esto quiere decir que cuando los lotes son pequeños o de un solo dato, la red debe actualizar sus pesos muchas veces durante una sola época, lo cual implica mucho más tiempo de entrenamiento. Actualizar los pesos dato por dato puede producir que haya muchas variaciones en los mismos ya que pueden existir muchas diferencias de un dato a otro, al actualizarlos en bloques se permite entrenar a la red para casos mas generalizados.

En la Figura 6.24 se presentan las gráficas generadas al variar el tamaño del lote, el modelo actual utiliza un tamaño de 12 datos por lote y ofrece valores de precisión bastante altos, sin embargo se realizaron pruebas disminuyendo y aumentado este número. El disminuir el tamaño del lote a 6 se pueden observar que las gráficas pasan por valores de precisión muy altos en algunas de sus épocas, sin embargo, no es un resultado muy estable ya que se presentan muchas variaciones de una época a otra, propiciando que al final de las 100 épocas converjan a valores cercanos al 95% de precisión, que era el valor que ya se había alcanzado anteriormente. Por otra parte, al aumentar el tamaño se obtienen curvas mucho más suaves, la transición de una época otra es lenta pero en un constante ascenso, sin embargo al finalizar todas las épocas no logra valores de precisión tan altos como al utilizar un tamaño de lote de 12, por esta razón al final del problema se conserva esta variable igual a la seleccionada para el modelo inicial.



(a) 6 lotes en LSTM

(b) 6 lotes en GRU

(c) 12 lotes en LSTM

(d) 12 lotes en GRU

(e) 24 lotes en LSTM

(f) 24 lotes en GRU

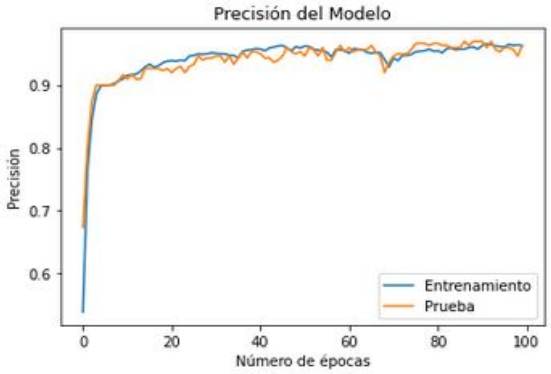
Figura 6.24: Gráficas de precisión variando la cantidad de lotes en redes LSTM y GRU.

6.4 Modelo final

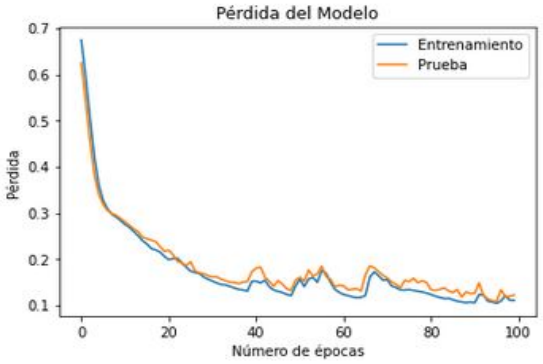
Una vez que se han escogido la mejor combinación de los parámetros de la red se propone el resultado como el modelo final que será utilizado para la predicción, detección y clasificación de fallos. En la Tabla 6.2 se resumen los parámetros que fueron elegido para para el modelo final. Comparándolo con el modelo inicial, la única variable que se conservó igual fue el tamaño del lote, todas las demás se cambiaron con el objetivo de obtener un valor más alto de precisión. En la Figura 6.25 se pueden ver las gráficas de precisión y pérdida del modelo final, tanto para red GRU como para la LSTM, como se observa ambas logran valores de precisión que están alrededor del 95%, lo cual implica una gran mejora con respecto al inicial que se encontraba en un 30% en el mejor de los casos. Además, ahora las curvas de error convergen a valores cercanos a cero y menores a 1, lo cual no ocurría en el modelo inicial y como se mencionaba anteriormente, al tenerse una salida que esta entre 0 y 1, por ser una probabilidad, se espera que el error se mueva dentro de esta franja y sea lo más pequeño y cercano a cero posible.

Tabla 6.2: Configuración final de la red neuronal diseñada para la predicción, detección y clasificación de fallos

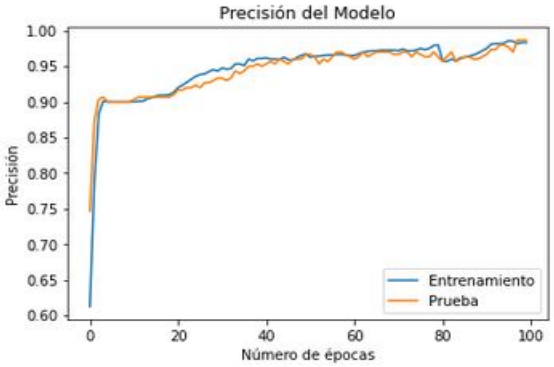
Parámetro	Valor o configuración
Optimizador	Adam
Función de error	Entropía cruzada binaria
Función de activación de la salida	Sigmoide
Número de capas ocultas	2
Número de neuronas de las capas	30
Número de épocas	100
Tamaño del lote	12



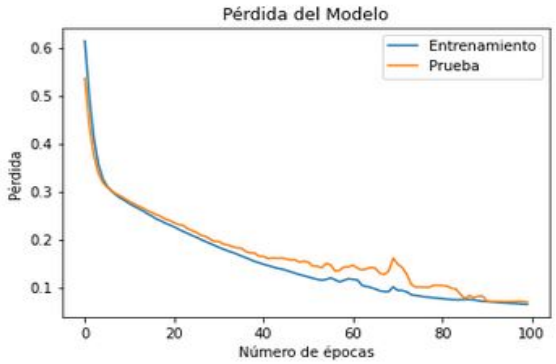
(a) Precisión LSTM



(b) Error LSTM.



(c) Precisión GRU



(d) Error GRU.

Figura 6.25: Gráficas de precisión y error para el modelo final diseñado según la Tabla 6.2 utilizando capas LSTM y GRU.

6.4.1 Validación del modelo

Para la validación del modelo final se utilizó el conjunto de 5000 datos generados, independientes de los datos de entrenamiento, de manera que son datos que la red neuronal nunca ha visto. Aunque la validación podría limitarse a estos datos, se decidió establecer la precisión y error promedio en las etapas de entrenamiento y prueba también. Para esto se realizaron 10 entrenamientos de cada una de las redes, de manera que se demostrara que el experimento era repetible y los altos valores de precisión se obtenían por una buena elección en los parámetros de la red. En la Tabla 6.3 se pueden observar los valores promedio para cada una de estas etapas y en con una de las redes utilizadas.

Tabla 6.3: Valores de precisión y error promedio del modelo final para los datos de entrenamiento, los de prueba y los de validación.

Tipo de dato	Precisión[%]	Desviación estándar precisión[%]	Error	Desviación estándar error
Entrenamiento LSTM	94.851	1.488	0.1354	0.0467
Entrenamiento GRU	97.743	0.871	0.0654	0.0133
Prueba LSTM	94.733	1.404	0.1414	0.0493
Prueba GRU	96.765	1.055	0.0877	0.0207
Validación LSTM	92.820	0.1973	0.0125	0.0493
Validación GRU	94.340	1.828	0.1487	0.0465

Algo a destacar de los promedios calculados, es que la red GRU es la que alcanza el valor de precisión más alto de todos, en su etapa de entrenamiento con un 97.74%, además es la red que obtiene en los valores más altos en cada una de las etapas, demostrando que es la red que mejor se desempeña para este proyecto. Esto queda aun más claro al observar la Figura 6.26, donde de una manera mucho más visual se puede notar como en general los datos de precisión obtenidos con las red GRU son más alto que los obtenidos con la red LSTM.

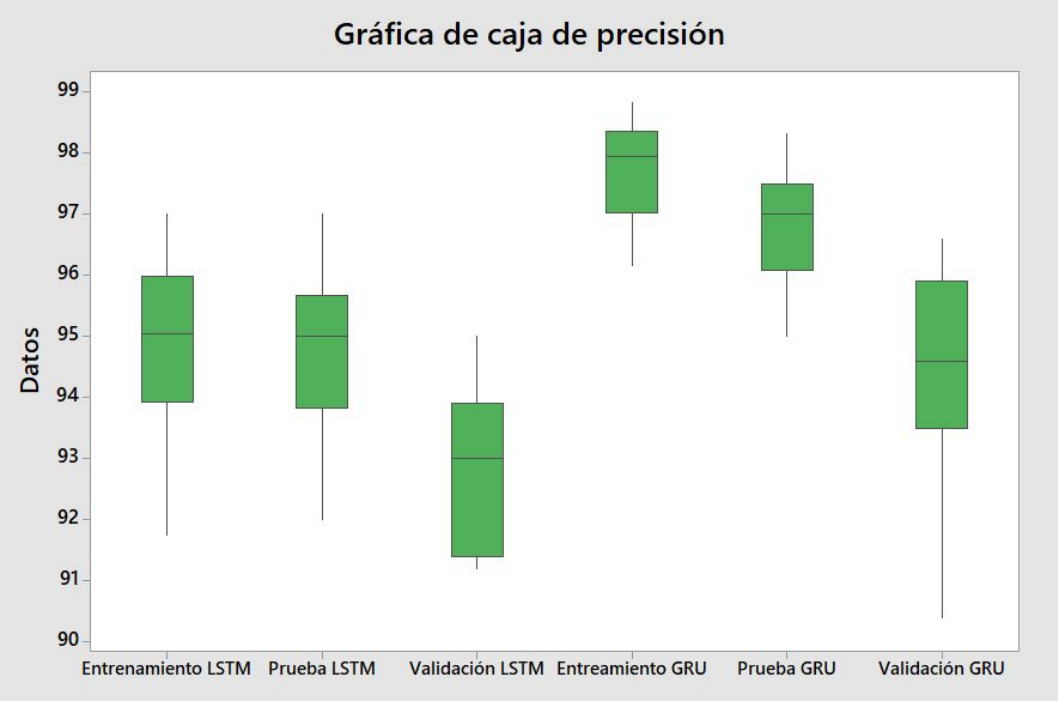


Figura 6.26: Diagrama de caja sobre la distribución de la precisión del modelo.

La precisión reflejada por la gráficas no indica que la red diseñada sea capaz de

detectar todos fallos con ese valor de precisión, este valor solo refleja lo cercanos que son los valores obtenidos en la salida comparado a los esperados, sin embargo, esto se da como un resultado general, que abarca todas las salidas de la red. Por ejemplo, en este problema donde existen diez salidas, una para uno de los fallos, se espera que la red tenga en una de las salidas el valor de 1 y para las otras 9 un valor de 0, si la red tiene como resultados 8 valores cercanos a cero y 2 cercanos a 0.5, aun así resultaría en una alta precisión ya que en acertó casi todos los valores, pero eso no indica que el fallo que este obteniendo la probabilidad más alta sea el que debía ser detectado. Con el objetivo de obtener un valor de precisión más certero y establecer el porcentaje de éxito en detectar cada uno de los tipos de fallos, se realizó una prueba donde el conjunto de validación fue pasado a la red neuronal uno a uno, comprando si el fallo con la mayor probabilidad a la salida era el que se esperaba. En la Tabla 6.4 se muestran los resultados de esta prueba, tanto para la red LSTM como para la GRU, ya que, aunque se estableció que la red GRU era la que trabaja mejor en este problema, es bueno verificar que fallos es capaz de detectar cada una de ellas.

Tabla 6.4: Valores de precisión de los datos de validación para cada una de las categorías de fallo.

Tipo de fallo	Precisión con LSTM[%]	Precisión con GRU[%]
Motor Dañado	100	100
Batería dañada	100	100
Hélice Rota	60	40
Hélice desprendida	100	40
Desconexión de fases o Desmagnetización	0	60
Rodamiento dañado	40	0
Motor bloqueado	100	100
Ruptura de soporte	60	60
Hélice en sentido contrario o Aflojamiento de soporte	20	100
Funcionamiento normal	100	100

En la Figura 6.27 se pueden visualizar mejor los datos de la Tabla 6.4. Inicialmente se puede observar que ambas redes tienen total éxito detectando cuando se esta en

funcionamiento normal, lo cual es muy importante ya que no serviría que la red detecte fallos por error cuando esté operando normal, ya que se podría incurrir en aterrizajes innecesarios. También son capaces de detectar el fallo del motor bloqueado, el cual se considera uno de los más críticos, ya que los motores se dañan con facilidad en presencia de este problema. Adicional a estos dos, ambas redes son capaces de detectar el estado de motor y batería dañada.

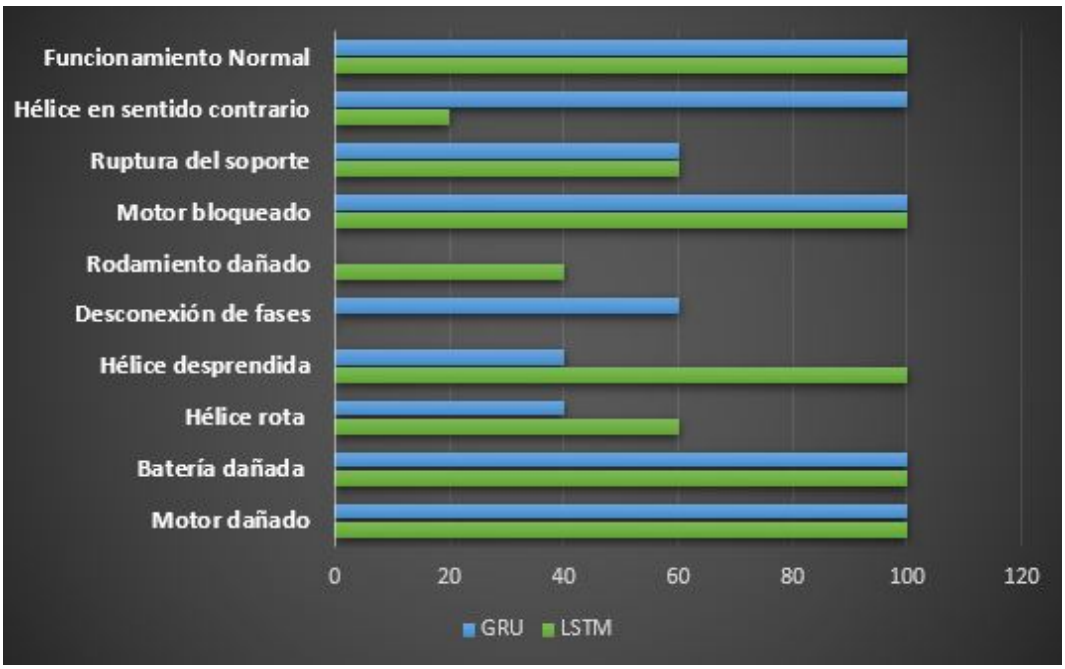


Figura 6.27: Gráfica de barras de la precisión de las redes LSTM y GRU para cada categoría de fallo.

La red GRU destaca en la detección del fallo de colocar una hélice en sentido contrario y tiene una alta precisión también cuando ocurre una desconexión de fases y una ruptura del soporte, sin embargo, es muy baja en la detección de un desprendimiento o rotura de la hélice y es nula en la detección de fallos en los rodamientos. Por otra parte, la LSTM si destaca en la detección de un desprendimiento de la hélice o una rotura de misma, aunque es incapaz de reconocer cuando se presenta una desconexión de fases. A manera de promedio la red GRU obtiene una precisión en la detección de fallos de un 70% y la red LSTM de un 68%, dejando de igual forma a la red GRU

como la logra dar una mejor solución a este tipo de problema. Además, cabe destacar que aunque no logra la detección de fallos en rodamientos, este no es considerado un fallo inmediatamente crítico, ya que la red se entrenó con la idea de detectar cuando se empiezan a notar anomalías, por lo que aunque sería ideal que fuera capaz de detectarlo, es mejor que logre la detección de un fenómeno como la desconexión de fases, el cual si implica la inmediata desconexión de la alimentación del motor.

Una vez que se genera un modelo final se procedió a guardar este en un archivo .json y lo pesos de la red en otro .h5, ambas son extensiones compatibles con Python, aunque también pueden procesarse como archivos .h de manera que sean compatibles con C. Los archivos generados tienen un peso total de 65KB, por lo que esta sería la cantidad memoria mínima necesaria para poder cargarlo a un dispositivo.

Capítulo 7: Análisis económico

A lo largo de este trabajo se han mencionado un gran número de componentes, sensores y piezas que se han adquirido o manufacturado para llevar a cabo este proyecto. Aunque algunos procesos como los de impresión 3D o la confección de los PCB, se realizaron con material y equipo ya disponible se hará un aproximado de los gastos que representan. En la Tabla 7.1 se observa un desglose de los costos aproximados de la implementación del proyecto. La mayoria de los precios se obtuvieron de la página donde se adquirieron como [51] y [52], sin embargo, algunos fueron calculados como aproximados, consultando otras páginas con productos similares, ya que ya se contaba con estos.

Tabla 7.1: Costos aproximados para la implementación del proyecto.

Descripción	Cantidad	Costo (€)
Meses de trabajo del desarrolador del proyecto	5	5000
UAV con mando remoto y PixHawk	1	800
Mini PC Intel NUC7i5BNK	1	380
Impresión 3D	2	15
Arduin UNO	1	18
Sensor HXS 50-NP	1	10.21
Sensor LM35	2	0.4
Acelerómetro MMA8451	1	8.41
Sensor EagleTree RPM	1	13.25
Tornillos y tuercas	6	10
Relé	1	11
Otros	-	20
Suma Neta	-	6286.27

Como se puede observar los gastos más significativos se presentan al inicio de la

Tabla y representan principalmente el costo del UAV y los meses de investigación para el desarrollo del proyecto. Sin embargo, este proyecto busca ser implementable en cualquier vehículo multirrotor, por lo que se podría replicar en vehículo de menor valor, lo cual podría reducir este costo, además el tiempo de investigación se vería drásticamente reducido considerando que ya el conocimiento se ha recopilado con este trabajo.

El tercer costo en importancia es la Mini PC, en este caso se utilizó este dispositivo aprovechando que ya se encontraba montado sobre el vehículo, sin embargo, la red neuronal puede correrse incluso sobre el mismo microcontrolador, o en otras mini computadoras como lo sería un Raspberry Pi, por lo que este gasto se puede reducir también.

Finalmente, el resto de los gastos corresponde a los sensores y equipo para el montaje, donde la categoría de otros abarca cables, resistencias y otros gastos no considerados, necesarios en el desarrollo del proyecto. Por lo que considerando esto, se puede decir que el costo de los componentes completamente necesarios para la aplicación del proyecto en algún otro UAV es de 91.27 euros. Se debe considerar que la aplicación de este proyecto puede prevenir pérdidas económicas sobre el valor de UAV, por daño en motores u otros elementos del vehículo, que en este caso corresponde a un valor de 800 euros, por lo que, aunque este proyecto puede representar una inversión alta, puede prevenir pérdidas mayores que el valor de su implementación.

Capítulo 8: Conclusiones y recomendacio

En este capítulo se resumen las conclusiones alcanzadas con el desarrollo de este proyecto y de igual forma se ofrecen las recomendaciones al cliente para la mejora de este en el desarrollo de un concepto alterno y a mayor escala.

8.1 Conclusiones

Este proyecto permitió el desarrollo de un sistema capaz de la detección y clasificación de fallos, en UAVs con multirrotores, por medio de la implementación de una red neuronal recurrente, la cual opera sobre datos que se recolectan en tiempo real durante el vuelo.

Para poder diseñar el sistema de detección, primero se realizó una categorización y conceptualización de los todos los posibles fallos en el vehículo, al menos en el subsistema de propulsión, que es sobre el que trabaja el proyecto. Al final se establecieron 9 categorías de fallo y una de funcionamiento normal, la conceptualización permitió también escoger las variables en las cuales se reflejan estos fallos, de manera que se pudo escoger sensores adecuados para la toma de datos.

Seguidamente se diseñaron las placas PCB y los elementos de sujeción de estas, que permiten correcta operación del sistema de adquisición de datos, así como la conexión con el mini PC sobre el que corre la red neuronal. Se llevó a cabo un estudio por medio de simulaciones que avalan el concepto de los elementos de sujeción y aseguran su funcionamiento con al menos un factor de seguridad de 1.97, que cumple con el 1.5 establecido en las especificaciones.

Posteriormente se diseñó la red neuronal, encargada de la detección y clasificación de los fallos. Para esto se diseñaron dos modelos, uno con capas ocultas LSTM y otro con capas GRU, ambas redes recurrentes, donde se obtuvieron para las dos resultados de precisión muy altos, sin embargo, la red GRU destaca con un promedio de precisión de 97.7% en los datos de entrenamiento y un 94.3% en los datos de validación. Además, se comprobó sobre el conjunto de validación su capacidad de clasificar correctamente el fallo, donde la red GRU pudo reconocer sin problema 35 muestras de 50, para dar un porcentaje de éxito de 70%.

Considerando el elemento de corrección de fallos del proyecto, se planteó un sistema de corrección y comunicación de fallos, así como el diagrama de flujo que seguiría este proceso, este subsistema quedó pendiente de ser desarrollado.

Finalmente se presentó un análisis económico de los costos de implementación de este proyecto, donde se determinó que es viable y es capaz de prevenir la pérdida o daño de elementos del vehículo, que pueden llegar a sumar gastos mayores que los del costo de su implementación.

8.2 Recomendaciones

- Primeramente, se recomienda probar el sistema propuesto en un vuelo real, de manera que se puede comprobar su nivel de precisión al enfrentarse a datos reales.

- En busca de mejorar la precisión de la herramienta de predicción, se propone recolectar un gran número de datos reales y recolectados por el sistema de adquisición de fallos, provocando cada uno de los fallos propuestos, de manera que se pueda reentrenar la herramienta con estos nuevos datos.

- Como se mencionaba en el capítulo 5, la utilización del Arduino UNO como microcontrolador, se debía principalmente a una indicación del cliente debido a la disponibilidad del mismo y que el proyecto se plantea como una prueba

de concepto, sin embargo una de las recomendaciones para futuros conceptos es utilizar otro microcontrolador, que ofrezca las siguientes mejoras:

  - Una mayor velocidad de reloj, actualmente se el sistema ofrece 16MHz sin embargo la recolección de datos máxima lograda es de 30Hz, por lo que un sistema con un reloj más rápido podría permitir una mejor operación.
  - Que tenga un memoria flash más grande, el Arduino UNO ofrece 32KB de memoria y como se describió en el capitulo 6, el peso del modelo de la red neuronal ronda los 65KB, por lo que no es posible cargarlo actualmente a esta placa, si se deseara correr todo el sistema sobre un solo microcontrolador.
  - Un mayor número de entradas analógicas, actualmente se cuenta con 6, sin embargo este concepto se aplica solo a un motor, por lo que para ampliarlo se necesitarían más o implementar un sistema de multiplexado para la lectura de todos los sensores.
  - Entre los microcontroladores que se podrían utilizar se encuentran : la placa Teensy 4.1, que ofrece casi 40 veces más velocidad que el Arduino UNO así como mayor cantidad de entradas y un modulo MicroSD integrado. Por otra parte se puede utilizar también un Arduino DUE, el cual tiene una muchísima mejor capacidad en todos los aspectos mencionados que el Arduino UNO.

- Se recomienda realizar pruebas de vuelo con el sensor de velocidad y el de corriente y verificar que no se obtiene datos con los errores vistos en el capítulo 6, de presentarse se recomienda probar con un sistema de filtrado, para mejorar la señal o considerar cambiar estos sensores.

- Probar la red con conjuntos de datos de distintos tamaños, actualmente se entrenó para operar con conjuntos de 100 datos, sin embargo, se puede probar el efecto de aumentar o disminuir este valor una vez que se tenga datos reales, considerando

que se podrían obtener buenos resultados con una menor cantidad de datos, lo que aceleraría la toma de decisiones.

- Finalmente se recomienda la implementación del sistema de comunicación con el sistema propuesto por [46], donde se utilizó una puerta de enlace en lugar de los módulos de transmisión 3DR y se demostró que se podía aumentar la distancia para la comunicación con el vehículo, aumentando la potencia del enlace de 100mW a 1W.

Referencias bibliográficas

[1] B. F. Sánchez Pinzón, J. R. Tapia Ortega, and P. Rosa, "Drones : General aspects and social applications", *Visión electrónica*, vol. 10, no. 2, pp. 262–273, 2016.

[2] G. Singhal, B. Bansod, and L. Mathew, "Unmanned Aerial Vehicle classification , Applications and challenges : A Review", *Preprint*, no. November, 2018. DOI: 10.20944/preprints201811.0601.v1.

[3] Grupo Integrado de Ingeniería, *Presentación*. [Online]. Available: http://www.gii.udc.es/presentacion.

[4] P. Misra, G. Kandaswamy, P. Mohapatra, K. Kumar, and P. Balamuralidhar, "Structural health monitoring of multi-rotor micro aerial vehicles", *DroNet 2018 - Proceedings of the 2018 ACM International Conference on Mobile Systems, Applications and Services*, pp. 21–26, 2018. DOI: 10.1145/3213526.3213531.

[5] T. Audronis, *Building Multicopter Video Drones*. Packt Publishing Ltd, 2014.

[6] P. Yedamale, "Brushless dc (bldc) motor fundamentals", *Microchip Technology Inc*, vol. 20, pp. 3–15, 2003.

[7] M. Mikkelsen, "Development , Modelling and Control of a Multirotor Vehicle Written by :", p. 118, 2015.

[8] B. Tefay, B. Eizad, P. Crosthwaite, S. Singh, and A. Postula, "Design of an integrated electronic speed controller for compact robotic vehicles", *Proceedings of the 2011 Australasian Conference on Robotics and Automation*, pp. 7–9, 2011.

[9] E. KUANTAMA, D. CRACIUN, and R. TARCA, "Quadcopter Body Frame Model and Analysis", *ANNALS OF THE ORADEA UNIVERSITY. Fascicle of Management and Technological Engineering.*, vol. Volume XXV (XV), 2016/1, no. 1, 2016, ISSN: 1583-0691. DOI: 10.15660/auofmte.2016-1.3205.

[10] A. Bondyra, P. Ga, S. Gardecki, and A. Kasi, "Development of the Sensory Network for the Vibration-based Fault Detection and Isolation in the Multirotor UAV Propulsion System", vol. 2, no. Icinco, pp. 102–109, 2018. DOI: 10.5220/0006846801020109.

[11] S. Rajagopalan, J. M. Aller, J. A. Restrepo, T. G. Habetler, and R. G. Harley, "Detection of rotor faults in brushless DC motors operating under nonstationary conditions", *IEEE Transactions on Industry Applications*, vol. 42, no. 6, pp. 1464–1477, 2006, ISSN: 00939994. DOI: 10.1109/TIA.2006.882613.

[12] Roger, "A Guide to Understanding LiPo Batteries", *Roger's Hobby Center*, pp. 1–9, 2018. [Online]. Available: https://rogershobbycenter.com/lipoguide.

[13] G. E. G. Jr, C. S. Kulkarni, E. Hogge, A. Hsu, and N. Ownby, "A Study of the Degradation of Electronic Speed Controllers for Brushless DC Motors", 2017.

[14] W. H. Kersting, "Causes and effects of single-phasing induction motors", *IEEE Transactions on Industry Applications*, vol. 41, no. 6, pp. 1499–1505, 2005, ISSN: 00939994. DOI: 10.1109/TIA.2005.857467.

[15] F. Mahmouditabar, A. Vahedi, and P. Ojaghlu, "Investigation of Demagnetization Effect in an Interior", no. March, 2018. DOI: 10.22068/IJEEE.14.1.22.

[16] D. K. Athanasopoulos, P. D. Karagkounis, J. C. Kappatou, M. Ieee, and S. Tsotoulidis, "Demagnetization Faults Analysis in a BLDC Motor for Diagnostic Purposes", *2014 International Conference on Electrical Machines (ICEM)*, pp. 1862–1868, 2014. DOI: 10.1109/ICELMACH.2014.6960437.

[17] A. Verner, "LSTM Networks for Detection and Classification of Anomalies in Raw Sensor Data LSTM Networks for Detection and Classification of Anomalies in Raw Sensor Data by in Computer Science College of Engineering and Computing", no. May, 2019. DOI: 10.13140/RG.2.2.19049.54888.

[18] H. Izakian, "Anomaly Detection in Time Series Data using a Fuzzy C-Means Clustering", pp. 1513–1518, 2013.

[19] Hong Guo, L. B. Jack, and A. K. Nandi, "Feature generation using genetic programming with application to fault classification", *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 35, no. 1, pp. 89–99, 2005.

[20] G. A. Barreto and L. Aguayo, "Time Series Clustering for Anomaly Detection Using Competitive Neural Networks Time Series Clustering for Anomaly Detection Using Competitive Neural Networks", no. May 2014, 2009. DOI: 10.1007/978-3-642-02397-2.

[21] D. T. Shipmon, J. M. Gurevitch, P. M. Piselli, and S. T. Edwards, "Time Series Anomaly Detection; Detection of anomalous drops with limited features and sparse examples in noisy highly periodic data", 2017. arXiv: 1708.03665. [Online]. Available: http://arxiv.org/abs/1708.03665.

[22] S. Calderón Ramírez, "Introducción a las redes neuronales : Redes Recurrentes", *Instituto Tecnológico de Costa Rica*, pp. 1–25, 2019.

[23] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning", 2015. DOI: 10.1038/nature14539.

[24] A. Sherstinsky, "Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network", pp. 1–39, 2018. arXiv: arXiv:1808.03314v1.

[25] C. Olah, *Understanding LSTM Networks*, 2015. [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[26] K. Greff, R. K. Srivastava, J. Koutn, and B. R. Steunebrink, "LSTM : A Search Space Odyssey", pp. 1–12, 2015. DOI: 10.1109/TNNLS.2016.2582924. arXiv: arXiv:1503.04069v2.

[27] J. M. Czum, "Dive Into Deep Learning", *Journal of the American College of Radiology*, 2020, ISSN: 1558349X. DOI: 10.1016/j.jacr.2020.02.005.

[28] S. Khandani, "Engineering Design Process : Education Transfer Plan", no. August, pp. 1–24, 2005. [Online]. Available: http://www.iisme.org/ETPExemplary.cfm.

[29] F. Orjales Saavedra, "Comunicación personal", Grupo Integrado de Ingeniería, Ferrol, Tech. Rep., 2020.

[30] *Ntm prop drive serie 2826 1000 kvr*, 2826, Hobby King. [Online]. Available: https://hobbyking.com/es_es/ntm-prop-drive-28-26-1000kv-235w.html.

[31] *Xrotor40a*, HobbyWing. [Online]. Available: https://www.hobbywingdirect.com/collections/xrotor-user-manual.

[32] *Arduino uno*, Arduino Uno Rev3, ARDUINO. [Online]. Available: https://store.arduino.cc/usa/arduino-uno-rev3.

[33] *Current transducer hxs 50-np*, 50-NP, LEM, 2014. [Online]. Available: https://www.lem.com/en/hxs-50np.

[34] *Brushless motor rpm sensor v2*, Eagle Tree Systems, 2011. [Online]. Available: https://www.eagletreesystems.com/index.php?route=product/product&product_id=64.

[35] *Lm35 precision centigrade temperature sensor*, LM35DZ, Texas Instruments, 1999. [Online]. Available: https://www.ti.com/lit/gpn/lm35.

[36] *Mma8451q 3-axis,14-bit/8-bit digital accelerometer*, Freescale Semiconductor, 2013. [Online]. Available: http://www.adafruit.com/datasheets/MMA8451Q-1.pdf.

[37] IPC-2221, "Generic Standard on Printed Board Design", *Ipc*, no. February, 1998.

[38] A. Alberto Relaño Pastor Director and J. Hidalgo García, "UNIVERSIDAD CARLOS III DE MADRID ESCUELA POLITÉCNICA SUPERIOR DEPARTAMENTO DE CIENCIA E INGENIERÍA DE LOS MATERIALES E INGENIERÍA QUÍMICA PROYECTO FIN DE CARRERA INGENIERÍA INDUSTRIAL Estudio comparativo de piezas de ABS y PLA procesadas mediante modelado por deposición fundida", 2013. [Online]. Available: https://e-archivo.uc3m.es/bitstream/handle/10016/18015/PFC%7B%5C_%7DAntonio%7B%5C_%7DRelano%7B%5C_%7DPastor.pdf?sequence=1.

[39] B. M. Tymrak, M. Kreiger, and J. M. Pearce, "Mechanical properties of components fabricated with open-source 3-D printers under realistic environmental conditions", *Materials and Design*, vol. 58, pp. 242–246, 2014, ISSN: 18734197. DOI: 10.1016/j.matdes.2014.02.038. [Online]. Available: http://dx.doi.org/10.1016/j.matdes.2014.02.038.

[40] J. Bourabah, *PLA vs ABS: comparación de filamentos para impresión 3D*, 2020. [Online]. Available: https://all3dp.com/es/1/pla-vs-abs-comparacion-impresion-3d/ (visited on).

[41] A. International and S. Specification, "Standard Specification for Carbon and Alloy Steel Externally Threaded Metric", vol. 01, no. February 1999, pp. 1–8, 2002.

[42] *Ssr-40 dd*, FOTEK. [Online]. Available: https://cdn.sparkfun.com/datasheets/Components/General/SSR40DA.pdf.

[43] *Intel® nuc nuc7i5bnk*, Intel. [Online]. Available: https://ark.intel.com/content/www/es/es/ark/products/95061/intel-nuc-kit-nuc7i5bnk.html.

[44] *Cube flight controller*, NEX. [Online]. Available: https://ardupilot.org/copter/docs/common-thecube-overview.html.

[45] *V2.0 3dr radio telemetry 433mhz 915mhz data transmission module*, 3DR Radio. [Online]. Available: https://www.banggood.com/New-Upgraded-V2_0-3DR-Radio-Telemetry-433MHZ-915MHZ-Data-Transmission-Module-For-Android-Smartphone-APM-Pixhawk-PX4-p-1304758.html?ID=510651&cur_warehouse=CN#jsReviewsWrap.

[46] P. A. A. Castillo, "Diseño y desarrollo de una estructura de software-hardware para el hospedaje de rutinas de comportamiento autónomo en vehículos aéreos no tripulados Informe", PhD thesis, Instituto Tecnológio de Costa Rica, 2019, p. 93.

[47] A. Pilot, *MAVLink*. [Online]. Available: https://ardupilot.org/dev/docs/mavlink-basics.html.

[48] ——, *APM Planner 2*. [Online]. Available: https://ardupilot.org/planner/.

[49] Autodesk, *Inventor Professional 2018*. [Online]. Available: https://www.autodesk.com/products/inventor/overview.

[50] K. J. BUDYNAS, RICHARD G. NISBETT, "Diseño en Ingeniería Mecánica de Shigley", *Igarss 2014*, no. 1, pp. 1–5, 2014, ISSN: 0717-6163. DOI: 10.1007/s13398-014-0173-7.2. arXiv: arXiv:1011.1669v3.

[51] RS, *RS Componentes*. [Online]. Available: https://es.rs-online.com.

[52] HobbyKing, *HobbyKing.com*. [Online]. Available: https://hobbyking.com/.

Anexos

9.1 Anexo 1: Algoritmo para la recolección de datos

```
1
2 //Proyecto Final de Graduacion
3 //Dilan Andrey Loria Quesada
4 //Algoritmo para la recoleccion de datos
5
6 //Declaracion de Librerias
7 #include <SPI.h>
8 #include "SdFat.h"
9 #include <FreqMeasure.h>
10 #include <Wire.h>
11 #include <Adafruit_MMA8451.h>
12 #include <Adafruit_Sensor.h>
13
14
15 //Declaracion de Variables y Pines
16
17 SdFat SD;
18 #define SD_CS_PIN 9
19 File file;
20 bool FLAG;
21 unsigned long time;
22 float ID =0;
23 int cuenta = 0,1cuenta,1cuenta,1cuenta,1cuenta2=0;
24 float TempM;
25 float TempB;
26 float Corriente,1Corriente2;
27 float SumaCorriente;
28 float Tension;
29 float Suma=0;
30 int Cuenta=0;
31 float RPM=0;
32 float RPM2=0;
33 float Ax=0,1Ay=0,1Az=0;
34
35 float ADatos[5][8];
36 float SDArray[5][8];
37
38 Adafruit_MMA8451 mma = Adafruit_MMA8451();
39
40
```

```
41
42 void setup() {
43
44
45
46 Serial.begin(2000000);
47
48 //Inicializacion del medidor de frecuencia
49 FreqMeasure.begin();
50
51 //Inicializacion del acelerometro
52 mma.begin();
53 mma.setRange(MMA8451_RANGE_4_G);
54
55 //Inicializacion de la sensor SD
56
57 if (!SD.begin(SD_CS_PIN)) { // IInicializar SD
58     Serial.println("No se pudo inicializar.");
59   }
60
61   if (SD.exists("FILE.txt")) {
62     Serial.println("El archivo existe.");
63     if (SD.remove("FILE.txt") == true) {
64       Serial.println("Se elimino el archivo.");
65     } else {
66       Serial.println("No se pudo eliminar el archivo.");
67     }
68     SD.remove("FILE.txt");
69   }else{
70     Serial.println("Archivo revisado");
71     }
72   file = SD.open("FILE.txt",1FILE_WRITE);
73   if (file){
74     file.close();
75     Serial.println("Archivo inicializado");
76     }
77    else{
78     Serial.println("No se pudo inicializar el archivo");
79     }
80 file = SD.open("FILE.txt",1FILE_WRITE);
81       if (file) {
82 file.println("ID,Tension,Corriente,TempM,TempB,Ax,Ay,Az,RPM');}
83 file.close();
84
85
86 //Inicializacion de la interrupcion del TIMER 2
87
88 SREG = (SREG & 0b01111111);
89 TCNT2 = 0;
90 TIMSK2 = TIMSK2|0b00000001;
91 TCCR2B = 0B00000111;
92 SREG = (SREG & 0b01111111) | 0b10000000;
93
94
```

```
95 }
96
97 //Funcion del la Interrupcion del TIMER2 cuando se desborda
98
99 ISR(TIMER2_OVF_vect){
100     FLAG=HIGH;
101
102 }
103
104
105 //Inicia el ciclo
106
107 void loop() {
108
109
110 //Se realiza le toma de datos solo si hubo interrupcion
111
112 if (FLAG==HIGH) {
113
114 //Lectura de Tension
115 Tension = analogRead(1);
116 Tension= (Tension/1024)*5;
117 Tension= (Tension*14.3)/4.6;
118
119
120 //Lectura de Temperatura Motor
121 TempM = analogRead(3);
122 TempM = (5.0 * TempM * 100.0)/1024.0;
123
124
125
126 //Lectura de Temperatura Bateria
127 TempB = analogRead(0);
128 TempB = (5.0 * TempB * 100.0)/1024.0;
129
130
131
132 //Lectura de Corriente
133 Corriente = analogRead(2);
134 Corriente = (Corriente/1024.0)*5.0 ;
135 Corriente= 77.444*Corriente - 205.55;
136
137
138
139 //Lectura de Vibracion
140 mma.read();
141 sensors_event_t event;
142 mma.getEvent(&event);
143 Ax = event.acceleration.x;
144 Ay = event.acceleration.y;
145 Az = event.acceleration.z;
146
147
148 //Se almacenan los datos en un Array
```

```
149
150 ADatos[cuenta][0]= Tension;
151 ADatos[cuenta][1]= Corriente2;
152 ADatos[cuenta][2]= TempM;
153 ADatos[cuenta][3]= TempB;
154 ADatos[cuenta][4]= Ax;
155 ADatos[cuenta][5]= Ay;
156 ADatos[cuenta][6]= Az;
157 ADatos[cuenta][7]= RPM;
158
159
160 cuenta++;
161 cuenta2++;
162
163 //Si el Array ADatos ya tiene 5 datos, se crea una copia en otro array
164
165 if (cuenta==5){
166 SREG = (SREG & 0b01111111);
167 memcpy(SDArray,1ADatos,1sizeof SDArray);
168 cuenta=0;
169
170 SREG = (SREG & 0b01111111) | 0b10000000;
171   }
172
173 FLAG=LOW;
174 }
175
176
177 //Lectura de Velocidad(Lectura en Pin 8)
178 if (FreqMeasure.available()) {
179 Suma = Suma + FreqMeasure.read();
180 SumaCorriente = SumaCorriente + Corriente;
181 Cuenta = Cuenta + 1;
182
183 if (Cuenta == 15) {
184    float frequency = FreqMeasure.countToFrequency(Suma/Cuenta);
185   RPM2=(frequency*60)/6;
186    if (RPM2>5500){
187        RPM2=RPM2/3;
188        }
189     if (abs(RPM2-RPM)>1000){
190         float RPM3=(RPM2+RPM)/2.1;
191         if (abs(RPM3-RPM)>800){
192             float RPM4=(RPM3+RPM)/2;
193             if (abs(RPM4-RPM)>500){
194                 float RPM5=(RPM4+RPM)/2;
195                  if (abs(RPM5-RPM)>500){
196                 RPM=(RPM5+RPM)/2; }}}
197   } else {RPM=RPM2;}
198
199   Corriente2=SumaCorriente/Cuenta;
200   Suma = 0;
201   SumaCorriente=0;
202   Cuenta = 0;
```

```
203
204 }}
205
206 //El Array copiado es alamcenado en la SD
207
208 if (cuenta2==5){
209
210 file = SD.open("FILE.txt",1FILE_WRITE);
211 for(int i = 0; i < 5; i++) {
212
213       ID++;
214          if (file) {
215          file.print(ID);
216          file.print(",");
217          file.print(SDArray[i][0]);
218          file.print(",");
219          file.print(SDArray[i][1]);
220          file.print(",");
221          file.print(SDArray[i][2]);
222          file.print(",");
223          file.print(SDArray[i][3]);
224          file.print(",");
225          file.print(SDArray[i][4]);
226          file.print(",");
227          file.print(SDArray[i][5]);
228          file.print(",");
229          file.print(SDArray[i][6]);
230          file.print(",");
231          file.println(SDArray[i][7]);}
232   }
233   file.close();
234   cuenta2=0;
235   }
236 }
```

Algoritmo 9.1: Código para la recolección de datos implementado en Arduino

9.2 Anexo 2: Diagrama de los circuitos y PCB real

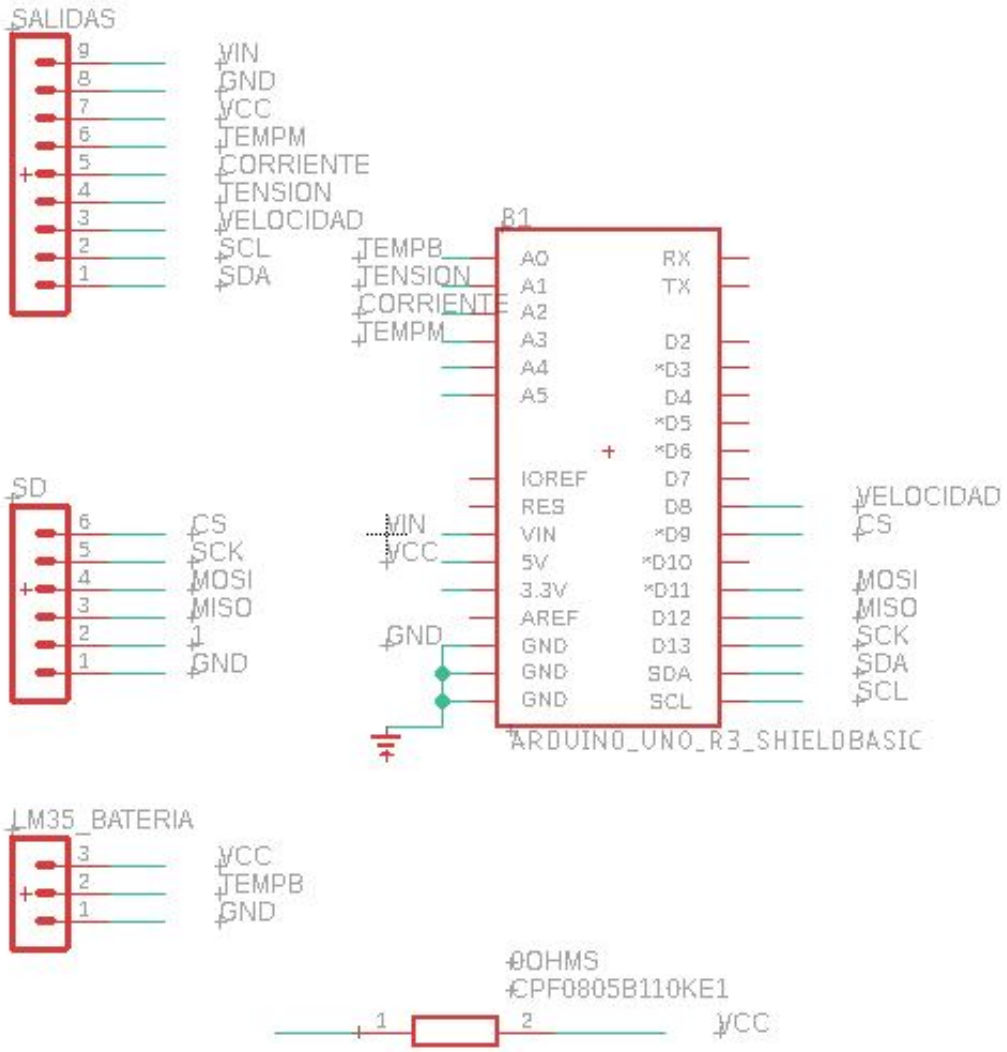


Figura 9.1: Circuito del PCB de la carcasa central.

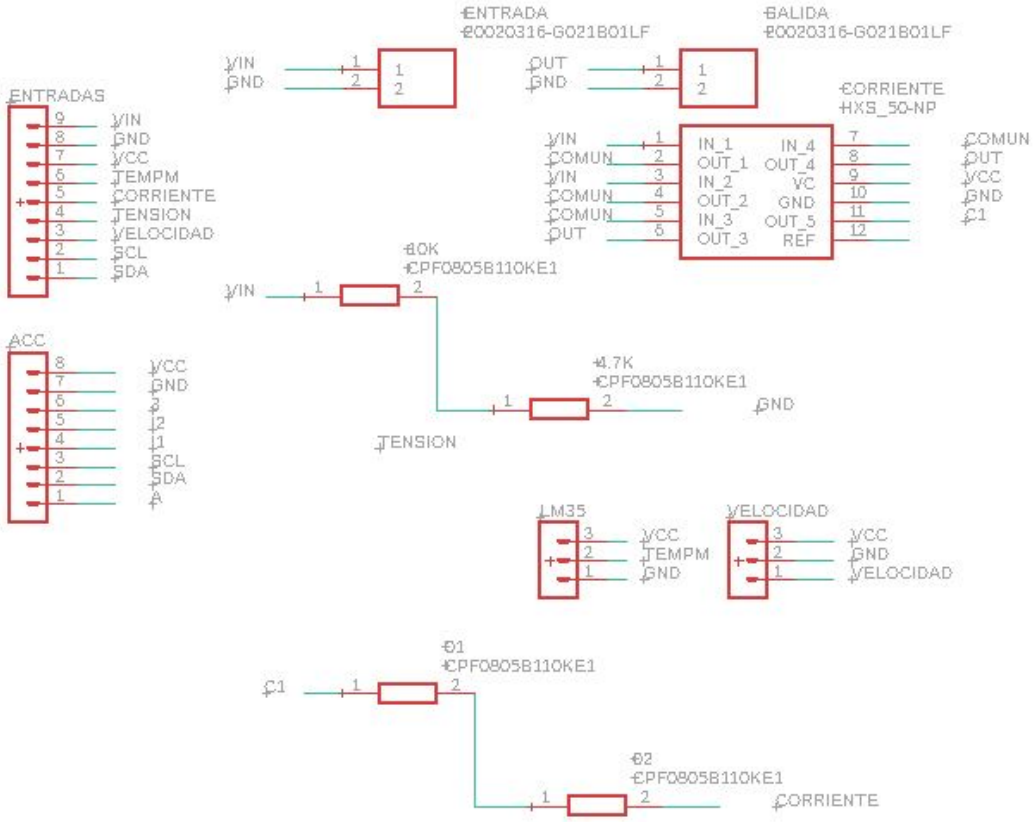


Figura 9.2: Circuito del PCB de los sensores.

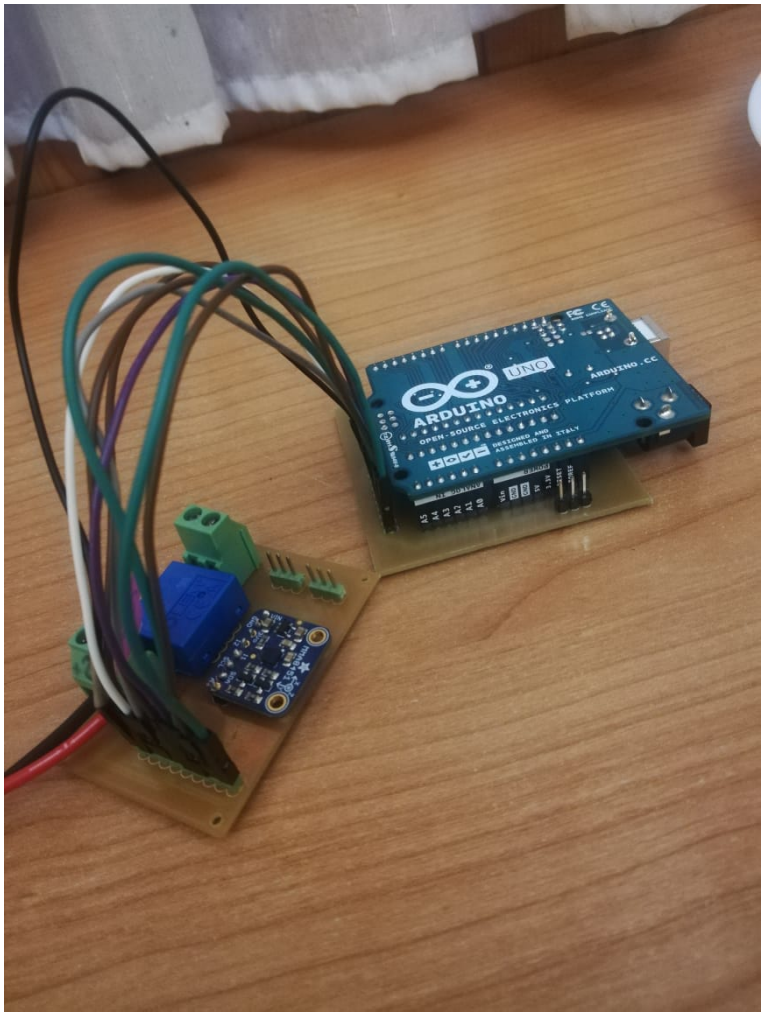
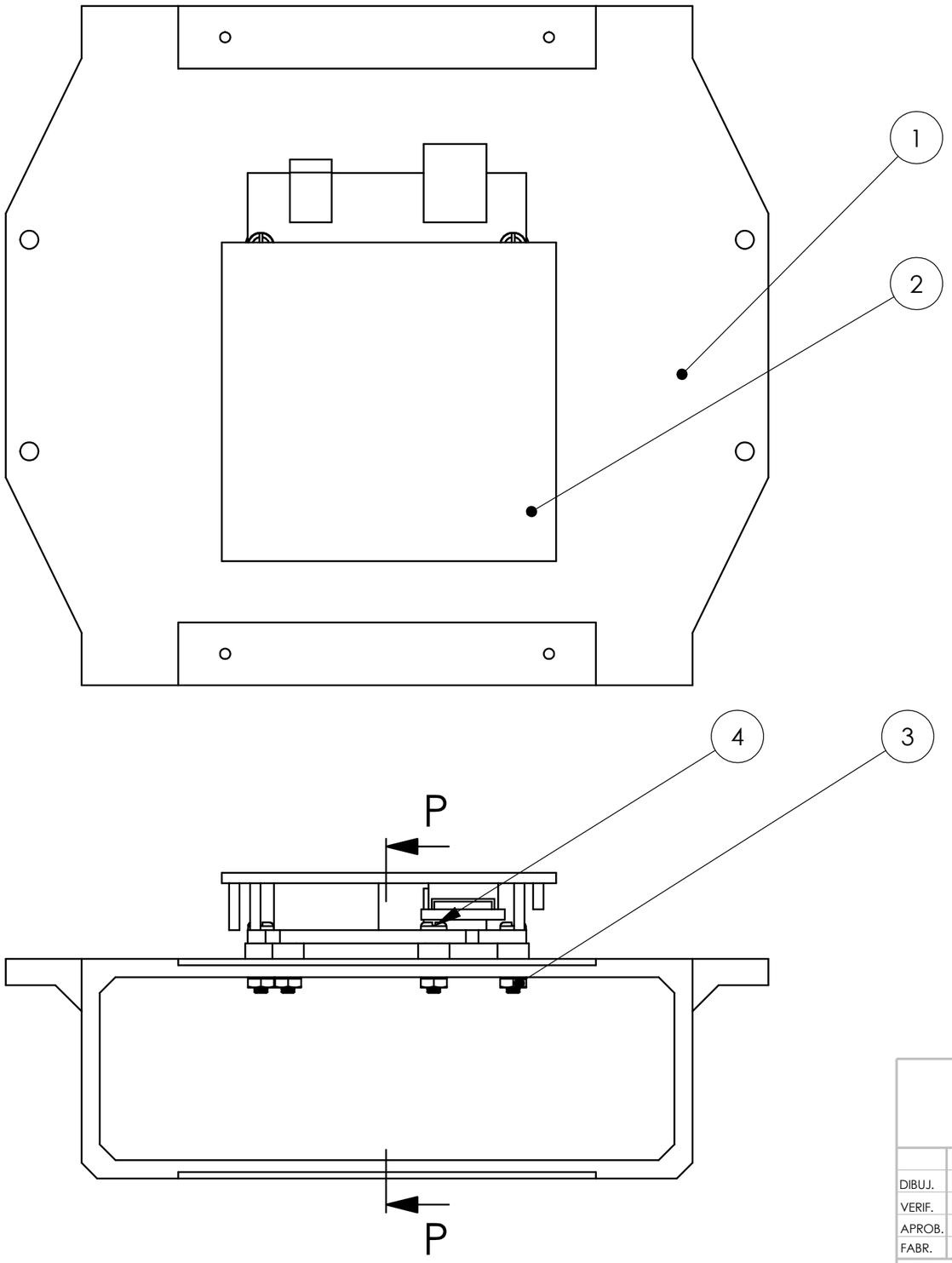
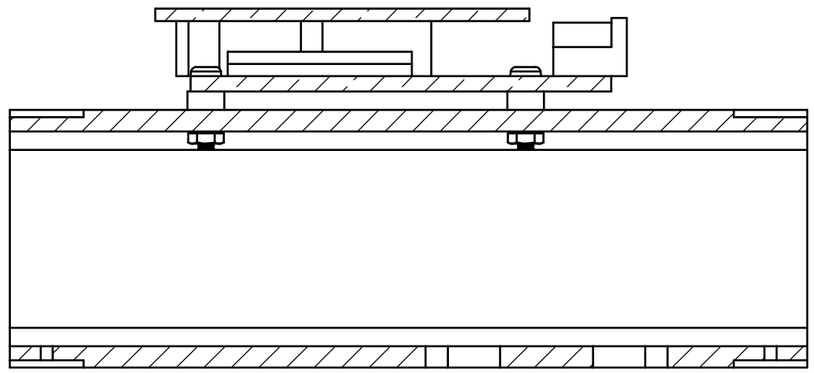


Figura 9.3: PCBs manufacturados.

9.3 Anexo 3: Planos de los elementos de sujeción del sistema de adquisición de datos.



Pieza	Designación	Denominación	Descripción	Cantidad
	DAL-001-000	Ensamble General 1		
	DAL-001-001	Dibujo Explosionado 1		
1	DAL-001-002	Carcasa		1
2		PCB	Representación simbólica	1
3		Tuerca M2.5	Hexagon Nut ISO 4032 - M2.5 - W - N	4
4		Tornillo M2.5	ISO 1580 - M2.5 x 12 - 12S	4



Corte P-P

Escala 1 : 1

TEC | Tecnológico de Costa Rica

ÁreaAcadémica de Ingeniería Mecatrónica

CURSO: Proyecto de Graduación

REVISIÓN: 1

	NOMBRE	FIRMA	FECHA
DIBUJ.	D.Loría		25/6/20
VERIF.	D.Loría		
APROB.			
FABR.			

ACABADO GENERAL: 12

MATERIAL: --

DENOMINACIÓN: Ensamble General 1

SISTEMA: FIRST ANGLE PROJECTION

TOLERANCIAS: LINEAL: X ± 0.5, .X ± 0.3, .XX ± 0.05; ANGULAR: ±° 1

TOLERANCIA EN: mm

DESIGNACIÓN: DAL-001-000

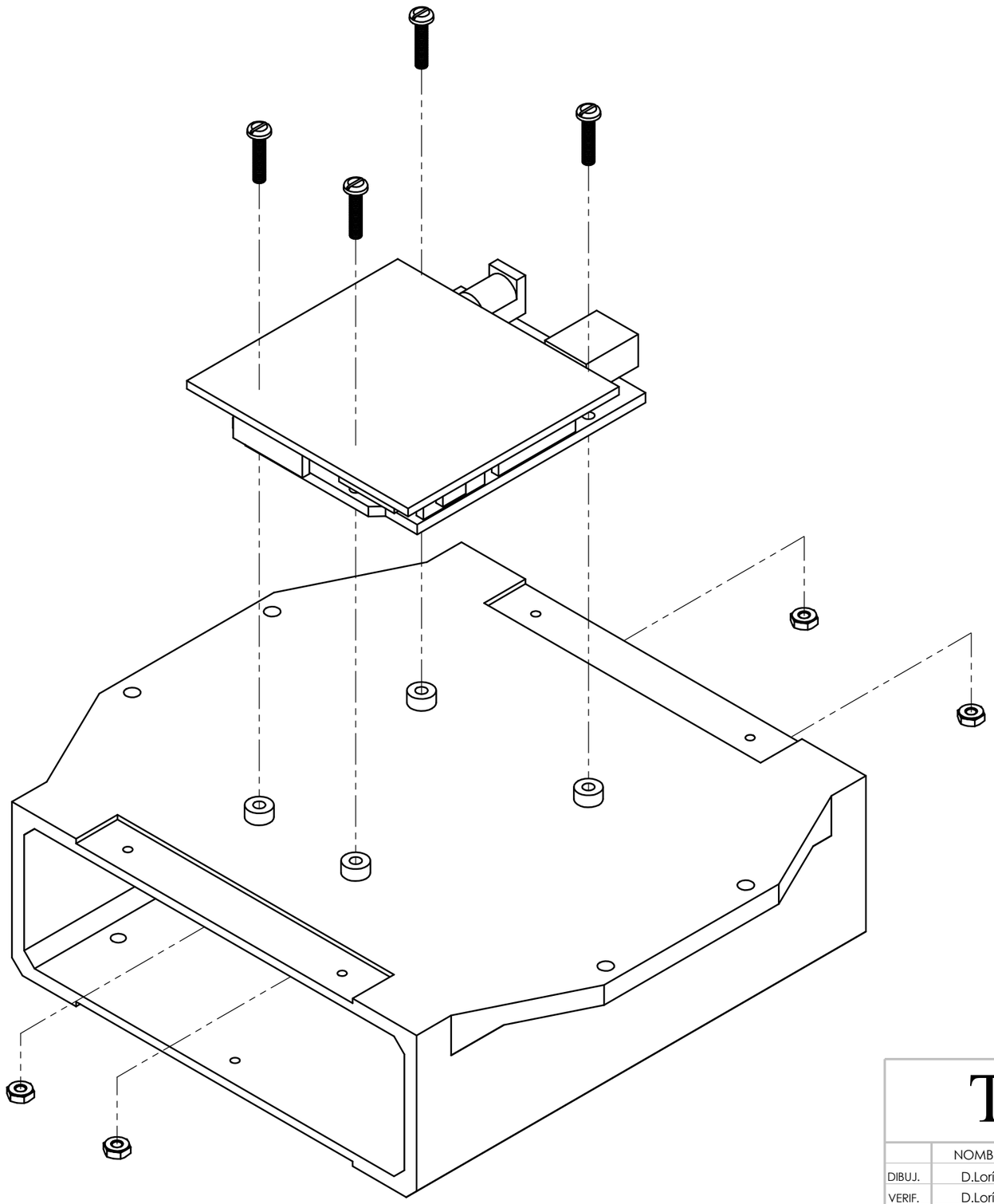
ESCALA: 1:1

FORMATO: A3

PESO: -- gramos

UNIDADES: mm

HOJA 1 DE 7



Pieza	Designación	Denominación	Descripción	Cantidad
	DAL-001-000	Ensamble General 1		
	DAL-001-001	Dibujo Explosionado 1		
1	DAL-001-002	Carcasa		1
2		PCB	Representación simbólica	1
3		Tuerca M2.5	Hexagon Nut ISO 4032 - M2.5 - W - N	4
4		Tornillo M2.5	ISO 1580 - M2.5 x 12 - 12S	4

TEC | Tecnológico de Costa Rica

ÁreaAcadémica de Ingeniería Mecatrónica

CURSO: Proyecto de Graduación

REVISIÓN: 1

	NOMBRE	FIRMA	FECHA
DIBUJ.	D.Loría		25/6/20
VERIF.	D.Loría		
APROB.			
FABR.			

ACABADO GENERAL: 12

MATERIAL: --

DENOMINACIÓN: Dibujo explosionado 1

SISTEMA: FIRST ANGLE PROJECTION

TOLERANCIAS: LINEAL: X ± 0.5, .X ± 0.3, .XX ± 0.05; ANGULAR: ±° 1

TOLERANCIA EN: mm

DESIGNACIÓN: DAL-001-001

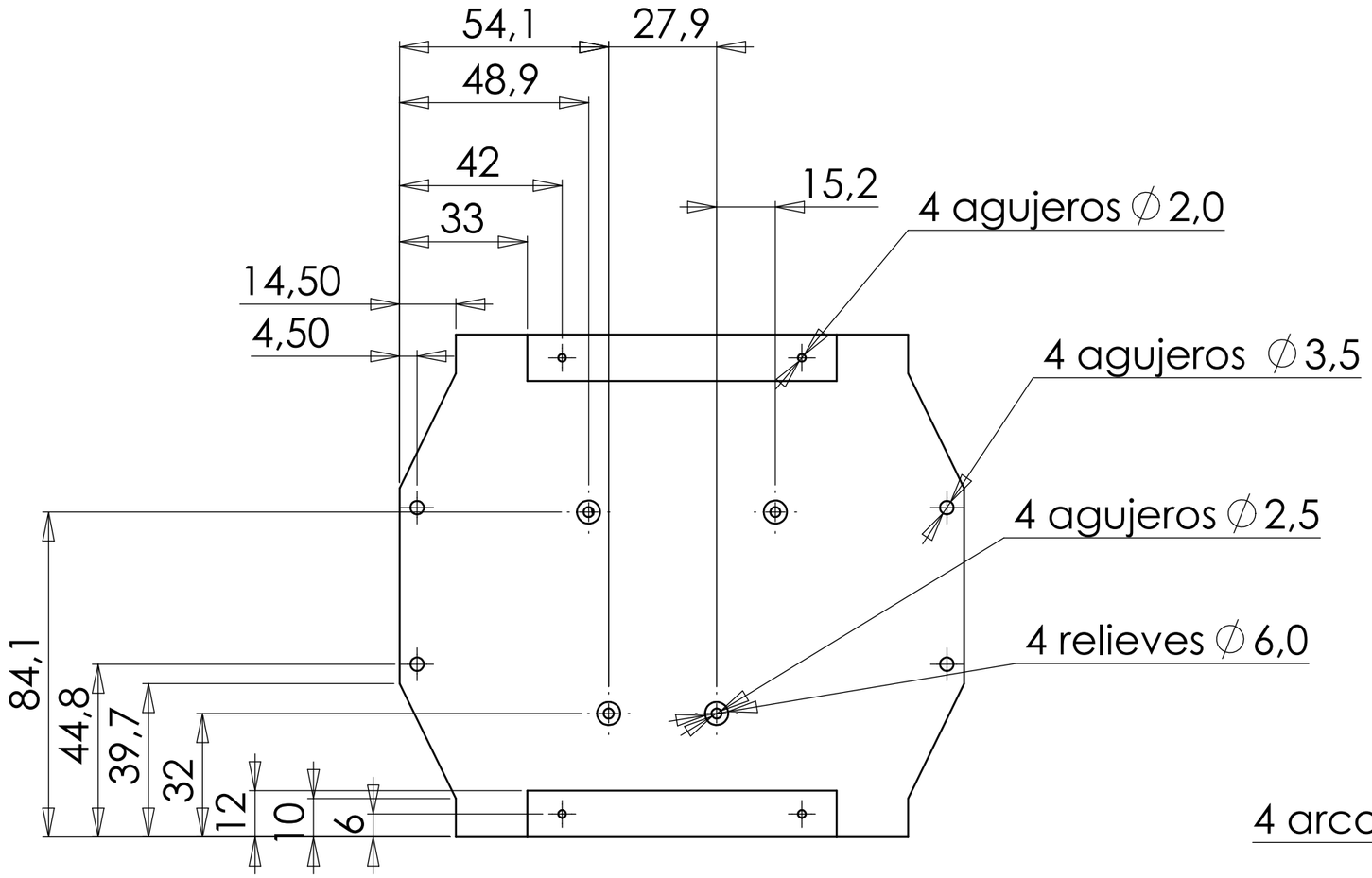
ESCALA: 1:1

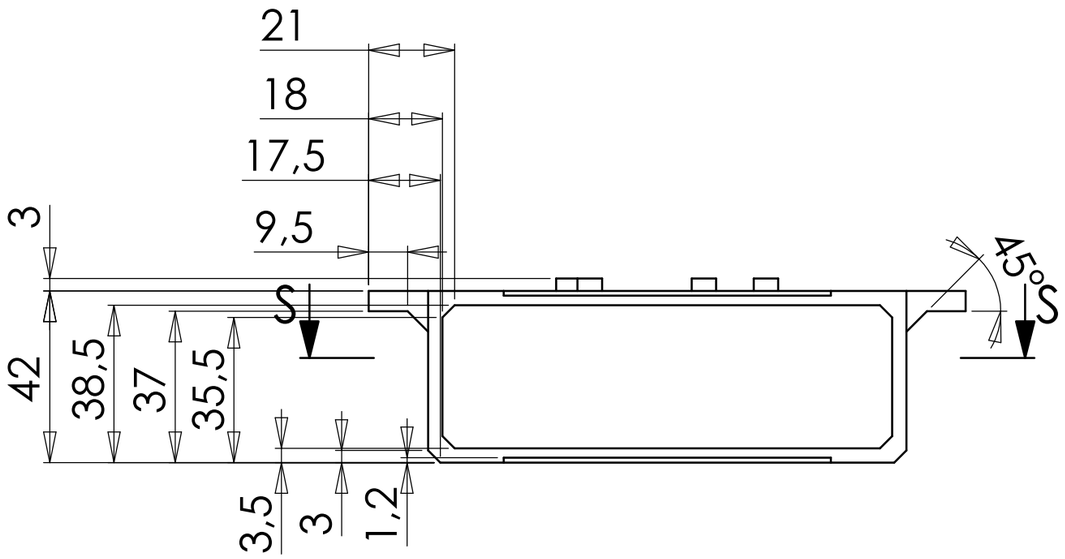
FORMATO: A3

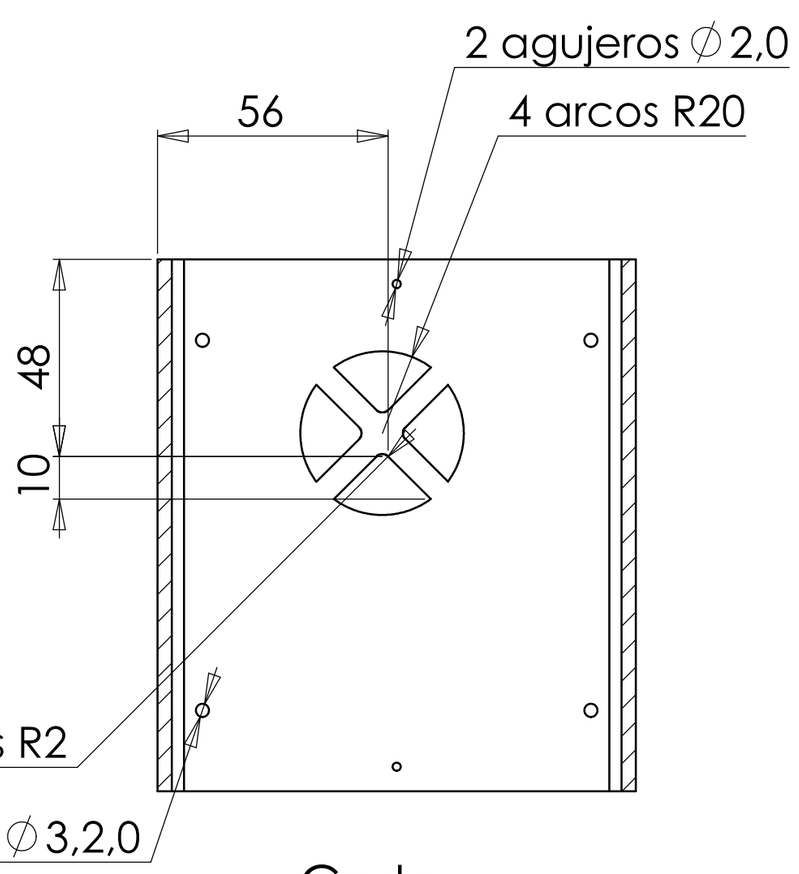
PESO: -- gramos

UNIDADES: mm

HOJA 2 DE 7







Corte S-S

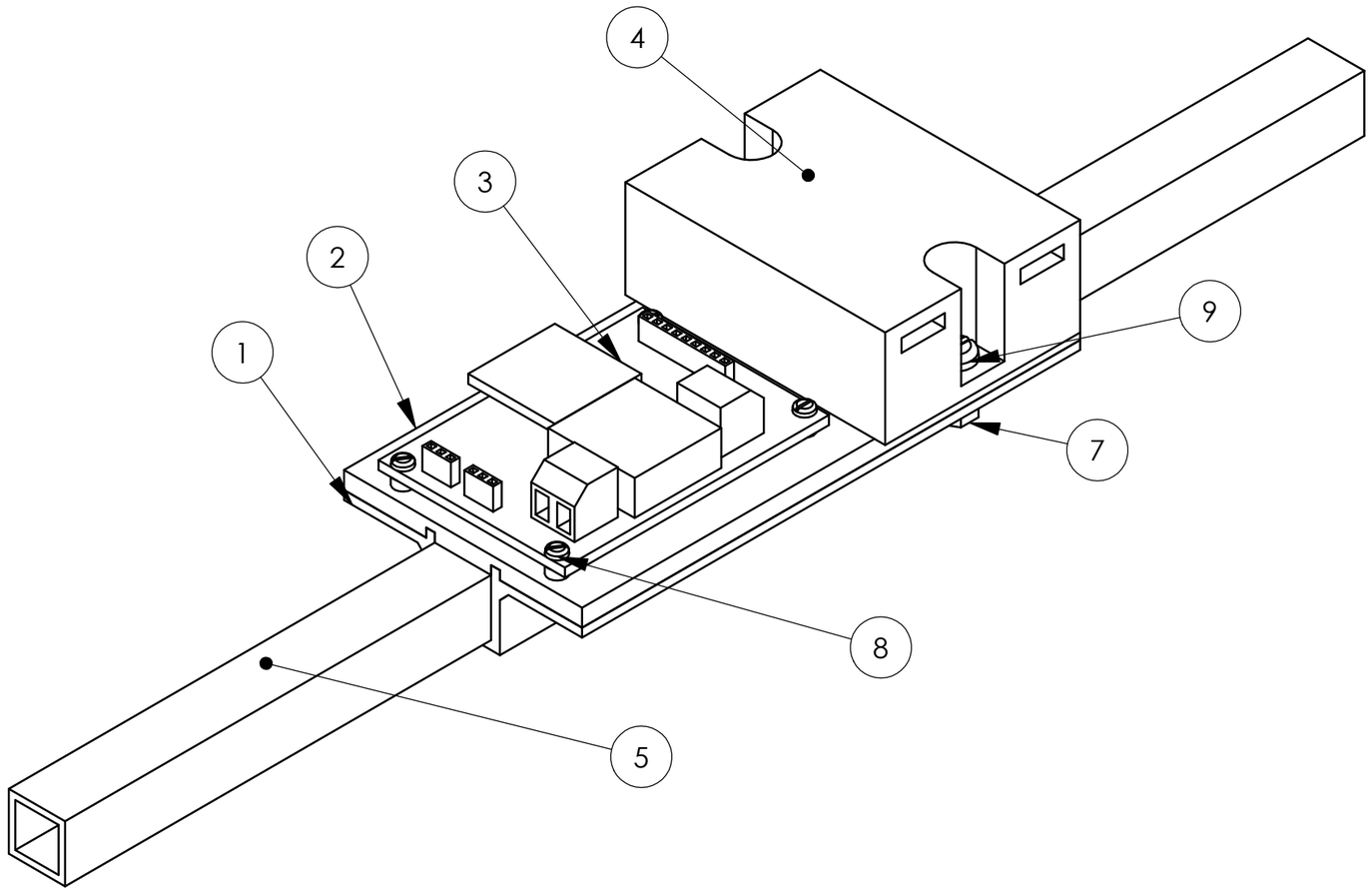
Escala 1 : 1.5

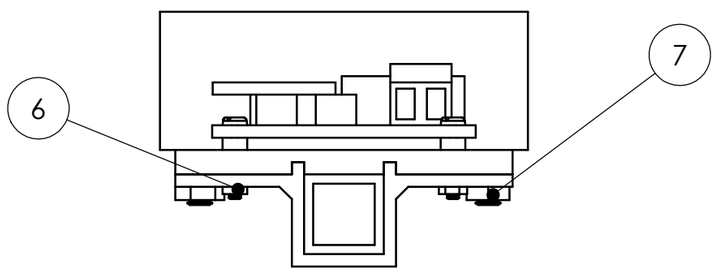
Las cotas cuya tolerancia no se encuentren en el plano, se rigen por la información de tolerancias del cajetín.

TEC | Tecnológico de Costa Rica		ÁreaAcadémica de Ingeniería Mecatrónica	REVISIÓN: 1
		CURSO: Proyecto de Graduación	

	NOMBRE	FIRMA	FECHA
DIBUJ.	D.Loría		25/6/20
VERIF.	D.Loría		
APROB.			
FABR.			

ACABADO GENERAL: 12	DENOMINACIÓN: Carcasa		
MATERIAL: ABS			
SISTEMA: FIRST ANGLE PROJECTION	TOLERANCIAS: LINEAL: X ± 0.5, .X ± 0.3, .XX ± 0.05; ANGULAR: ± 1°	TOLERANCIA EN: mm	DESIGNACIÓN: DAL-001-002
		PESO: -- gramos	UNIDADES: mm
ESCALA: 1:1,5	FORMATO A3	HOJA 3 DE 7	





Pieza	Designación	Denominación	Descripción	Cantidad
	DAL-002-000	Ensamble General 2		
	DAL-002-001	Dibujo explosionado 2		
1	DAL-002-002	Soporte		1
2	DAL-002-003	Tapa		1
3		PCB	Representación simbólica	1
4		Relé	Representación simbólica	1
5		Brazo	Representación simbólica	1
6		Tuerca M2,5	ISO - 4036 - M2,5 - S	4
7		Tuerca M4	ISO - 4036 - M4 - S	2
8		Tornillo M2,5	ISO 1580 - M2,5 x 12 - 12S	4
9		Tornillo M4	ISO 1580 - M4 x 12 - 12S	2

TEC | Tecnológico de Costa Rica

ÁreaAcadémica de Ingeniería Mecatrónica

CURSO: Proyecto de Graduación

REVISIÓN: 1

	NOMBRE	FIRMA	FECHA
DIBUJ.	D.Loría		25/6/20
VERIF.	D.Loría		
APROB.			
FABR.			

ACABADO GENERAL: 12

MATERIAL: --

DENOMINACIÓN: Ensamble General 2

SISTEMA: FIRST ANGLE PROJECTION

TOLERANCIAS: LINEAL: X ± 0.5, .X ± 0.3, .XX ± 0.05; ANGULAR: ± 1°

TOLERANCIA EN: mm

DESIGNACIÓN: DAL-002-000

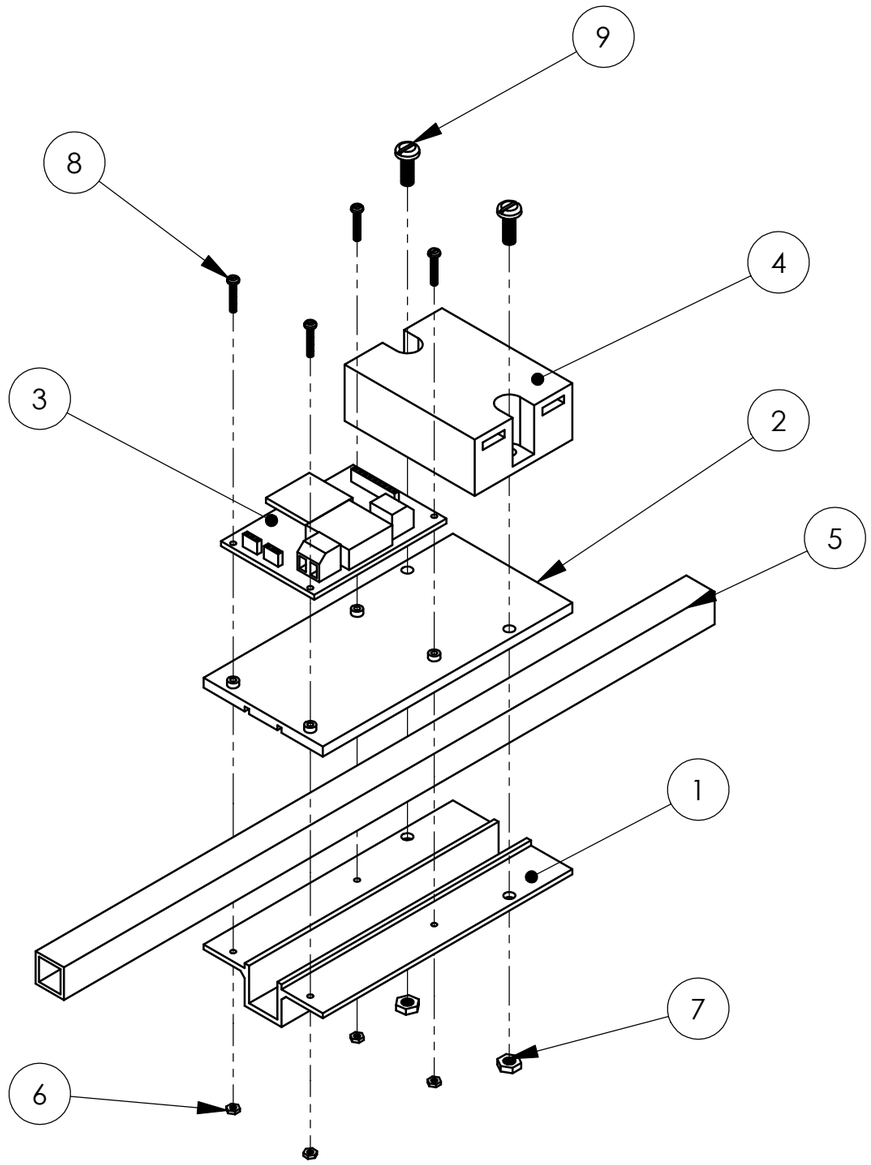
ESCALA: 1:1

FORMATO: A3

PESO: -- gramos

UNIDADES: mm

HOJA 4 DE 7



Pieza	Designación	Denominación	Descripción	Cantidad
	DAL-002-000	Ensamble General 2		
	DAL-002-001	Dibujo explosionado 2		
1	DAL-002-002	Soporte		1
2	DAL-002-003	Tapa		1
3		PCB	Representación simbólica	1
4		Relé	Representación simbólica	1
5		Brazo	Representación simbólica	1
6		Tuerca M2,5	ISO - 4036 - M2,5 - S	4
7		Tuerca M4	ISO - 4036 - M4 - S	2
8		Tornillo M2,5	ISO 1580 - M2,5 x 12 - 12S	4
9		Tornillo M4	ISO 1580 - M4 x 12 - 12S	2

TEC | Tecnológico de Costa Rica

ÁreaAcadémica de Ingeniería Mecatrónica

CURSO: Proyecto de Graduación

REVISIÓN: 1

	NOMBRE	FIRMA	FECHA
DIBUJ.	D.Loría		25/6/20
VERIF.	D.Loría		
APROB.			
FABR.			

ACABADO GENERAL: 12

MATERIAL: --

DENOMINACIÓN: Dibujo explosionado 2

SISTEMA: FIRST ANGLE PROJECTION

TOLERANCIAS: LINEAL: X ± 0.5, .X ± 0.3, .XX ± 0.05; ANGULAR: ±° 1

TOLERANCIA EN: mm

DESIGNACIÓN: DAL-002-001

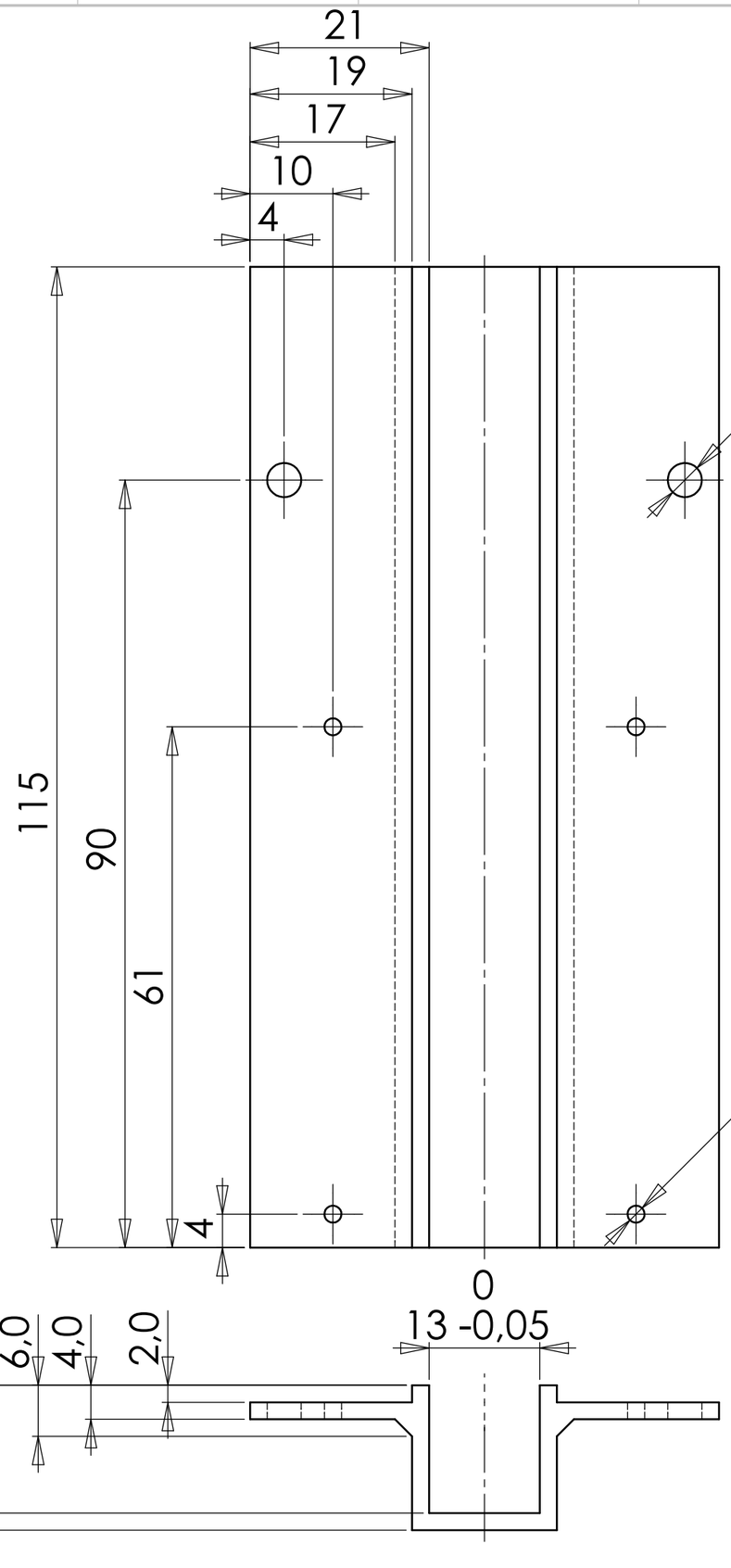
ESCALA: 1:2

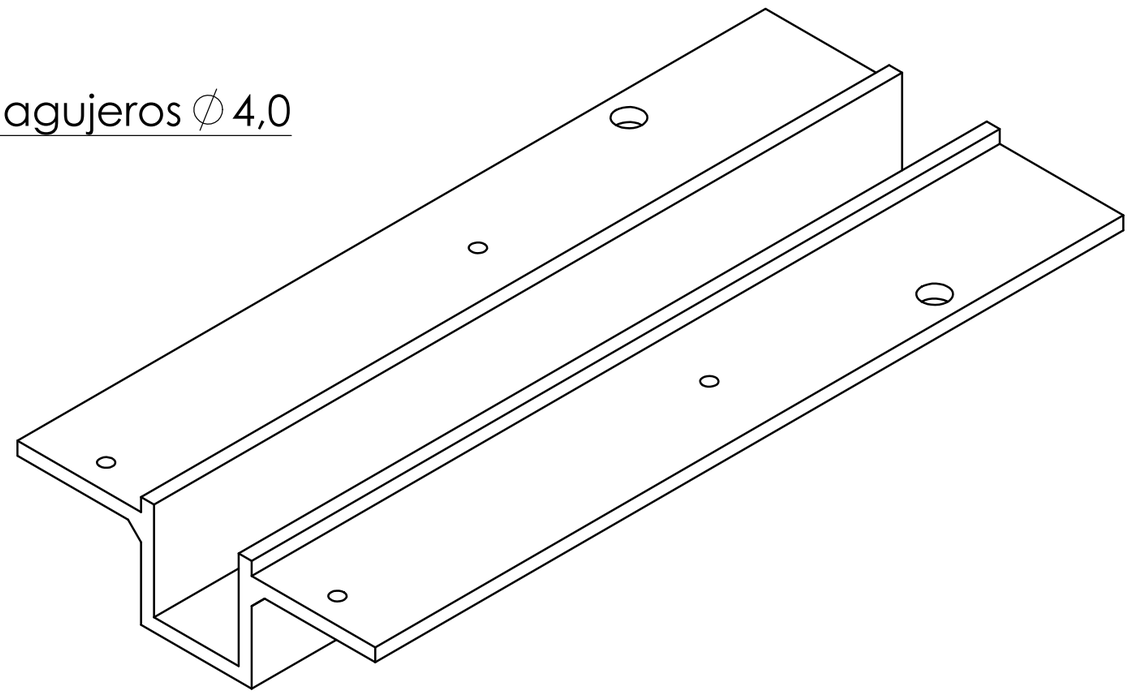
FORMATO: A3

PESO: -- gramos

UNIDADES: mm

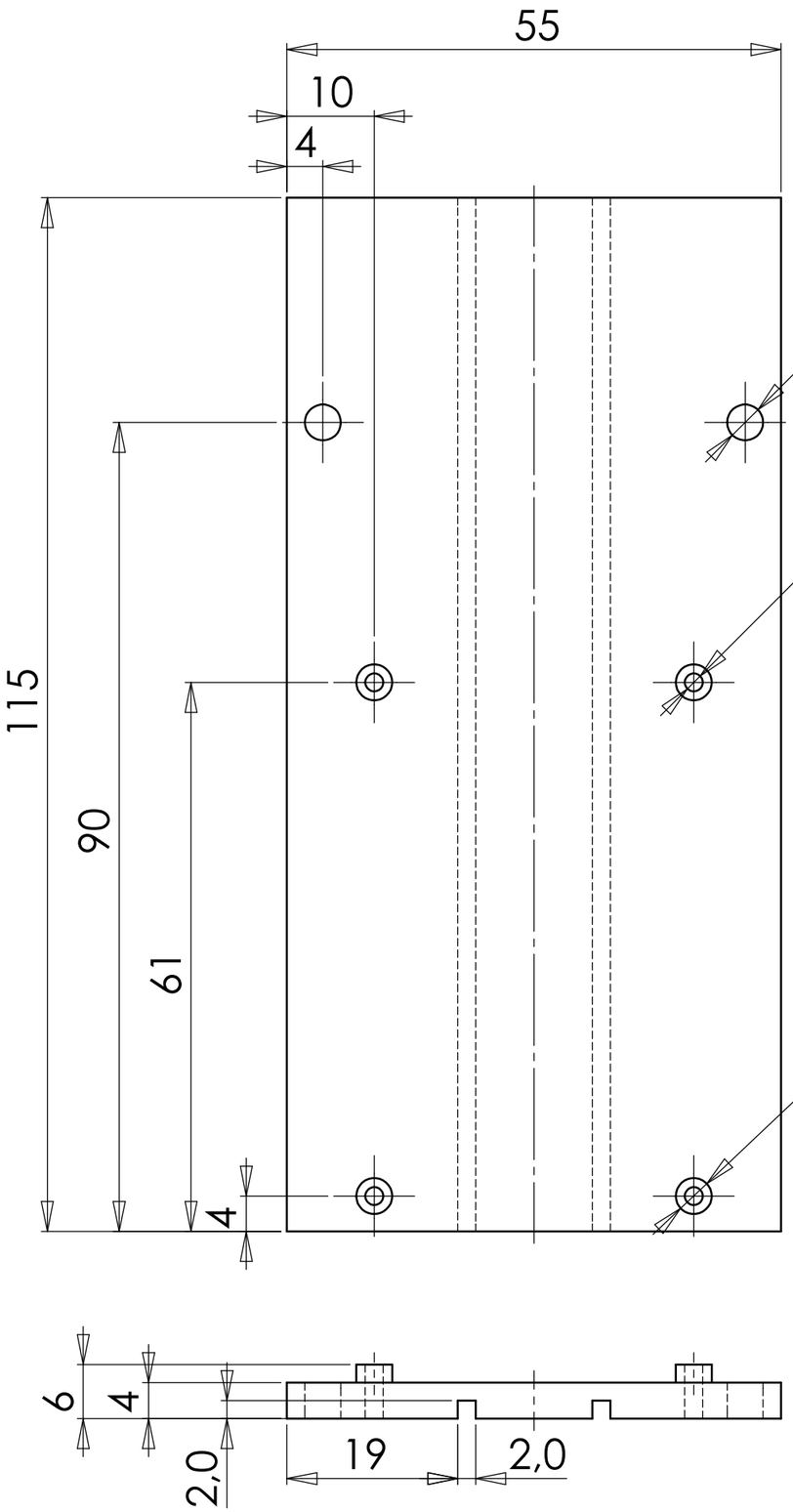
HOJA 5 DE 7





Las cotas cuya tolerancia no se encuentren en el plano, se rigen por la información de tolerancias del cajetín.

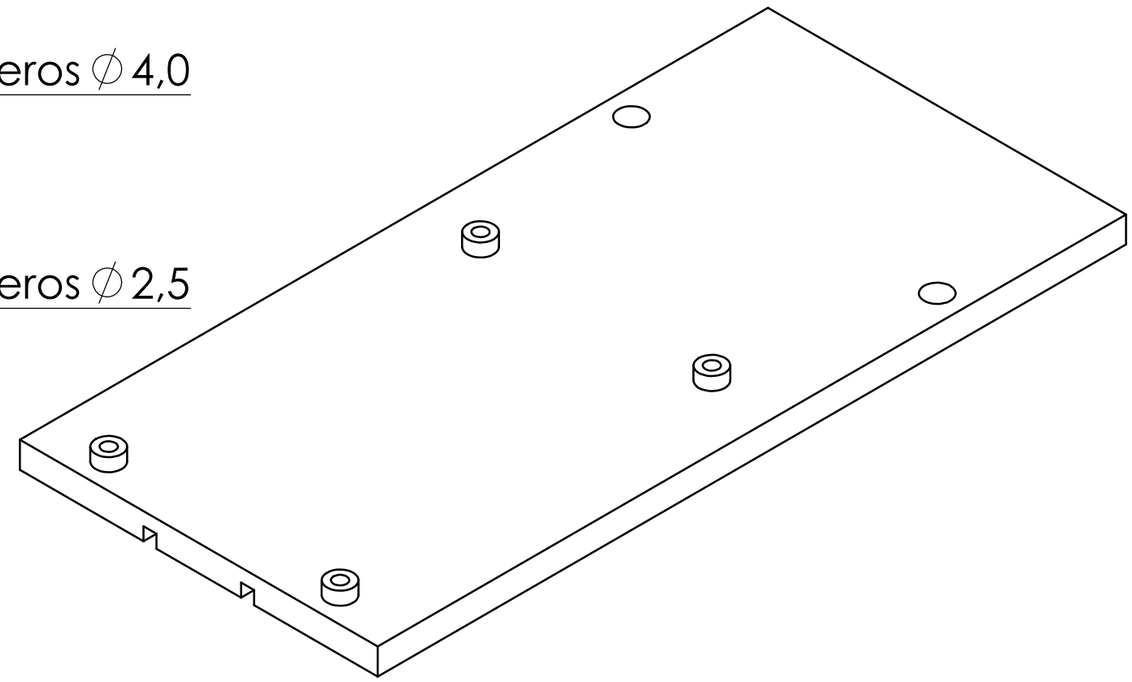
TEC | Tecnológico de Costa Rica				ÁreaAcadémica de Ingeniería Mecatrónica	REVISIÓN: 1
				CURSO: Proyecto de Graduación	
	NOMBRE	FIRMA	FECHA	ACABADO GENERAL: 12	DENOMINACIÓN: Soporte
DIBUJ.	D.Loría		25/6/20		
VERIF.	D.Loría				
APROB.				MATERIAL: ABS	
FABR.					
SISTEMA: FIRST ANGLE PROJECTION	TOLERANCIAS: LINEAL: X ± 0.5, .X ± 0.3, .XX ± 0.05, ANGULAR: ±° 1			TOLERANCIA EN: mm	DESIGNACIÓN: DAL-002-002 / ESCALA: 1,5:1 / FORMATO: A3
				PESO: -- gramos	UNIDADES: mm / HOJA 6 DE 7



2 agujeros ∅ 4,0

4 agujeros ∅ 2,5

4 relieves ∅ 4



Las cotas cuya tolerancia no se encuentren en el plano, se rigen por la información de tolerancias del cajetín.

TEC | Tecnológico de Costa Rica		ÁreaAcadémica de Ingeniería Mecatrónica	REVISIÓN: 1
		CURSO: Proyecto de Graduación	

	NOMBRE	FIRMA	FECHA
DIBUJ.	D.Loría		25/6/20
VERIF.	D.Loría		
APROB.			
FABR.			

ACABADO GENERAL: 12	DENOMINACIÓN: Tapa			
MATERIAL: ABS				
SISTEMA: FIRST ANGLE PROJECTION	TOLERANCIAS: LINEAL: X ± 0.5, .X ± 0.3, .XX ± 0.05, ANGULAR: ± 1°	TOLERANCIA EN: mm	DESIGNACIÓN: DAL-002-003	ESCALA: 1,5:1 / FORMATO: A3
		PESO: -- gramos	UNIDADES: mm	HOJA 7 DE 7

9.4 Anexo 4: Algoritmo para la predicción y detección con una Red LSTM

```

# Algoritmo para la detccion y clasificacion de fallos en UAVs

Proyecto de Graduacion

Dilan Andrey Loria Quesada

2020

#Se importan las librerias  necesarias para la ejecucion del programa

import numpy as np
np.random.seed(4)
import matplotlib.pyplot as plt
import pandas as pd

from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense, LSTM, GRU
from keras.models import model_from_json

# Commented out IPython magic to ensure Python compatibility.
#Se monta el proograma sobre la una carpeta en Google Drive

from google.colab import drive
drive.mount('/content/drive', force_remount=True)
# %cd 'drive/My Drive/ProyectoGraduacionDilan'

# Lectura de los datos desde el dos documentos de excel en el drive.

dataset_Y = pd.read_excel('LabelsENTRENAMIENTO.xlsx', index_col='ID', parse_dates=[
dataset_X = pd.read_excel('ENTRENAMIENTO.xlsx', index_col='Item', parse_dates=['Iten

datatest_X = pd.read_excel('VALIDACION.xlsx', index_col='Item', parse_dates=['Item'
datatest_Y = pd.read_excel('LabelsVALIDACION.xlsx', index_col='ID', parse_dates=['II

#Normalizacion del set de entrenamiento y validacion
```

```
sc = MinMaxScaler(feature_range=(0,10000))
set_entrenamiento_escalado = sc.fit_transform(dataset_X)
set_validacion_escalado = sc.fit_transform(datatest_X)

#Se agrupan los datos del conjunto de entrenamiento en vectores de la longitud
#de "Time Step"

time_step = 100
X_train = []
Y_train = []
m = len(set_entrenamiento_escalado)

for i in range(time_step,m+1,time_step):

    X_train.append(set_entrenamiento_escalado[i-time_step:i,:])


Y_train.append(dataset_Y)
X_train = np.array(X_train)

#Red LSTM: Se declaran los parametros y arquitectura de la red

dim_entrada = (X_train.shape[1],8)
dim_salida = 11
Neuronas = 50

modelo = Sequential()
modelo.add(LSTM(units=Neuronas, input_shape=dim_entrada))
modelo.add(Dense(units=dim_salida,activation= 'sigmoid'))
modelo.compile(optimizer = 'adam',loss='binary_crossentropy',metrics =['accuracy'])

#Se entrena el modelo y se alamacen valores del proceso en "histrory"

history = modelo.fit(X_train,Y_train, validation_split=0.10, epochs=100, batch_size

#Se grafican los valores de precision de entrenamiento y prueba
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Precision del Modelo')
plt.ylabel('Precision')
plt.xlabel('Epocas')
plt.legend(['Entrenamiento', 'Prueba'], loc='lower right')
```

```
80 plt.show()
81
82 # Se grafican los valores de perdida de entrenamiento y prueba
83 plt.plot(history.history['loss'])
84 plt.plot(history.history['val_loss'])
85 plt.title('Perdida del Modelo')
86 plt.ylabel('Perdida')
87 plt.xlabel('Epocas')
88 plt.legend(['Entrenamiento', 'Prueba'], loc='upper right')
89 plt.show()
90
91 #Se agrupan los datos del conjunto de validacion en vectores de la longitud
92 #de "Time Step"
93 time_step = 100
94 X_test = []
95 Y_test = []
96 m = len(set_validacion_escalado)
97
98 for i in range(time_step,m+1,time_step):
99
100     X_test.append(set_validacion_escalado[i-time_step:i,:])
101
102 Y_test.append(dataset_Y)
103 X_test = np.array(X_train)
104
105 #Evaluar el modelo con datos de validacion
106 scores = modelo.evaluate(X_test, Y_test, verbose=0)
107 print("%s: %.2f%%" % (modelo.metrics_names[1], scores[1]*100))
108
109 #Serializar el modelo a JSON
110 from tensorflow.keras.models import model_from_json
111 modelo_json = modelo.to_json()
112 with open("modelo.json", "w") as json_file:
113     json_file.write(modelo_json)
114
115 #Serializar los pesos a HDF5
116 modelo.save_weights("modelo.h5")
117 print("Modelo Guardado!")
118
119 # Se carga el modelo desde el drive
120 from tensorflow.keras.models import model_from_json
121 json_file = open('modelo.json', 'r')
```

```
122 loaded_model_json = json_file.read()
123 json_file.close()
124 loaded_model = model_from_json(loaded_model_json)
125
126 # Se caragn los pesos
127 loaded_model.load_weights("model.h5")
128 print("Loaded model from disk")
129
130 #Clasificacion con la red cargada a nuevos datos que vayan entrando
131
132 prediccion = loaded_model.predict(X_test[9:10])
133 categoria=np.argmax(prediccion)
134
135 if (categoria==0):
136   print("Motor Fallo")
137 elif (categoria==1):
138   print("Bateria Fallo")
139 elif (categoria==2):
140   print("Helice Rota")
141 elif (categoria==3):
142   print("Helice desprendida")
143 elif (categoria==4):
144   print("Desconexion de Fases")
145 elif (categoria==5):
146   print("Desmagnetizacion")
147 elif (categoria==6):
148   print("Rodamiento Falla")
149 elif (categoria==7):
150   print("Motor Bloqueado")
151 elif (categoria==8):
152   print("Aflojamiento de soporte")
153 elif (categoria==9):
154   print("Helice opuesta")
155 elif (categoria==10):
156   print("Funcionamiento Normal")
```

Algoritmo 9.2: Código de la red neuronal LSTM utilizada para la solución del problema.