TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE TUXTLA GUTIÉRREZ

Detección y localización de fallas en cuadrotores mediante aprendizaje de máquinas

Tesis que presenta

Luis Alejandro Méndez López

como requisito para obtener el grado de

Maestro en Ciencias en Ingeniería Mecatrónica

Director de tesis: **Dr. Francisco Ronay López Estrada**
Codirector de tesis: **M.C Ildeberto de los Santos Ruiz**

Tuxtla Gutiérrez, Chiapas, México

Febrero de 2020

DEDICATORIA

A mi familia, por su apoyo incondicional durante este periodo, ustedes han sido los cimientos de mi desarrollo, todos y cada uno de ustedes, han destinado tiempo para enseñarme cosas nuevas, para brindarme aportes invaluables que servirán para toda mi vida. Les agradezco infinitamente todo el amor que me brindan y la motivación por ser mejor día a día. ¡Gracias totales!

AGRADECIMIENTOS

Agradezco a mi director de Tesis, Dr. Francisco Ronay López Estrada, y a mi codirector el M.C. Ildeberto de los Santos Ruiz, por su tiempo, dedicación, enseñanzas, puntos de vista y su invaluable conocimiento que compartieron conmigo para el desarrollo del presente trabajo. Ustedes son mi mayor referente e inspiración para continuar preparándome en esta vida. A mis revisores el Dr. Héctor Ricardo Hernández de León y el M.C. Rafael Mota Grajales, por su apoyo incondicional y su vasta experiencia, para enriquecer este trabajo. Al ex-coordinador de la Maestría Dr. Carlos Ríos Rojas y al Coordinador en funciones, Dr. Rubén Grajales Coutiño, por abrirme las puertas del posgrado y estar presentes durante mi preparación en la Maestría, para que me fuese posible concluir satisfactoriamente este proceso.

Muchas gracias al CONACYT por haber financiado mi trabajo de investigación y preparación profesional, a través del programa de Becas Nacionales, dándome la oportunidad de crecer profesionalmente e incentivar en mí el gusto por la investigación científica.

Resumen

El presente trabajo aborda el tema de la detección y localización de fallas en los actuadores de un vehículo aéreo no tripulado (VANT), del tipo cuatrimotor, bajo un enfoque basado en el manejo de datos, a través de sensores que captan las vibraciones del vehículo, cuando éste se encuentra en vuelo estacionario. Estas vibraciones se procesan mediante técnicas de aprendizaje de máquina como el Análisis en Componentes Principales (*PCA*), para la extracción de sus características principales. Una vez extraídas las características, se post-procesan con algoritmos de clasificación como el *k-NN* y el *SVM* para que sea posible determinar si se presenta la falla, y en caso de que se encuentre presente, en cual de los cuatro motores se localiza. Al final se obtiene un algoritmo FDI robusto, y que se puede generalizar para otro tipo de vehículos.

Abstract

This paper addresses the issue of detecting and locating faults in the actuators of an unmanned aerial vehicle (UAV), of the four-engine type, under an approach based on data driven, through sensors that capture vehicle vibrations , when it is hovering. These vibrations are processed by machine learning techniques such as Principal Component Analysis (PCA), for the main feature extraction. Once the characteristics have been extracted, they are post-processed with classification algorithms such as *k-NN* and *SVM* so that it is possible to determine the presentation of the fault, and if it is present, in which of the four engines it is located. In the end, a robust FDI algorithm is obtained, which can be generalized for other types of vehicles.

Índice general

Índice de figuras

Capítulo 1

Introducción

Un vehículo aéreo no tripulado (VANT) es una aeronave sin piloto humano que se opera a través de una entrada electrónica iniciada por un controlador de vuelo o un sistema de control de gestión de vuelo autónomo a bordo (Nonami y cols., 2010). Este tipo de vehículos ha tenido un creciente interés debido a que proveen una gran área de aplicaciones en sectores como: transporte, exploración del medio y agricultura (Sharifi y cols., 2010); en vigilancia, búsqueda, rescate y aplicaciones de seguridad y militares (Mouloua y cols., 2001). Las capacidades para realizar vuelo estacionario, despegar y aterrizar verticalmente, los hace vehículos adecuados para estas tareas. Sin embargo, no todos los vehículos aéreos son capaces de desarrollar estas acciones. Los vehículos de ala fija, no son capaces de mantenerse en vuelo estacionario, dado que son vehículos que se desplazan de manera similar a los aviones, y son útiles para recorrer distancias largas, en un menor tiempo. Por su parte, los vehículos de ala rotativa, como los cutrimotores, tienen la capacidad de realizar vuelo estacionario y despegue vertical, por lo que son más aptos para las tareas antes mencionadas, y son objeto de estudio en sistemas de control.

Los vehículos cuatrimotores son vehículos que como su nombre lo indica, son elevados y propulsados por cuatro motores o también denominados rotores, lo que permite realizar maniobras en el aire y mantenerlos en vuelo estacionario, por lo que tienen gran potencial para vuelos en interiores y vuelos al aire libre. El control del vehículo cuatrimotor se logra variando la velocidad relativa de cada rotor para cambiar el empuje y el par producido por cada uno.

En los vehículos aéreos no tripulados los accidentes ocurren con frecuencia debido a fallas en los sistemas electrónicos, rotores y sensores(Mueller y D'Andrea, 2014; Guzmán-Rabasa y cols., 2019), ocasionando daño parcial o total al vehículo, generando pérdidas económicas (Figura 1.1). Por lo tanto, es de gran importancia la detección y localización de fallas en el vehículo para mejorar su seguridad y confiabilidad ante comportamientos anómalos. En la literatura se han propuesto diversas metodologías que proponen soluciones a este problema, aunque aún presentan deficiencias para la realización de la detección y el aislamiento de fallas. Por ejemplo, los métodos basados en modelos tienen el problema de que funcionan únicamente para cierto tipo de fallas que son previamente predeterminadas. En algunos casos la metodología basada en modelos se desarrolla mejor en la detección, pero no se obtienen buenos resultados en la identificación de la falla, y tampoco es posible generalizar para todos los tipos de vehículos aéreos.

En contraparte los métodos basados en datos tienen cierta flexibilidad al momento de realizar la detección y localización de la falla, dado que no necesitan un modelo matemático complejo que represente el funcionamiento del sistema. Basta con los datos provenientes de sensores, para generar un modelo del sistema, donde cualquier discrepancia con las condiciones nominales del modelo, represente una posible falla.

En este trabajo se aborda el problema de la detección y localización de fallas en los actuadores de un VANT cuatrimotor desde el enfoque basado en datos, a partir de las señales de vibración que son causadas por el mismo vehículo al estar en vuelo y que son captadas por sensores como los acelerómetros, lo que permite realizar un modelo del implícito del vehículo, e identificar a partir de esos mismos datos, cuando presenta alguna falla. Como resultado se obtuvo una metodología FDI robusta, que combina técnicas de estadística multivariable y aprendizaje de máquina, capaz de localizar las fallas con un margen de error muy pequeño.

1.1. Estado del arte

Existen dos metodologías para detectar fallas durante el vuelo del VANT; estas son las metodologías basadas en modelos matemáticos y las que se basan en datos. La primera se desarrolla a

Figura 1.1: Las fallas en los vehículos aéreos no tripulados generalmente ocurren debido a fallas en los sensores y actuadores

través de leyes físicas que rigen el movimiento traslacional y rotacional de los VANT generando señales residuales que comparan mediciones de sensores con estimaciones obtenidas del modelo (Saied y cols., 2015; Guzmán-Rabasa y cols., 2019) donde cualquier discrepancia se puede interpretar como una posible falla. Aunque este enfoque ha logrado aplicarse de manera exitosa (Vey y Lunze, 2016; Xian y Hao, 2019), usualmente es difícil parametrizar el modelo del sistema, debido a que algunos parámetros como los momentos de inercia, empujes y constantes aerodinámicas entre otros, no son fáciles de medir o estimar en laboratorio. En contraparte, los métodos basados en datos, utilizan un modelo implícito a partir de los datos empíricos del sistema, para que posteriormente se pueda utilizar un enfoque similar al de los métodos basados en modelos, en los que se caracteriza una zona de operación "nominal", y se compara con los datos experimentales (Gertler, 2015). Sin embargo, hay muchos desafíos importantes en el desarrollo de sistemas de Detección y Aislamiento de Fallas (FDI, *fault detection and isolation*) cuando el sistema es de dinámica variable y no lineal (H. Wang y cols., 2009), como es el caso de un cuatrimotor. Uno de estos problemas es el desarrollo de un sistema FDI que sea aplicable a cualquier vehículo aéreo, que además de esta generalización, permita identificar diversas fallas, cuando éstas se presenten durante el vuelo, cuya identificación se realice en un periodo de tiempo muy corto, para poder realizar acciones de control que eviten daños importantes en el vehículo, como la ruptura de alguna hélice.

En la literatura se encuentran publicadas algunas metodologías de sistemas basados en datos para la detección y aislamiento de fallas en este tipo de vehículos. Por ejemplo, en Yap (2014) se presentó el monitoreo de salud estructural para vehículos aéreos no tripulados,

considerando los efectos de tres posibles daños físicos, como una hélice rota, un tornillo suelto y un rotor dañado; para ello se instalaron acelerómetros microelectromecánicos (MEMS) y una Unidad de Medición Inercial (IMU) unidas el eje del cuatrimotor para inspeccionar la vibración característica y a través de la Transformada Rápida de Fourier (FFT) se analizaron las señales y se caracterizó el comportamiento de los rotores con estructuras defectuosas comparándose con el caso ideal de un rotor funcionando correctamente. Los datos obtenidos de los rotores fueron con 3 diferentes velocidades: baja velocidad (4 a 4.5 % del ciclo de trabajo o ancho de pulso de 800 a 900 μs), velocidad moderada (9 a 9.5 % del ciclo de trabajo o ancho de pulso de 1800 a 1900 μs) y velocidad alta (11 a 11.5 % del ciclo de trabajo o ancho de pulso de 2200 a 2300 μs). A pesar de que se logró detectar la falla y caracterizar las frecuencias dominantes para cada caso, mediante este algoritmo sólo se puede analizar un rotor a la vez, por lo que su análisis se complicaría al estar los 4 rotores funcionando, dado que existe una transmisión de vibración de un motor a los otros 3 motores, lo que implica relaciones aditivas y multiplicativas.

En Ghalamchi y Mueller (2018), se propuso la detección de fallas analizando el espectro de vibración en las trayectorias de vuelo. Para ello se incorporó un acelerómetro en el centro del cuerpo del cuatrimotor para el análisis del espectro de vibraciones. El argumento de esta técnica, se basa en que las hélices dañadas aplican vibración adicional al sistema durante la puesta en marcha del vehículo, y la detección de esas fallas ayuda a prevenir daños en otros componentes. Para ello se analizaron tres casos de fallas experimentalmente: una hélice dañada, dos hélices dañadas y dos hélices en posiciones opuestas dañadas. Se simularon las tres posibles fallas mediante un desbalance del vehículo, utilizando una masa de 0.001 g, colocada en el marco del vehículo. Se consideraron los efectos del desequilibro de la masa, sin extender este análisis al diagnóstico de fallas en el motor. A través de la FFT se analizó y caracterizó la frecuencia dominante para cada caso, con un desequilibrio en la masa de algunas de las propelas, y se detectó (mediante la firma del espectro) la ubicación de la hélice dañada, al comparar los diferentes espectros de vibraciones obtenidos en las trayectorias. En contraparte, a pesar del buen desempeño del método FDI, sólo se pudo realizar las trayectorias de vuelo bajo condiciones controladas en una arena de vuelo interior y fue necesario el uso de un sistema de captura de movimiento, lo que hizo mas costoso el sistema.

Una variante del estudio del Análisis en Componentes Principales (PCA), es el que se realizó en Li y cols. (2016), donde se utilizó el DPCA (Análisis en Componentes Principales Dinámico) para la detección y aislamiento de la falla. Esto debido a que el PCA no toma en cuenta las relaciones dinámicas de entrada y salida del sistema, por lo que se argumenta que la técnica DPCA con promedio móvil es más adecuada para este tipo de sistemas. En este artículo se aplicó el enfoque FDI para un sistema de control simulado de un vehículo de ala fija, en vuelo estable sin extenderse a los casos de maniobras de vuelo, y se utilizaron los estadísticos T^2 y *SPE* para el monitoreo de fallas de siete casos de estudio, por ejemplo, la desviación de medición en el transductor del ángulo de ataque o la desviación de medición vertical, por mencionar algunos. A través del método propuesto se obtuvieron desempeños satisfactorios en la detección de fallas con ambos índices estadísticos. Además de que el MA-DPCA fue capaz de reflejar perturbaciones desconocidas. En contraparte, en el aislamiento de la falla, el estadístico *SPE* no mostró un buen rendimiento, debido a que éste es vulnerable a la dinámica del sistema.

Otros autores proponen la combinación de técnicas para la detección y el aislamiento de fallas basadas en el manejo de datos y el aprendizaje de máquina, para tener un sistema más robusto. Como ejemplo en Jiang y cols. (2015) manejaron un metodología FDI basada en datos utilizando las señales de vibración producto de la aceleración en el vehículo. Se colocó un *smartphone* en el centro del marco del VANT, y a través del acelerómetro dentro del teléfono celular, se registraron los datos de las vibraciones del vehículo, bajo tres casos de estudio: una hélice completa, una hélice fracturada y una hélice distorsionada. Para el análisis de las señales se realizó la descomposición de los datos en paquetes de Wavelet para el análisis de dichas vibraciones. Dónde las desviaciones estándar de los coeficientes del paquete Wavelet construyeron los vectores de características que se utilizaron como señales de entrada para una RNA (Red Neuronal Artificial), con el objetivo de hacer el diagnóstico de las señales, donde la señal de salida de la RNA reflejaba es estado de salud estructural del rotor. Este sistema combinado obtuvo un 98 % de precisión en su desempeño. Los resultados erróneos se debieron principalmente a las perturbaciones en el ambiente y a la inexactitud en la medición por parte del acelerómetro.

En Baskaya y cols. (2017) se realizó un enfoque de implementación de prácticas de aprendizaje

automático para detectar y diagnosticar fallas en un VANT de ala fija. En primer lugar, los autores realizaron una simulación del modelo de un avión, que se utilizó para generar datos y probar los algoritmos diseñados. Los datos que se extrajeron fueron los que proporcionaron los sensores de aceleración y posición, donde se consideraron como fallas a las pérdidas de efectividad en la superficie de control del vehículo. Para la reducción y extracción de características de los datos, se hizo uso del PCA fuera de línea. La etapa de clasificación utilizaron el método supervisado de máquinas de vectores de soporte (SVM), separando los datos en dos clases: condiciones de vuelo nominal y condiciones de vuelo con falla. Los resultados demostraron que para las mediciones simuladas, el SVM da resultados muy precisos sobre la clasificación de la pérdida de efectividad en la superficie de control. Estas mediciones se realizaron en una cámara

También se han combinado métodos acústicos y redes neuronales artificiales. Como ejemplo se tiene en (Iannace y cols., 2019), donde a partir de las mediciones del ruido producidos por la hélice en su rotación en un VANT se utilizaron para construir un modelo de clasificación para detectar una hélice desequilibrada. Estas mediciones se realizaron en una cámara anecoica y luego se analizaron para caracterizar el fenómeno. Posteriormente un algoritmo de RNA fue construido para la etapa de detección. Este modelo mostró un valor de precisión de 97.63 %. Sin embargo este enfoque sólo es aplicable para el diagnóstico antes de iniciar el vuelo en el vehículo, para verificar que las condiciones de operación sean las adecuadas.

Algunos trabajos consideran un enfoque combinando sistemas basados en datos y sistemas basados en modelos, por ejemplo, Freeman y cols. (2013) consideraron la generación residual basada en modelos y la detección de anomalías basada en datos para un VANT pequeño que utilizó ambos tipos de enfoques y aplicó esos algoritmos a los datos experimentales de pruebas de vuelo con fallas y sin fallas; el enfoque basado en el modelo, utilizó métodos robustos de filtrado (un filtro H ∞) lineal para rechazar perturbaciones exógenas, como el viento, para proporcionar robustez para modelar errores y poder detectar fallas en los alerones; el enfoque basado en datos se desarrollo para operar con los datos en bruto de pruebas de vuelo, sin un conocimiento previo detallado de la dinámica del sistema. El detector basado en datos, se encargó de procesar las señales de errores en el control, y generaba un puntaje

de probabilidad de la falla. Ambos enfoques detectaron fallas reales en alerones aún con la presencia de perturbaciones. Los detectores basados en datos reconocieron fácilmente el comportamiento anómalo y tuvieron ventaja significativa con respecto al basado en modelo al mostrar un mejor rendimiento ante fallas inesperadas, sin embargo, no tuvieron un buen desempeño en simulaciones lineales con incertidumbre.

Los métodos de detección de fallas basados en datos también han sido utilizados para la detección de fallas en sensores abordo del VANT. En Sun y cols. (2017) se presenta un esquema de FDI basado en un sistema de inferencia neuro-difuso adaptativo (ANFIS) que combina un mecanismo de entrenamiento de datos en línea con el sistema de decisión basado en ANFIS. Asimismo, en B. Wang y cols. (2018) se presenta un sistema embebido en FPGA de un modelo de predicción basado en máquinas de vectores de soporte de mínimos cuadrados para el FDI de los sensores del VANT.

En este trabajo se considera una técnica basada en el análisis de componentes principales para la detección de fallas en los actuadores. Un sistema de detección de falla basado en PCA implica, de igual manera, un modelo implícito del sistema a partir de datos obtenidos de sensores. El criterio de selección subyacente es que el diagnóstico de fallas basado en datos permite evitar la carga del modelado preciso requerido en el enfoque basado en un modelo matemático. Para la parte experimental se realizaron dos análisis. El primer análisis fue realizado a través de los datos de sensores (encoders) que se encuentran dentro de una plataforma de vuelo tipo giroscopio (Valencia-Palomo y cols., 2018), que proporcionan información del desplazamiento del VANT en cada uno de los ángulos (alabeo, cabeceo y guiñada). Un segundo análisis se realiza con los datos de las aceleraciones lineales en cada uno de los ejes (x,y,z) obtenidos en la unidad de medida inercial (IMU) inmersa en la tarjeta de control de vuelo del VANT. El conocimiento de las características del sistema en condiciones de operación libre de fallas para plantear escenarios de detección y localización de fallas como un problema de tratamiento digital de señales.

El trabajo se plantea en términos de clasificación de datos a partir de la extracción de sus características principales en condiciones nominales y de la comparación con los datos del vehículo cuando se induce una falla.

1.2. Planteamiento del problema

En los vehículos aéreos no tripulados, las fallas más comunes son las que se presentan en los actuadores de los rotores, ocasionadas por el desprendimiento o ruptura, total o parcial de las hélices, cuando el vehículo se encuentra en pleno vuelo. Lo que se traduce en pérdidas económicas. Por lo tanto, en el problema del diagnóstico de fallas en VANT, se encuentra el desafío de la identificación de condiciones anormales y localización de fallas, cuando se inicia la puesta en marcha del vehículo, para que inmediatamente, sea posible combinarse con técnicas de control tolerante a fallas, dando la posibilidad de actuar y corregir su desplazamiento. A pesar de que en la literatura se han realizado numerosas metodologías para afrontar este problema, aún presentan deficiencias, y todavía representa un desafío significativo.

En este trabajo se propone implementar un método para la localización y detección de fallas en el VANT, a través de un enfoque basado en datos, utilizando técnicas estadísticas multivariables, que permita la extracción de características de dichos datos, y combinado con técnicas de aprendizaje de máquina, sea posible la realización de un algoritmo generalizado, y obteniendo como resultado un método confiable, robusto y eficiente para localizar y detectar dichas fallas.

1.3. Hipótesis

Mediante la técnica de análisis de componentes principales (PCA), en conjunto con técnicas de aprendizaje de máquina (*Machine Learning*), es posible realizar la detección y clasificación de fallas en los actuadores de un vehículo aéreo no tripulado cuatrimotor a través del análisis de datos obtenidos de sensores de desplazamiento angular y aceleración lineal.

1.4. Objetivos

1.4.1. Objetivo general

Diseñar un sistema de detección y localización de fallas en actuadores de un vehículo aéreo no tripulado de tipo cuatrimotor basado en técnicas de aprendizaje automático combinando Análisis de Componentes Principales y Técnicas de Aprendizaje de Máquina.

1.4.2. Objetivos específicos

- Obtener datos de los desplazamientos angulares a través de los encoders incorporados en la plataforma para simulación de vuelos, y de las aceleraciones lineales a través del acelerómetro incorporado en la IMU de la tarjeta controladora del cuatrimotor.
- Generar una base de datos que contenga información con las mediciones de los sensores.
- Diseñar algoritmos de extracción de características mediante PCA para reducir la dimensión de los datos y caracterizar las zonas de operación.
- Diseñar algoritmos de clasificación y localización de fallas en actuadores basados en técnicas de aprendizaje computacional
- Validar los algoritmos diseñados de manera experimental en la plataforma de vuelo tipo giroscopio dentro del laboratorio de control y automatización.
- Comparar los resultados del enfoque combinado con algoritmos basados en el manejo de datos que consideren únicamente una técnica para la detección de fallas, como el Análisis en Componentes Principales o la Transformada de Fourier.

1.5. Estructura de la tesis

La tesis está organizada en 5 capítulos. A continuación se presenta una breve reseña del contenido en cada capítulo.

En el Capítulo 2 se presenta el fundamento teórico donde se abordan los temas que fueron la base para el desarrollo de este trabajo, por ejemplo, los sistemas de detección y localización de fallas o los algoritmos de clasificación, sólo por mencionar algunos.

En el Capítulo 3 todos los materiales y los métodos utilizados para el desarrollo de la parte experimental, esencial para obtener los resultados y comprobar la efectividad de la metodología propuesta, por ejemplo, la tarjeta controladora de vuelo utilizada para el experimento o el software necesario para el diseño de los algoritmos.

Los resultados y las discusiones se encuentran plasmados en el Capítulo 4, acompañados de gráficas que hacen de manera visual más entendible las aportaciones de la presente tesis.

En el Capítulo 5 se presentan las conclusiones de los resultados obtenidos, así como las perspectivas de los trabajos que se pueden realizar a futuro para mejorar dicha metodología propuesta. En la sección de anexos se encuentra la programación e información adicional que se requiere para replicar (en caso de ser necesario) este trabajo.

Capítulo 2

Fundamento teórico

En este capítulo se abordan los conceptos generales sobre los temas relacionados con este trabajo de investigación, así como las bases físicas y matemáticas para el desarrollo del mismo. Se presenta una explicación acerca de los sistemas de clasificación de fallas, y se describe el sistema de detección y clasificación de fallas basado en el análisis de componentes principales (PCA), los clasificadores mediante vecinos cercanos y las máquinas de vectores de soporte (SVM).

2.1. Vehículos Aéreos no Tripulados

Los vehículos aéreos no tripulados (VANT) han sido denominados de diferentes maneras a lo largo del tiempo, como drones, aviones robot, aeronaves sin piloto, vehículos pilotados a distancia, aeronaves pilotadas a distancia, entre otros términos que describen a éstas aeronaves que vuelan bajo el control de un operador sin una persona a bordo. Usualmente se les denomina VANT, y cuando se combinan con estaciones de control en tierra y enlaces de datos, forman un sistema de vehículo aéreo no tripulado. Los VANT varían ampliamente en tamaño y capacidad. Por ejemplo, pueden tener una envergadura tan amplia como un Boeing 737 o más pequeño que un radio control. Aunque a menudo están asociados con la actividad militar, existe un gran interés y desarrollo en el sector privado. Esto se debe en gran medida al costo decreciente de la tecnología en los VANT, y al hecho de que poseen ventajas funcionales

distintas al de los vehículos aéreos tripulados. Los estándares de referencia desarrollados por la comunidad internacional de VANT para su clasificación, se basan en parámetros tales como la altitud de vuelo, resistencia, velocidad, despegue máximo, peso y tamaño. Las tres características principales del sistema VANT son:

- Aeronaves con sensores característicos comunes
- Estación de control en tierra, que puede incluir un centro de procesamiento de datos
- Operador o una serie de instrucciones de software

A pesar de que las consideraciones de diseño y de rendimiento en un VANT son similares a los que se presentan en la aviación tripulada, en los diseños del VANT no se toma en cuanta a un piloto a bordo. Esto propicia la ventaja de reducir la resistencia aerodinámica y el peso (debido a la eliminación de elementos como la cabina), así como la capacidad de mantener una mayor estabilidad y permiten realizar maniobras de vuelo más complejas. El avance tecnológico ha propiciado a que las plataformas de los VANT sean más confiables en términos de control de vuelo, y las tecnologías avanzadas de telecomunicaciones permiten el control a gran altura sobre distancias considerables. En Cavoukian (2012) se clasifican a este tipo de vehículos en 3 clases que se presentan a continuación: Los Micro y mini-VANT, son los de menores dimensiones.Vuelan a alturas bajas (por debajo de los 300 metros). El diseño de los vehículos de esta categoría se basan en que su uso es en el ambiente urbano, para volar dentro de edificios, volar a través de pasillos, transportar dispositivos de escucha y grabación, o transportar cámaras de televisión en miniatura. Tienen un peso que va desde los 100 gramos para los micro-VANT y de menos de 30 kilogramos para los mini-VANT. Su principal uso es en aplicaciones civiles y militares; La segunda clase es la de los VANT tácticos, los cuales son más pesados (de 150 a 1500 kilogramos). Vuelan a alturas de 3,000 a 8,000 metros y que actualmente se utilizan para realizar aplicaciones militares. Los vehículos de esta categoría pueden funcionar durante más de 40 horas en un rango máximo de 3000 kilómetros. La tercera categoría es la de los VANT estratégicos, los cuales tienden a ser plataformas más pesadas con rangos de vuelo más largos, llegan a pesar hasta 12,000 kilogramos y tienen una altura máxima de vuelo de unos 20,000 metros. Se aplican en la observación de puntos estratégicos

de la tierra, mapeo y monitoreo atmosférico.

Otra clasificación que se le otorga a los vehículos, es de acuerdo a las características de vuelo. Esta clasificación divide a los VANT en vehículos de ala fija y vehículos de ala rotatoria (Figura 2.1). Dependiendo el tipo de misión u objetivo que se pretende realizar, uno de estos modelos se impondrá sobre el otro. Por ejemplo, en aquellos casos donde se requiera que el dispositivo realice maniobras en forma estacionaria y/o a baja velocidad, el vehículo mas adecuado sería el de ala rotativa. En cambio, si se desea utilizar estos dispositivos para realizar vuelos a velocidades superiores en menor tiempo, como en aplicaciones para recolectar datos cartográficos, la opción más adecuada es la de optar por el uso de un vehículo de ala rotatoria.

Figura 2.1: Clasificación de los vehículos aéreos no tripulados.

2.2. Métodos de detección y localización de fallas

Los métodos de detección y localización de fallas (FDI, *Fault detection and isolation*) o también referidos en la literatura como métodos de detección y diagnóstico de fallas (FDD, *Fault detection and diagnosis*) han sido objeto de numerosos estudios, especialmente en el área de control y automatización. Las fallas son disfunciones de varios elementos de los sistemas técnicos (planta) que pueden afectar varias partes del sistema técnico principal o dispositivos que interactúen con éste (Gertler, 2015). Se pueden agrupar a las fallas en varias categorías: Fallas en actuadores, en sensores, en sistemas y operativas. Una falla se puede clasificar como aditiva o multiplicativa. Las fallas aditivas representan aumentos o decrementos en los estados del sistema y corresponden a fallas en sensores y actuadores. Las fallas multiplicativas representan aumentos o decrementos de los parámetros del sistema y corresponden a fallas en sus componentes internos.

La detección de fallas tiene como objetivo principal determinar y señalar si existe una falla. La localización de una falla permite identificar en que punto físico del sistema se ubica dicha falla. El diagnóstico de falla aporta información específica sobre la falla.
Los sistemas utilizados para FDI son:

- Control de límites o sistema de alarma. Este sistema compara límites preestablecidos con mediciones de plantas individuales con límites respectivos. Presentan deficiencias dado que tienen una sensibilidad de falla muy limitada. Sin, embargo son ampliamente usados en aplicaciones industriales.
- Métodos basados en modelos matemáticos. Estos sistemas comparan mediciones de la planta con las estimaciones obtenidas del modelo obtenido en otras mediciones. Cualquier discrepancia se puede interpretar como una falla posible.
- Métodos basados en manejo de datos. Se basan en la estimación de modo implícito a partir de datos empíricos de la planta, para que posteriormente, se pueda usar un enfoque similar basados en modelos, en los que se caracteriza una zona de operación "normal", y posteriormente se compara con los datos experimentales.

Otra clasificación que se puede encontrar para los sistemas FDI es el que menciona Zhang y Jiang (2008), en el que se analizan más a fondo los aspectos cualitativos y cuantitativos, de los enfoques basados en modelos y de los enfoques basados en datos, como se observa en la Fig.2.2. Según Zhang y Jiang (2008) la introducción de nuevos sistemas para el diagnóstico de fallas, como los que se basan en datos, se debe a que los sistemas tecnológicos modernos requieren sistemas de control más sofisticados, para cumplir con los requisitos de mayor seguridad y desempeño. Por ejemplo, el diseño de un sistema de control convencional para un sistema complejo, puede tener como resultado un desempeño insatisfactorio, o incluso inestable, ante el mal funcionamiento de sensores, actuadores u otros componentes del sistema. El enfoque basado en datos permite manejar una gran dimensión de datos, y con ayuda de técnicas de reducción de dimensión de datos, se puede identificar y resaltar información importante dentro de ese conjunto de datos. Particularmente, dentro de los sistemas de manejo

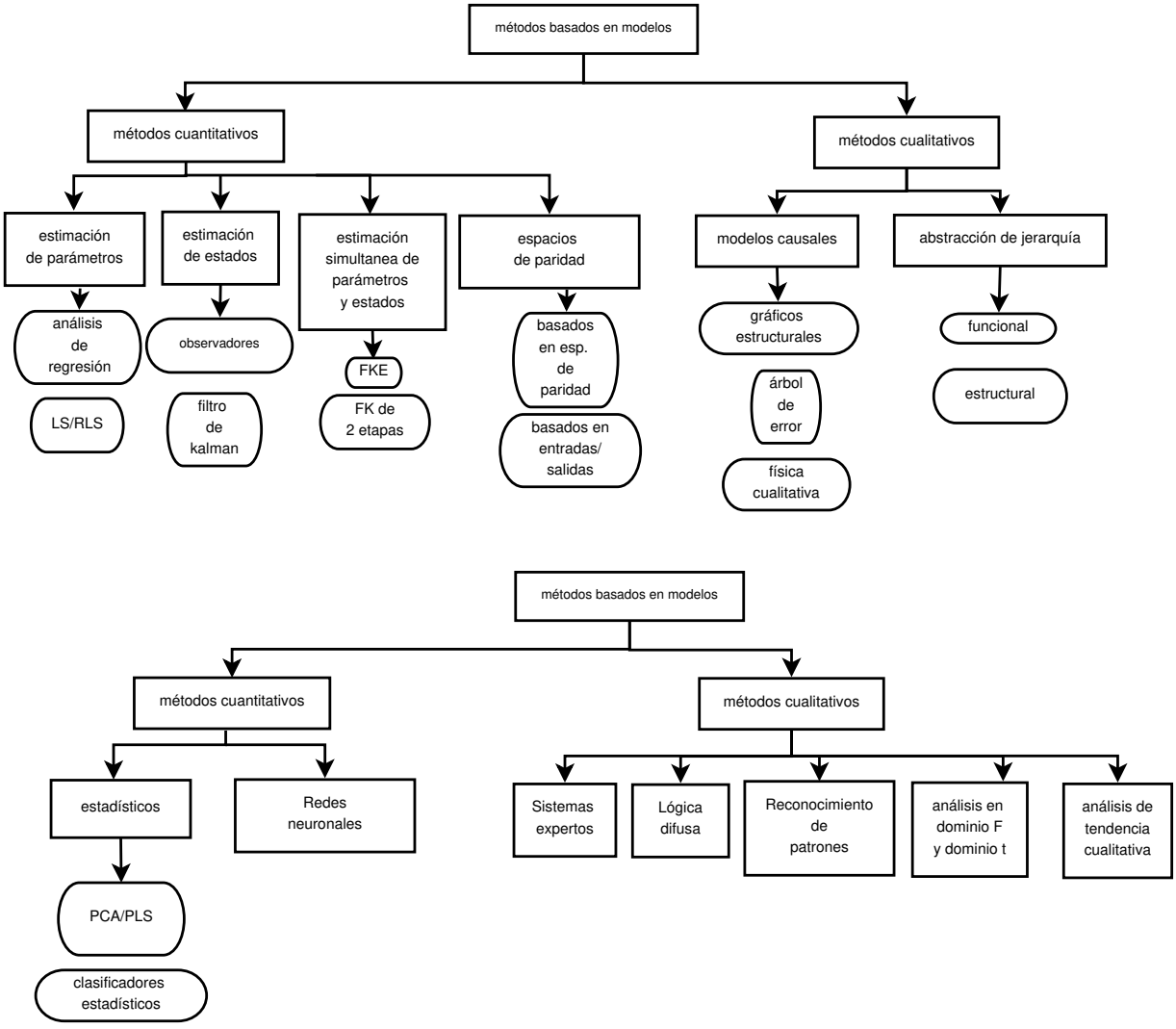


Figura 2.2: Clasificación de los métodos de detección y diagnóstico de falla. (Zhang y Jiang, 2008)

de datos, el Análisis de Componentes Principales, se ha usado ampliamente en el monitoreo de plantas complejas.
A continuación se presenta el Sistema de detección de fallas basados en modelos y el Sistema de PCA, siendo este último el de interés para la realización de este trabajo.

2.2.1. Métodos de detección de fallas basados en Modelos Matemáticos

La obtención de un modelo matemático de la planta se obtiene mediante métodos de identificación del sistema, a partir de la compresión e interpretación de la dinámica de la planta. Esta es la parte más importante para la FDI, aunque generalmente no se considera parte del esfuerzo. Dichos modelos pueden ser: Lineales o no lineales; Estáticos o dinámicos; Continuos o discretos. La mayoría de las veces los sistemas son dinámicos de tiempo discreto.
Cualquier discrepancia en la comparación de los resultados medidos de la planta con las estimaciones obtenidas a través del modelo matemático, es un indicativo de falla (Willsky, 1976). Esto se expresa como:

$$e_i(t) = y_i(t) - \widehat{y}_i(t) \tag{2.1}$$

Donde $e_i(t)$ es el residuo o error, $y_i(t)$ es la salida medida, e $\widehat{y}_i(t)$ es la salida estimada. Cuando el residuo es igual a cero, se define como la ausencia de una falla (Fig. 2.3).
Las fallas, los ruidos y perturbaciones hacen que los residuos sean distintos de cero. Es por ello que el algoritmo de FDI debe diseñarse de manera robusta ante las perturbaciones y errores del modelo, e insensible al ruido.

Como se mencionó anteriormente, las fallas se pueden clasificar como aditivas o multiplicativas. En la siguiente relación de entrada-salida $\mathbf{U}(t)$ es el vector de observación de entradas de la planta, $\mathbf{y}(t)$ es el vector de salidas de mediciones de la planta, $\mathbf{p}(t)$ es el vector de fallas aditivas y t es el tiempo discreto. $\mathbf{M}(q)$ y $\mathbf{S}(q)$ son las matrices de función de transferencia en el operador de cambio q, y θ

$$\mathbf{y}(t) = \mathbf{M}(q,\theta)\mathbf{u}(t) + \mathbf{S}(q,\theta)\mathbf{p}(t) \tag{2.2}$$

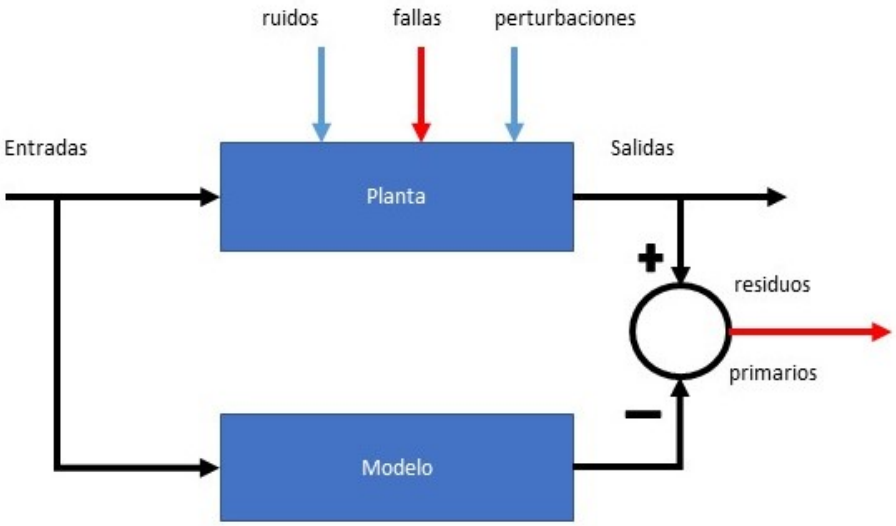


Figura 2.3: Redundancia Analítica. (Gertler, 2015)

El vector residual primario $\mathbf{e}(t)$, en respuesta a la falla aditiva es:

$$\mathbf{e}(t) = \mathbf{y}(t) - \mathbf{M}(q,\theta)\mathbf{u}(t) = \mathbf{S}(q,\theta)\mathbf{p}(t) \tag{2.3}$$

Si existen fallas multiplicativas, entonces $\theta = \theta^0 + \Delta\,\theta$ donde θ^0 es el vector de parámetro nominal y θ^0 es el cambio (la falla paramétrica); Se obtiene el vector residual $\mathbf{e}(t)$, que en Gertler (2013) define como:

$$\mathbf{e}(t) = \mathbf{y}(t) - \mathbf{M}(q,\theta^0)\mathbf{u}(t) = \Sigma_j(\partial\mathbf{M}(q,\theta)/\partial\theta_j)\mathbf{u}(t)\Delta\theta_j \tag{2.4}$$

Para facilitar el aislamiento de fallas, los residuos primarios $\mathbf{e}(t)$ están sujetos a alguna manipulación de mejora. Los generadores residuales toman observaciones de entrada y salida de la planta y generan residuos mejorados, utilizando el modelo matemático de la planta (Figura 2.4).

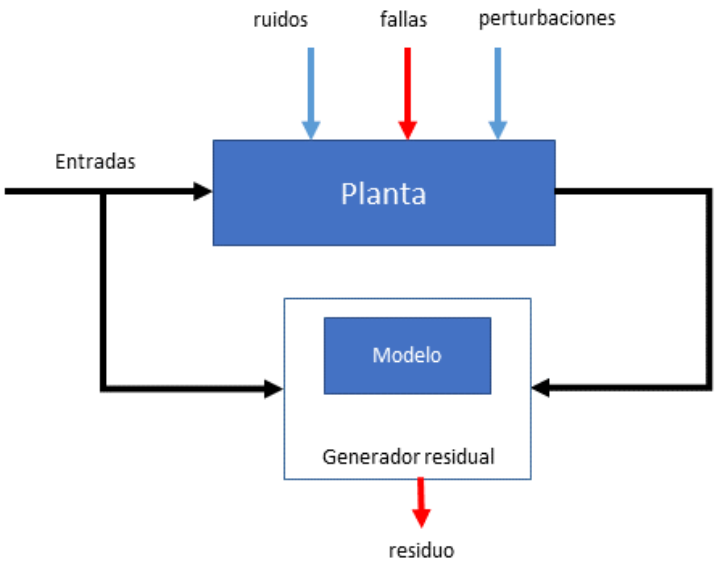


Figura 2.4: Generador residual basado en modelos (Gertler, 2015)

2.2.2. Métodos de detección de fallas basados en datos

Los métodos de detección de fallas basados en datos tienen la característica de manejar altas dimensiones de datos, que generalmente se apoyan de técnicas que permitan resaltar información importante dentro del volumen de datos. Sin embargo, aún presentan desafíos en su uso cuando el sistema es variante en el tiempo y altamente no lineal. La detección de fallas impulsada por datos ha pasado por tres fases importantes durante su desarrollo. Estas tres fases son: la detección de fallas basada en señal, basada en estadística multivariable y la basada en el conocimiento del sistema. La característica común que presentan estos métodos es que todos usan datos del sistema sin procesar y el conocimiento del sistema para lleva a cabo la detección de falla (FD) requerida. Dentro del primer grupo de métodos FD basados en datos, se encuentran los que se basan en señales.

Éstos utilizan métodos de procesamiento de señal que consisten en funciones de correlación, identificación de modelos de señal, verificación de paridad de señal y análisis espectral utilizando la transformada rápida de Fourier (FFT), o la transformación de Wavelet. Esto es similar a la detección de señal y detección de tendencia para variables importantes que utilizan datos disponibles. La idea clave es que los cambios inesperados en la magnitud, el cambio de fase y/o el cambio en las frecuencias de las señales importantes, puedan considerarse como fallas

en el sistema. Dentro del control de procesos la estadística multivariable se ha aplicado para detectar cambios de distribución anormales de datos de calidad en las líneas de producción para que se generen alarmas en tiempo real cuando los datos se encuentran fuera de los límites de distribución superior e inferior. La técnica clave en el análisis multivariable es el PCA, cuyas características es reducir un volumen de datos de alta dimensión en un volumen dimensional más bajo, donde los datos de baja dimensión contiene la mayor parte de la información útil, que es al mismo tiempo, la mayor variación contenida de datos pertenecientes al conjunto original. Los ejes de proyección se denominan componentes principales.

Como Tal el PCA ha sido ampliamente utilizado en el control de procesos industriales como una técnica estándar para el análisis de datos y la identificación de anomalías de procesos. En términos de detección de fallas, un conjunto de componentes de PCA deben determinarse para el conjunto de datos en buen estado, y después, la detección de fallas comprobando si los nuevos datos entrantes se encuentran, o no, en el espacio abarcado por los componentes principales en buen estado.

Dado que el PCA no utiliza relaciones de entrada y salida directamente, es una técnica más adecuada para la detección de fallas en lugar de la etapa de diagnóstico. Además de que el PCA supone que los datos son distribuidos en distribución gaussiana, lo que limita su aplicación en procesos industriales complejos, que exhiben una variación en el tiempo, no son gaussianas y de características no lineales. Sin embargo, para superar estas dificultades se han realizado modificaciones al estándar FD basado en PCA. Estas modificaciones son técnicas dinámicas de PCA y su acoplamiento con redes neuronales, las cuales ya se han aplicado a procesos industriales. Además del PCA existen otras técnicas como alternativas para la solución del problema de FD, por ejemplo, el análisis de componentes independientes (ICA), el cual, divide los datos observados en una combinación lineal de componentes independientes. Esto permite el uso de gráficos estadísticos para realizar el monitoreo en línea requerido y llegar al diagnóstico de fallas. Una observación ineteresante es que el ICA, no requiere que los datos se extraigan de una distribución gaussiana y, es más aplicable a los procesos industriales, donde generalmente el conjunto de datos no obedece a la distribución normal.

2.2.3. Métodos cualitativos basados en modelos y estrategias de búsqueda

Estos métodos están basados en la estrategia de razonamiento de causa-efecto sobre el comportamiento del sistema. Los métodos más populares en esta categoría son los arboles de fallas y la teoría de grafos. Sin embargo, una limitante de estos métodos es la generación de un gran número de hipótesis de posibles fallas, haciendo más difícil la toma de decisiones (Puig y cols., 2004).

2.2.4. Métodos basados en el historial de proceso

Estos métodos requieren de una gran cantidad de datos del sistema a revisar. De igual forma se pueden subdividir en cualitativos y cuantitativos. De los métodos cualitativos de esta categoría destacan: Sistemas expertos y Análisis de tendencias (Qualitative Trend Analysis). Los primeros consisten en una serie de antecedentes (una serie de eventos) y una parte de consecuencias, la cual mapea dichos eventos a una falla conocida Maurya y cols. (2005). La información del proceso entra al sistema en forma de estos antecedentes y consecuencias. Esto implica un mapeo explícito de síntomas conocidos a causas originarias de la falla. El QTA utiliza la información presente en las mediciones de los sensores. Existen dos pasos básicos: identificación de las tendencias en las mediciones e interpretación de tendencias en términos de los escenarios de fallas Maurya y cols. (2007). Por otro lado, los métodos cuantitativos de esta categoría, son métodos estadísticos (PCA/PLS). El Análisis de las Componentes Principales (PCA), es una herramienta estadística aplicable a sistemas multivariantes, que permite la transformación de los datos multivariantes a un espacio de menor dimensión el cual retiene la información más relevante acerca del proceso Ding y cols. (2010). Esta compresión de información favorece el uso de esquemas de monitoreo de procesos multivariantes mediante técnicas aplicadas a procesos univariados como gráficas de Control Estadístico de Procesos (SPC). Así, una muestra actual del proceso se compara con las condiciones de operación normal resumidas en la gráfica de SPC, para detectar fallas en sensores y actuadores así como del proceso.

2.2.5. Metodología de diseño de algoritmos FDI

En Blanke y cols. (2006) se propone una metodología para el diseño de sistemas FDI, la cual se basan esencialmente en 5 etapas, que se encuentran descritas a continuación:

1. Análisis del sistema a dos niveles: a nivel de componentes mediante un análisis de propagación de fallas a través de todos los subsistemas más relevantes, así como una evaluación de la severidad de los mismos y a nivel de estructura de cara a analizar la redundancia presente en el sistema que ayudará en el diseño del sistema de diagnóstico y reposición.

2. Diseño del sistema de diagnóstico a partir del análisis estructural y teniendo en cuenta las medidas disponibles y las fallas que se desean diagnosticar. En el caso de que no se puedan diagnosticar todas las fallas que se deseen, se deberá modificar la instrumentación disponible hasta conseguirlo. El sistema de diagnóstico de fallas deberá no sólo detectar y aislar las fallas sino también estimar su tamaño.

3. Diseño de los mecanismos de tolerancia para cada uno de las fallas consideradas según se trate de fallas en sensores, actuadores y/o planta.

4. Diseño del supervisor a partir de la información acerca de las fallas proporcionada por el sistema de diagnóstico, el supervisor deberá activar los mecanismos de tolerancia que se han diseñado para cada uno de ellos.

5. Aplicación y pruebas en simulación y sobre el sistema real.

La primera tarea a realizar en un sistema de control tolerante activo consiste en el diagnóstico de la falla en tiempo real, llegando no sólo a su detección y aislamiento sino también a la estimación de su magnitud. Por lo tanto, el diagnóstico de falla se puede a su vez dividir en tres etapas según su profundidad:

- Detección de la falla: decisión de si existe o no una falla así como la determinación de el momento de su de aparición.
- Aislamiento de la falla: localización del componente en el cual se ha producido la falla.
- Identificación y estimación de la falla: identificación del modo de falla y estimación de su magnitud.

Como todo proceso, existen diversas características o elementos a considerar en el diseño de algún algoritmo de diagnóstico. A continuación se presentan las características deseables en un sistema de diagnóstico:

- Rápida detección y diagnóstico: El sistema debe responder rápidamente en la detección y diagnóstico de fallas. Sin embargo, esto es difícil de lograr debido a que una respuesta rápida implica que haya alta sensibilidad a las influencias de altas frecuencias.
- Aislabilidad: el sistema debe ser capaz de distinguir entre dos o más fallas. Es decir identificar la falla ocurrida de forma precisa.
- Robustez: el sistema debe ser robusto ante ruido e incertidumbre del modelo matemático, ruido de medición y perturbaciones externas. De forma que no se vea afectado o atenúe el efecto de dichas señales espurias.
- Identificación de nuevas fallas: el sistema de diagnóstico debe ser capaz de decidir, si el proceso está trabajando de forma normal o no. En caso de que funcione de manera anormal, debe ser capaz de reconocer si se debe a una falla conocida o bien si se trata de un nuevo tipo de falla.
- Estimación del error de clasificación. Esta característica es para generar confianza en el usuario, con base a la estimación de posibles errores que se puedan presentar al realizar la clasificación.
- Adaptabilidad: Que el algoritmo sea capaz de adaptarse a cambios inesperados tanto de la estructura como del ambiente que rodea al sistema.
- Facilidad de explicación del origen de la falla: Además de la habilidad de identificar la fuente de la falla, un sistema debe ser capaz de proveer explicaciones de cómo se originaron y propagaron las fallas hasta las condiciones actuales del proceso.
- Identificación de múltiples fallas. Es un requerimiento difícil de lograr, pero es importante que el sistema pueda identificar varias fallas a la vez.

- Almacenamiento y requerimientos computacionales. Generalmente las soluciones rápidas en tiempo real requieren de algoritmos de baja complejidad computacional, sin embargo, requieren de mucho espacio de almacenamiento de datos. En la actualidad se prefiere buscar un balance entre estos requerimientos

2.3. Análisis en componentes principales

El análisis en componentes principales (PCA por sus siglas en inglés,*Principal Component Analysis*) es una técnica estadística multivariada que se usa ampliamente en el monitoreo de procesos industriales (Russell y cols., 2012). Mediante cálculos matriciales básicos el PCA permite construir un modelo *data-driven* (basado en datos) del comportamiento de un proceso, el cual puede ser utilizado después para determinar si las mediciones actuales de las variables del proceso se ajustan estadísticamente al comportamiento normal (dentro de ciertos límites de confianza) o si se infiere la falla u otro comportamiento anómalo.
El análisis de componentes principales se usa ampliamente en el monitoreo de plantas complejas con cientos de variables porque, al revelar relaciones lineales entre las variables, reduce significativamente la dimensionalidad del modelo de planta. Esta técnica es un enfoque estadístico, que permite la transformación de datos multivariados a un espacio de menor dimensión sin pérdida de generalidad (Jolliffe, 2011).Los nuevos componentes lineales son combinaciones lineales de las variables originales, con la característica adicional de que ahora se presentan como independientes. El PCA consta de dos fases: Entrenamiento y Monitoreo. En la fase de entrenamiento se crea un modelo de planta implícito a partir de los datos empíricos. En la fase de monitoreo, este modelo se utiliza para la FDI.

Para llevar a cabo el proceso del PCA se requiere de construir la matriz de mediciones

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_n \end{bmatrix} \in \mathbb{R}^{m \times n}; \tag{2.5}$$

de tal forma que se construye la matriz **X** como:

$$\mathbf{X} = \begin{bmatrix} x_1(1) & x_2(1) & \dots & x_n(1) \\ x_1(2) & x_2(2) & \dots & x_n(2) \\ \vdots & \vdots & \ddots & \vdots \\ x_1(m) & x_2(m) & \dots & x_n(m) \end{bmatrix}, \tag{2.6}$$

donde n representa el número de variables de proceso y m es el número de muestras tomadas para cada variable.

Posteriormente se requiere escalar los vectores de datos de **X** de tal forma que se tenga una media cero y varianza unitaria, para luego formar la matriz de puntuaciones estándar $\mathbf{Z} = \begin{bmatrix} \mathbf{z}_1 & \mathbf{z}_2 & \dots \mathbf{z}_n \end{bmatrix} \in \mathbb{R}^{m \times n}$

$$\mathbf{z}_k = \frac{1}{\sigma_k}(\mathbf{x}_k - \mu_k \mathbf{1}_m), \tag{2.7}$$

donde $\mathbf{1}_m = \begin{bmatrix} 1 & 1 & \dots & 1 \end{bmatrix}^T \in \mathbb{R}^m$, $\mu_k = \text{media}(x_k)$ y $\sigma_k = \text{std}(x_k)$. Esta estandarización permite una ponderación equitativa de la variabilidad de los datos debido a que las variables de proceso tienen sus valores en diferentes rangos y unidades de medida. A partir de esta matriz de datos se inicia la extracción de características del sistema. Esta técnica consiste en encontrar una matriz cuadrada ortogonal, que se define como:

$$\mathbf{P} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \dots \mathbf{p}_n \end{bmatrix} \in \mathbb{R}^{n \times n}; \tag{2.8}$$

donde **P** debe expresar el dato en $\mathbf{Z} \in \mathbb{R}^{m \times n}$ en términos de una nueva matriz con las nuevas coordenadas de datos $\mathbf{T} = \begin{bmatrix} \mathbf{t}_1 & \mathbf{t}_2 & \dots & \mathbf{t}_n \end{bmatrix} \in \mathbb{R}^{m \times n}$ referidas a la base ortonormal p_k (véase Fig. 2.5, donde el vector unitario $\mathbf{p_1}$ (primera componente) sigue la dirección de máxima varianza de los datos y $\mathbf{p_2}$ (segunda componente) representa una menor varianza de los datos):

$$\mathbf{T} = \mathbf{ZP}. \tag{2.9}$$

Y las mediciones estandarizadas pueden recuperarse a partir de **T** y **P**:

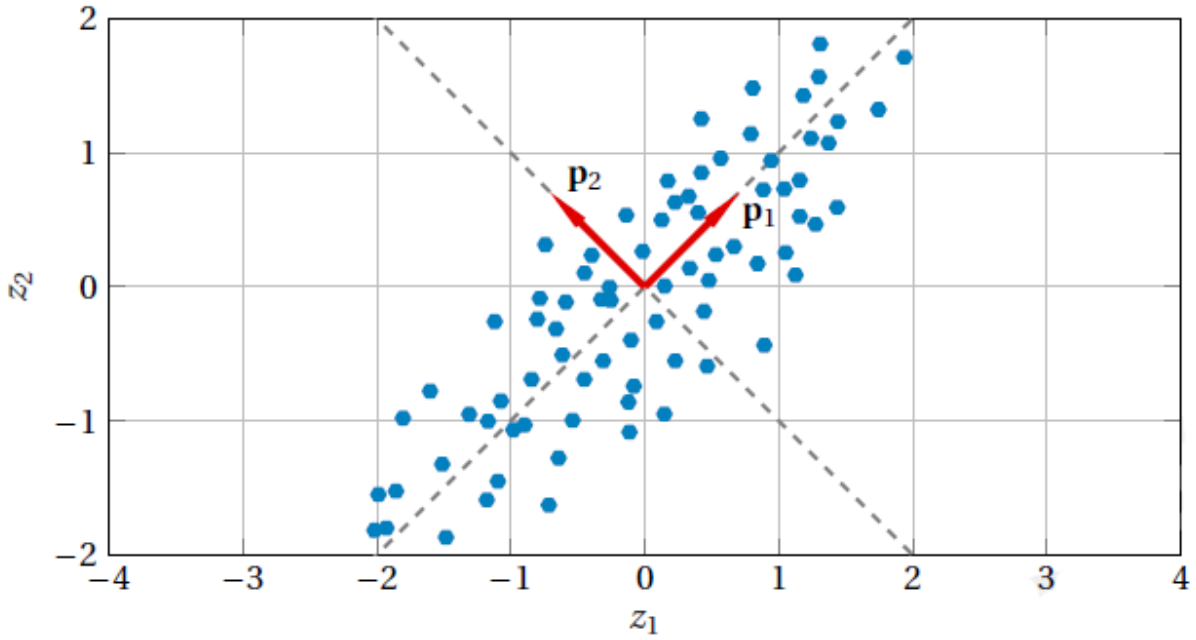


Figura 2.5: Base ortonormal para l cambio de coordenadas en el PCA, caso bidimensional.

$$\mathbf{Z} = \mathbf{T}\mathbf{P}^{\top}. \tag{2.10}$$

Las columnas $\mathbf{t}_n$ de la matriz **T** son las puntuaciones correspondientes a la nueva variable, sin significado físico, las cuales son expresadas como combinaciones lineales de las variables originales $\mathbf{z}_k$ agrupadas en **Z**. Una forma de obtener la base **P** es a través de una descomposición en valores singulares (SVD) de **Z** (Strang y cols., 2016). Otra forma es mediante una descomposición en eigenvalores de la matriz de covarianza (EVD). La EVD se computa en menos tiempo que la SVD cuando el número de observaciones *m*, supera el número de variables *n*, pero es menos precisa porque el número de condición de **S** es el cuadrado del número de condición de**Z**. La matriz de covarianza de **Z** denominada **S** se calcula a partir de la siguiente ecuación.

$$\mathbf{S} = \text{cov}(\mathbf{Z}) = \frac{1}{m-1}(\mathbf{Z}^{\top}\mathbf{Z}). \tag{2.11}$$

Posteriormente, y para este trabajo en particular, se realiza una eigen-descomposición de **S**, dado que requiere de menor carga computacional. Con lo que se obtiene la expresión factorizada de la matriz de covarianza.

$$\mathbf{S} = \mathbf{P}\Lambda\mathbf{P}^{\top}; \tag{2.12}$$

donde $\Lambda \in \mathbb{R}^{n\times n}$ es una matriz diagonal con sus eigenvalores ordenados en orden decremental y tal que:

$$\Lambda = \text{diag}(\begin{bmatrix}\lambda_1 & \lambda_2 & ...\lambda_n\end{bmatrix}), \quad \lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n \geq 0. \tag{2.13}$$

Finalmente, **P** es la matriz formada con los eigenvectores ortonormales donde $\mathbf{p}_k$ es el eigenvector correspondiente a λ_k.. Puede demostrarse que λ_k. es igual a la varianza de la k-ésima Componente principal (Jolliffe, 2011):

$$\lambda_k = \text{var}(\mathbf{t_k}) = \frac{1}{m-1}\mathbf{t_k}^{\top}\mathbf{t_k} \tag{2.14}$$

De modo que la fracción de la varianza total explicada por las primeras q Componentes Principales están dadas por:

$$\text{TEV}(q) = \frac{\sum_{k=1}^{q}\lambda_k}{\sum_{k=1}^{n}\lambda_k}, \tag{2.15}$$

cuyo denominador es n cuando se usan datos normalizados para la varianza unitaria.

Si una λ_k es cero su CP (componente Principal) correspondiente es descartable, pues no contribuye en nada para explicar la variabilidad de los datos. Incluso se pueden descartar las componentes con valores pequeños distintos de cero. El número de CP que si son significativas para explicar la variabilidad de los datos puede determinarse por validación cruzada (Wold, 1978) y por otros métodos (Tamura y Tsujita, 2007), o bien fijando de manera arbitraria el porcentaje de la varianza que las CP deban explicar. Cuando se determina el q número de Componentes principales significativas, es posible expresar que $\mathbf{P} = [\tilde{\mathbf{P}}\ \bar{\mathbf{P}}]$ donde $\tilde{\mathbf{P}}$=[$\mathbf{p_1}$ $\mathbf{p_2}$ ... $\mathbf{p_q}$] $\in \mathbb{R}^{n\times q}$ denominada "matriz de carga"genera el *subespacio principal* que contiene la variabilidad esencial del proceso, mientras que $\bar{\mathbf{P}}$=[$\mathbf{p_{q+1}}$ $\mathbf{p_{q+2}}$ ... $\mathbf{p_{q+3}}$] $\in \mathbb{R}^{n\times n-q}$ genera

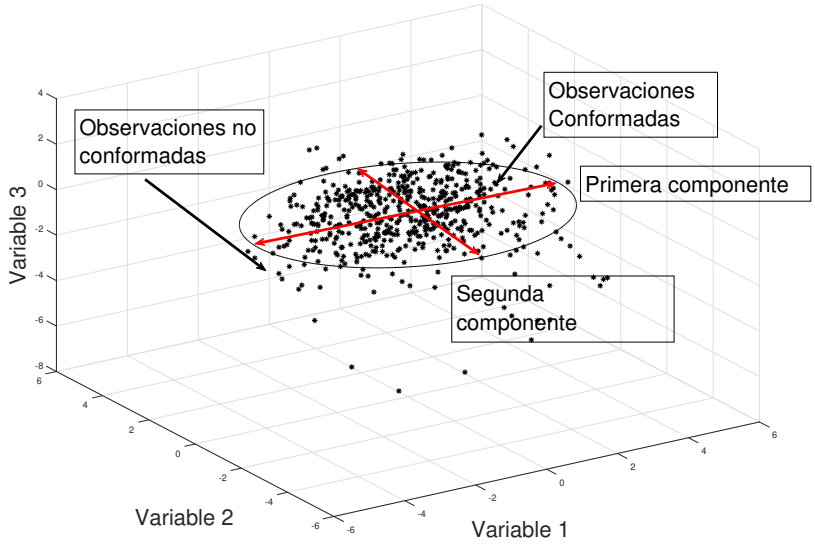


Figura 2.6: Vista isométrica del esquema general del análisis PCA.

el *subespacio residual* asociado al ruido y a las variaciones débilmente correlacionadas con la física del proceso. Análogamente, es posible hacer una separación en la matriz de las nuevas coordenadas, $\mathbf{T} = [\bar{\mathbf{T}}\ \tilde{\mathbf{T}}]$ donde $\bar{\mathbf{T}}$ contiene las coordenadas de las proyecciones de los datos en el subespacio principal y $\tilde{\mathbf{T}}$ contiene las proyecciones del subespacio residual. Puesto que los subespacios principal y residual son ortogonales, la información que contiene uno no lo contiene el otro, de manera que se complementan:

$$\mathbf{Z} = \begin{bmatrix} \bar{\mathbf{T}} & \tilde{\mathbf{T}} \end{bmatrix} \begin{bmatrix} \bar{\mathbf{P}} & \tilde{\mathbf{P}} \end{bmatrix}^{\top} = \bar{\mathbf{T}}\bar{\mathbf{P}}^{\top} + \tilde{\mathbf{T}}\tilde{\mathbf{P}}^{\top} = \bar{\mathbf{Z}} + \tilde{\mathbf{Z}} \tag{2.16}$$

Puede considerarse que las matrices $\mathbf{P}$ y Λ junto con los vectores de medias $\mu = [\mu_1, \mu_2, ...\mu_n]$ y que las varianzas $\sigma = [\sigma_1, \sigma_2, ...\sigma_n]$ usadas en la estandarización constituyen un modelo data-driven del proceso.

A continuación se describe cómo utilizar este modelo para detectar fallas.

Con el modelo PCA definido por $\mathbf{P}, \Lambda, \mu, \sigma$ es posible inferir si cualquier nueva medición $\mathbf{x_{new}} = \begin{bmatrix} x_1, & x_2, & ..., & x_n, \end{bmatrix}^{\top}$ es compatible con el"comportamiento normal"del proceso. Para ello se estandariza $\mathbf{x_{new}}$ con las medias y las desviaciones estándar del modelo. Luego, la muestra estandarizada $\mathbf{z_{new}}$ puede ser expresada en términos de la base ortonormal $\mathbf{P}$ con las

coordenadas:

$$\mathbf{t_{new}} = \mathbf{z_{new}}^{\top}\mathbf{P}. \tag{2.17}$$

Las proyecciones de la muestra $\mathbf{z_{new}}$ sobre los subespacios principal y residual están dadas por:

$$\bar{\mathbf{z}}_{\mathbf{new}} = \overline{\mathbf{PP}}^{\top}\mathbf{z_{new}}. \tag{2.18}$$

$$\widetilde{\mathbf{z}}_{\mathbf{new}} = \widetilde{\mathbf{PP}}^{\top}\boldsymbol{z_{new}} = [\mathbf{I} - \overline{\mathbf{PP}}^{\top}]\mathbf{z_{new}}. \tag{2.19}$$

De modo que $\mathbf{z_{new}} = \widetilde{\mathbf{z}}_{\mathbf{new}} + \bar{\mathbf{z}}_{\mathbf{new}}$. Para determinar la existencia de fallas no es necesario comparar la muestra actual $\mathbf{z_{new}}$ con todos los datos de entrenamiento en $\mathbf{Z}$ (ni con su versión transformada $\mathbf{T}$), porque pueden utilizarse estadísticos de prueba. Los dos estadísticos más usados para ello son el T^2 de Hotelling y el SPE (cuadrado del error de predicción):

$$T^2(\mathbf{z_{new}}) = \sum_{k=1}^{q}\frac{t_k^2}{\lambda_k} = \mathbf{t_q}^{\top}\Lambda_{\mathbf{q}}\mathbf{t_q} = \mathbf{z_{new}}^{\top}\overline{\mathbf{P}}\Lambda_q{}^{-1}\overline{\mathbf{P}}^{\top}\mathbf{z_{new}}, \tag{2.20}$$

$$SPE(\mathbf{z_{new}}) = \sum_{k=1}^{n}\tilde{z}_k^2 = \|\widetilde{\mathbf{z}}_{\mathbf{new}}\|^2 = \mathbf{z_{new}}^{\top}\widetilde{\mathbf{P}}\widetilde{\mathbf{P}}^{\top}\mathbf{z_{new}} = \mathbf{z_{new}}^{\top}(\mathbf{I} - \overline{\mathbf{PP}}^{\top})(\mathbf{z_{new}}) \tag{2.21}$$

DondeΛ_q =diag($[\lambda_1, \lambda_2, ..., \lambda_n]$) contiene solo los eigenvalores asociados al subespacio principal. El argumento $\mathbf{z_{new}}$ en 2.20 y 2.21 indica que T^2 y SPE se calculan para la muestra de prueba, aunque también es posible calcularlos para cada renglón de la matriz de entrenamiento $\mathbf{Z}$, pero estos pueden ser calculados mediante distribución de probabilidad sin necesidad de un cálculo exhaustivo. Cuando el proceso opera en condiciones nominales, los estadísticos T^2 y SPE (también llamados residuales) tienen valores pequeños y se infiere la existencia de fallas cuando sobrepasan determinados umbrales U_{T^2} (Fig. 2.7) y U_{SPE}, respectivamente.

En (Russell y cols., 2012), se establece U_{T^2} como:

$$U_{T^2} = \frac{q(m^2-1)}{m(m-q)}F_\alpha(q, m-q). \tag{2.22}$$

Donde $F_\alpha(q, m-q)$ es el punto crítico superior de $100\alpha\,\%$ en la distribución F de Fisher con q y $m-q$ grados de libertad. Generalmente $0.95 \leq \alpha \leq 0.99$ según la confianza y sensibi-

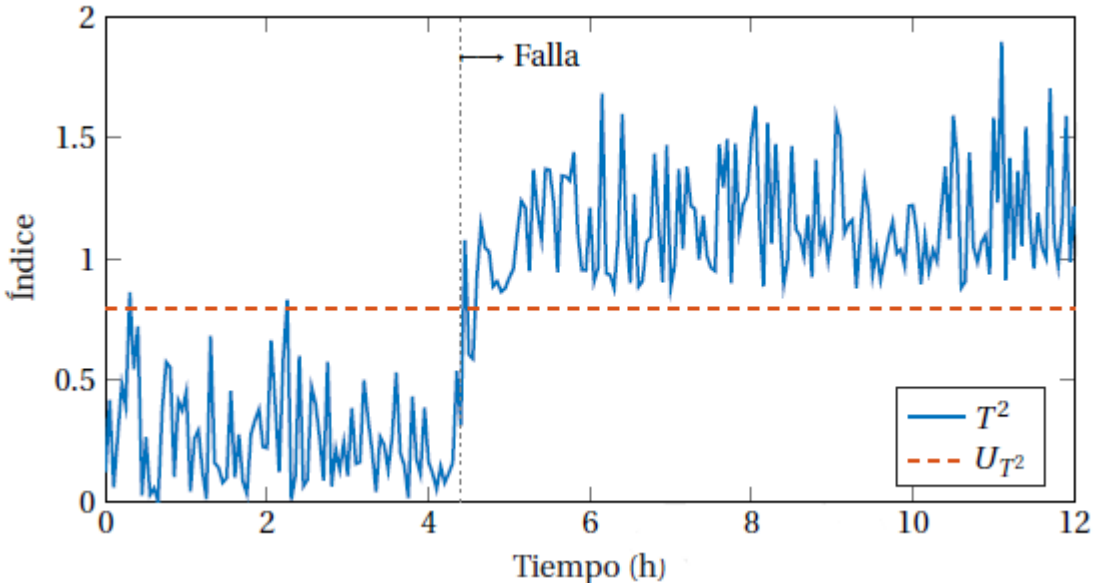


Figura 2.7: Monitoreo de procesos mediante el índice T^2. Se infiere una falla cuando $T^2 > U_{T^2}$, aunque el umbral se sobrepasa ocasionalmente durante la operación normal.

lidad deseadas. Si el número de muestras de entrenamiento m es grande puede usarse la aproximación de Qin (2012)

$$U_{T^2} \approx \chi^2_\alpha(q) \tag{2.23}$$

que corresponde a la distribución chi cuadrada de Pearson con nivel de confianza α y q grados de libertad. En términos geométricos, considerando el caso bidimensional, la región definida por la desigualdad $T^2(\mathbf{z}) \leq U_{T^2}$ es una elipse que contiene, con cierto nivel de confianza, a los puntos compatibles con la operación normal del proceso, mientras que en el exterior de la elipse se ubican los puntos asociados con alguna condición de falla. Se puede verificar que la región de confianza para T^2 tiene geometría elipsoidal partiendo de la definición de T^2 en 2.20. Así en el caso bidimensional, la región de confianza en el plano (t_1, t_2) es una elipse, porque la frontera de dicha región (donde $T^2 = U_{T^2}$ queda definida por la ecuación $t_1^2/\lambda_1 + t_2^2/\lambda_2 = U_{T^2}$.

Con respecto al umbral del índice SPE, también existen algunas propuestas basadas en la

distribución chi cuadrada por ejemplo en Nomikos y MacGregor (1995) se sugiere usar:

$$U_{SPE} = (v/2a)\chi^2_\alpha(2a^2/v) \tag{2.24}$$

donde a y v son la media y la varianza, respectivamente, de los índices SPE del conjunto de entrenamiento.Partiendo de 2.21 puede deducirse que la región de confianza para SPE tiene geometría esférica. Así para el caso bidimensional, la frontera de la región de confianza del plano $(\widetilde{z}_1, \widetilde{z}_2)$ es un círculo con ecuación $\widetilde{z}_1{}^2 + \widetilde{z}_2{}^2 = U_{SPE}$

Aunque tanto el índice SPE como el T^2 se utilizan para monitorear procesos, cada uno mide diferentes aspectos y sus role en el monitoreo no son simétricos. El índice SPE mide la variabilidad que rompe la correlación normal del proceso, lo que a menudo indica una situación anormal, mientras que el índice T^2 mide la distancia al origen del subespacio principal. La “región normal"definida por el límite de control U_{T^2} suele ser mucho mayor que la delimitada por U_{SPE}.Por lo tanto, generalmente se necesita una magnitud de falla grande para exceder el límite de control U_{T^2}. En cambio la región normal definida por el límite de control U_{SPE} solo incluye componentes residuales que son principalmente ruido. Por lo tanto las fallas con magnitudes pequeñas a moderadas pueden exceder fácilmente ese límite. Además cuando una muestra solo excede el límite U_{T^2} pero no sobrepasa el límite U_{SPE}, no rompe la estructura de correlación sino que simplemente se aleja del origen en el subespacio principal. Este caso podría ser una falla, pero también podría ser solo un cambio en la región de operación, lo que no necesariamente es una falla.

En Qin (2012) refiere que los límites de T^2 no son confiables cuando los datos del proceso no siguen los supuestos de una distribución normal multivariada. En estos casos, resulta más conveniente el monitoreo de procesos mediante SPE. Dado que la información recogida por T^2 y SPE no se duplica, sino que se complementa. En Qin (2009) se propone el monitoreo para detección de fallas mediante el uso combinado de ambos índices.

Tabla 2.1: Parámetros estadísticos de la matriz **X** donde $0.27 \leq \zeta \leq 0.33$ y $3 \leq \omega_n \leq 4$.

Dato	**Media**	**Desv. Est.**	**Varianza**
tr	0.5674	0.0577	0.0033
tp	0.9503	0.0953	0.0091
Mp	0.3730	0.0274	0.0007
ts	3.8635	0.4680	0.2190

2.3.1. Ejemplo de aplicación del PCA

El siguiente ejemplo de aplicación del PCA se muestra en Mina y Verde (2004), donde se considera como caso de estudio un sistema canónico de segundo orden:

$$\frac{Y(s)}{X(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{2.25}$$

Se asume que las variaciones normales de los parámetros son de $0.27 \leq \zeta \leq 0.33$ y de $3 \leq \omega_n \leq 4$. Las características consideradas de la respuesta al escalón son: tiempo de levantamiento *tr*, tiempo pico *tp*, sobrepaso *Mp* y tiempo de asentamiento *ts*, lo que resulta la matriz $\mathbf{X} = [tr \quad tp \quad Mp \quad ts]$. Las características en **X** se extraen de las respuestas al escalón que resultan de 35 combinaciones para los valores de ζ y ω_n dentro del rango de variaciones normales. En la Tabla 2.1 se presentan los parámetros de la media, desviación estándar y varianza de **X**.

Aplicando el procedimiento descrito en la sección anterior se escalan los vectores de datos y se forma la matriz de puntuaciones estándar de **X** denominada **Z**. Posteriormente se obtiene la matriz de correlación **S** de **Z**.

$$\mathbf{S} = \begin{bmatrix} 1 & 0.9937 & -0.1774 & 0.7122 \\ 0.9937 & 1 & -0.0667 & 0.7862 \\ -0.1774 & -0.0667 & 1 & 0.5609 \\ 0.7122 & 0.7862 & 0.5609 & 1 \end{bmatrix} \tag{2.26}$$

Como siguiente paso se proceden a calcular los eigenvalores con sus correspondientes eigenvectores, por lo que se obtiene:

$$\lambda_1 = 2.6383, \lambda_2 = 1.314, \lambda_3 = 0.0025, \lambda_4 = 0 \tag{2.27}$$

$$\mathbf{V} = \begin{bmatrix} \mathbf{v_1} & \mathbf{v_2} & \mathbf{v_3} & \mathbf{v_4} \end{bmatrix} = \begin{bmatrix} 0.5745 & -0.2945 & -0.3789 & 0.6631 \\ 0.5940 & -0.2011 & 0.2399 & -0.7411 \\ 0.1005 & 0.8601 & -0.5000 & 0.0091 \\ 0.5541 & 0.3648 & 0.7408 & 0.1052 \end{bmatrix} \tag{2.28}$$

de modo que la varianza total del conjunto de datos es:

$$\sum_{k=1}^{4} \lambda_k = 2.6383 + 1.314 + 0.0025 + 0 = 4 \tag{2.29}$$

Además puede demostrarse que la fracción de la varianza total explicada por q componentes principales, donde q representa para este ejercicio, las primeras 2 componentes principales está dada por:

$$\text{TEV}(q) = \frac{\sum_{k=1}^{q} \lambda_k}{\sum_{k=1}^{n} \lambda_k} = \frac{\sum_{k=1}^{2} \lambda_k}{\sum_{k=1}^{4} \lambda_k} = \frac{3.997}{4} \tag{2.30}$$

Es decir que con las dos primeras componentes se observa un 99.9 % de la variabilidad de los datos originales. Recordando de los datos de **Z** se deben expresar en términos una nueva base correspondiente a la matriz ortonormal, conformada por los dos primeros eigenvectores para este caso en particular. Para definir un umbral de los datos, es necesario aplicar la teoría estadística. La idea es mapear los datos transformados **T** a un conjunto univariado mediante el parámetro estadístico de Hotelling y a partir de estos seleccionar el umbral en estado normal en función de la variabilidad de los datos. Los datos transformados se obtiene a partir de

$$\mathbf{T} = \mathbf{ZP} \tag{2.31}$$

Recordando que Λ es una matriz diagonal que contiene los eigenvalores de **S** en orden decreciente de magnitud y **P** es la matriz con los eigenvectores $\mathbf{p}_k$ que constituyen la nueva base para los datos. Por lo que la matriz diagonal $\Lambda_\mathbf{q}$ contiene sólo los eigenvalores asociados al subespacio principal:

$$\Lambda_q = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} = \begin{bmatrix} 2.683 & 0 \\ 0 & 1.314 \end{bmatrix} \tag{2.32}$$

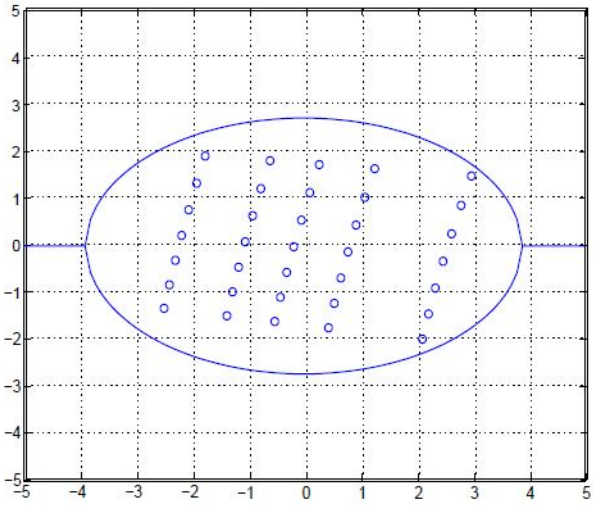


Figura 2.8: Región elíptica de los datos en estado normal (Mina y Verde, 2004)

A continuación se procede a calcular el estadístico T^2. El cual es una representación de la magnitud de cada uno de los datos transformados, z_k, de lo cual resulta un conjunto univariado de datos. Así es posible definir un límite de control, también denominado umbral de condición nominal con enfoque de procesos univariados. Asumiendo una distribución F con grado de confianza $\alpha = 0.99$, $q = 2$ y $m = 35$ el umbral o límite de control U_{T^2} para el ejercicio está dado por $U_{T^2} = 5.63$ La Figura 2.8 muestra los datos transformados sobre la base bidimensional, así como la región elíptica de control con α =0.99, que se define como el umbral para el comportamiento normal.

Para un nuevo dato de entrada x_{new} de $1xm$ ($m = 4$ para este caso) primero deberá ser estandarizado con la medias y las desviaciones estándar utilizadas en los datos de referencia. La muestra estandarizada z_{new} es mapeada sobre la nueva base donde se obtiene el nuevo vector. Para determinar el estado de esta muestra se calcula el parámetro estadístico T^2, así, si este valor resulta fuera del umbral U_{T^2} entonces se detecta una condición anormal.

Para la verificación del algoritmo se propone una serie de variaciones en los parámetros de ζ y ω_n, diferentes a las definidas en las condiciones nominales, donde se obtienen un nuevo conjunto de datos de prueba. La Tabla 2.2 muestra los intervalos de variación de los datos.

Con estos datos se lleva a cabo el análisis cuantitativo para la FDI. Dada la dimensión de los datos, es posible graficarlos conjuntamente para tener una apreciación de aquellos que se

Tabla 2.2: Intervalos de prueba generados con los valores de los parámetros ζ y ω_n dentro y fuera del rango de variación normal

ω_n	{2.6, 2.8, 4.2, 4.4}		{2.8,4.2}	{3.2, 3.8}
ζ	{0.23, 0.25}	{0.35,0.37}	[0.27, 0.33]	{0.25,0.35}
tr_p	[0.42, 0.72]	[0.46,0.80]	[0.45, 0.72]	[0.49,0.64]
tp_p	[0.73, 1.24]	[0.76, 1.30]	[0.77, 1.18]	[0.85, 1.04]
Mp_p	[0.44, 0.47]	[0.28, 0.30]	[0.33,0.41]	[0.30,0.44]
ts_p	[3.63, 6.68]	[2.45, 4.39]	[2.88, 5.29]	[3.0, 5.0]

salen de la región de estado nominal, como se muestra en la Figura 2.9.

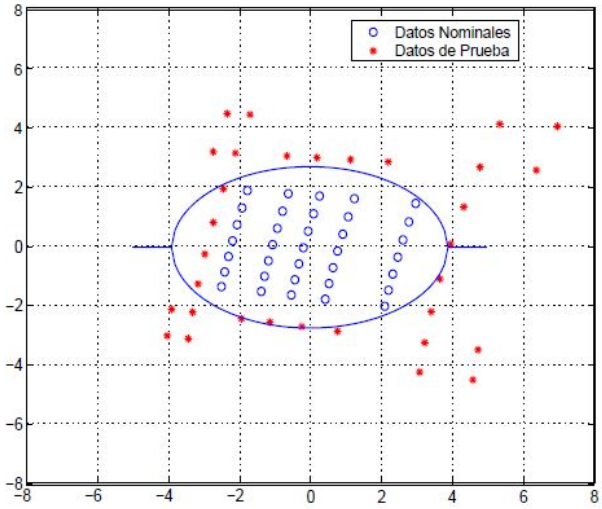


Figura 2.9: Datos de parámetros nominales y de prueba en el espacio bidimensional de componentes principales (Mina y Verde, 2004)

Se observa que existen 6 datos de prueba generados con parámetros anormales que caen dentro de la región nominal. Los que se encuentran en el límite de la región se podrían interpretar como falsas alarmas. Por otro lado, con base en el análisis del en términos del estadístico T^2, se puede decir que los seis casos satisfacen el umbral de estado nominal, lo cual valida una correcta clasificación. Mina y Verde (2004) concluyen el análisis del procedimiento, generando las respuestas al escalón de estos seis casos. Dichas respuestas se muestran en la Figura 2.10, sobrepuestas al patrón gráfico en condiciones normales. Desde el punto de vista de patrón de señales, estos seis casos se mantienen dentro del patrón nominal.

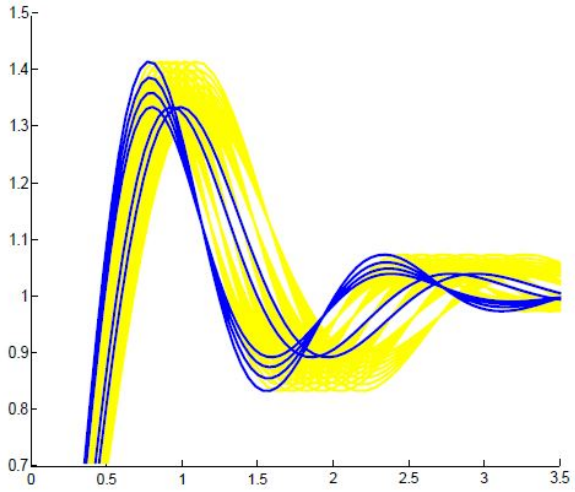


Figura 2.10: Respuesta al escalón de 6 condiciones de prueba sobrepuestas a la región nominal (Mina y Verde, 2004)

2.4. Aprendizaje de máquina

Los enfoques basados en modelos son útiles en sistemas, donde se suponen, existen modelos matemáticos precisos. En la actualidad, no se tiene un modelo, que nos ayude con exactitud a la FDI de los VANT, debido a que son sistemas no lineales y variantes en el tiempo. Sin embargo, el uso de los VANT en la actualidad para la realización de tareas, presenta un aumento abrupto. La variedad de vehículos de este tipo, el gasto de prácticas de modelo que representen su dinámica con exactitud, la dificultad para modelar dicha dinámica ante presencia de fallas, requieren enfoques desafiantes que permitan resolver el problema que se presenta para la FDI. En contraparte, la mayor eficiencia de los sensores a bordo, el aumento de las capacidades de computacionales de los procesadores y de los pilotos automáticos, y los avances de las técnicas de aprendizaje de máquina, pueden ofrecer soluciones eficientes para la detección de fallas.

En los métodos basados en datos, no es necesario un conocimiento absoluto sobre la dinámica interna del sistema. La única fuente de información de su comportamiento, son los datos disponibles de los sensores inmersos en dicho sistema. Como parte de la necesidad de proponer métodos basados en datos que sean más eficientes al momento de hacer un diagnóstico de falla, se han incorporado sistemas de aprendizaje de máquina para su trabajo

en conjunto ante el problema de el diagnóstico de fallas.

El aprendizaje de máquina consiste básicamente en automatizar, a través de diversos algoritmos, la identificación de patrones o tendencias en los bancos de datos, que ayuden a realizar el diagnóstico en sistemas de identificación y clasificación de fallas. Es por ello que resulta importante la elección del algoritmo más adecuado y también, el hecho de disponer de un gran volumen de datos de suficiente calidad.

Los tipos de aprendizaje de máquina pueden clasificarse básicamente en dos categorías diferentes: aprendizaje no supervisado y aprendizaje supervisado. En López Bautista (2018) se da una explicación de estos conceptos, y se clasifican a los algoritmos, de acuerdo al tipo de aprendizaje como se muestra en la Fig. 2.11 El aprendizaje no supervisado tiene lugar cuando no se disponen de datos etiquetados para el entrenamiento. Sólo se conocen los datos de entrada, pero no existen datos de salida que correspondan a alguna clase. Por lo tanto, solo es posible describir la estructura de los datos para intentar conseguir algún tipo de organización que simplifique el análisis. Es por ello que se dice que tienen un carácter de tipo exploratorio. Los tipos de algoritmos utilizados en el aprendizaje no supervisado son: el análisis en componentes principales, la descomposición en valores singulares y algoritmos de clustering.

En el aprendizaje supervisado los algoritmos trabajan con datos etiquetados, que intentan encontrar una función que, dadas las variables de entrada, les asigne una etiqueta de salida adecuada. El algoritmo se entrena con un histórico de datos y así realiza el aprendizaje para asignar una salida adecuada ante un nuevo valor. Los algoritmos habituales son: Los árboles de decisión, la clasificación de Naive Bayes, la regresión por mínimos cuadrados, la regresión logística y las máquinas de vectores de soporte. En el aprendizaje de máquina supervisado, es necesario etiquetar previamente los casos de entrenamiento, que generalmente se utiliza para la inferencia de fallas, centrada en los datos. En el caso de que suceda una falla que no se encuentre etiquetada, el resultado de la clasificación de dicha falla, se espera como una distribución de probabilidad de los modos normales disponibles, etiquetas de fallas identificadas, y una falla desconocida probable (Baskaya y cols., 2017).

Dentro de lo métodos de aprendizaje de máquina para la FDI, se encuentran las Redes Neuronales Artificiales(Schlechtingen y Santos, 2011) y las Máquinas de Vectores de Soporte

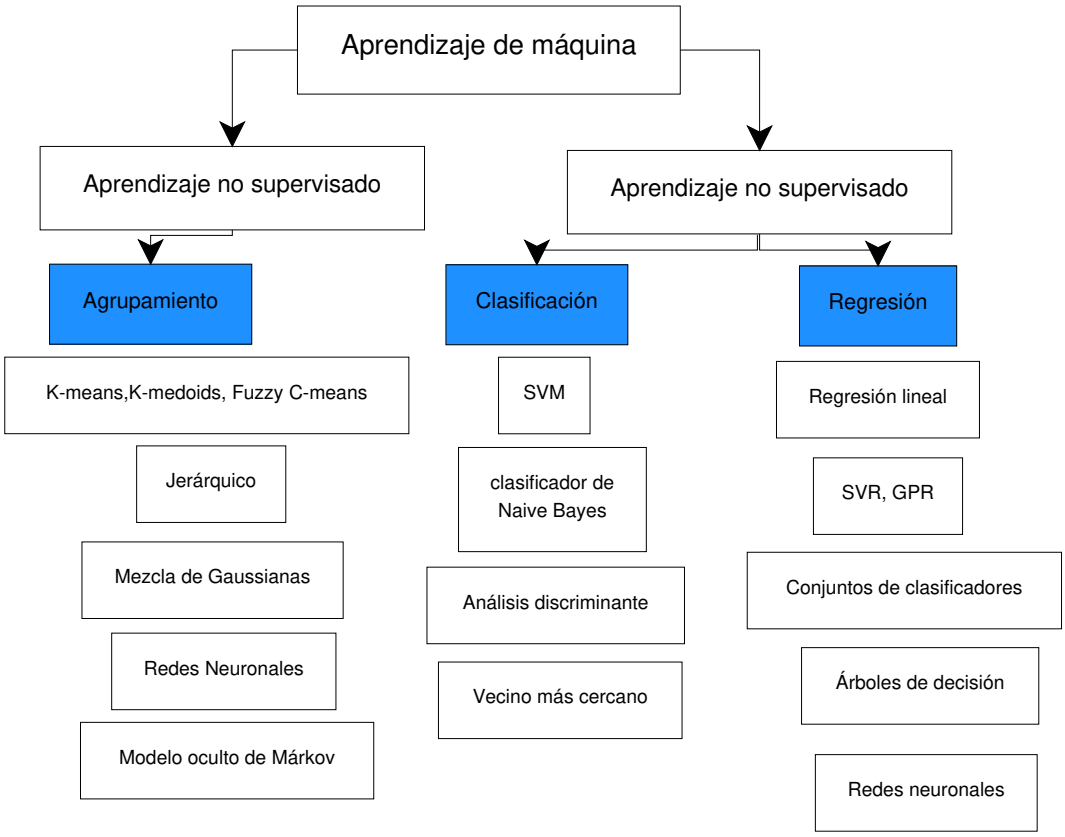


Figura 2.11: Tipos de aprendizaje de máquina

(Betancourt, 2005), en GUI y LIU (2002) también se argumentan como métodos de inteligencia artificial para la detección de fallas en sistemas complejos. La comparación entre otros métodos de aprendizaje de máquina, como el PCA, y métodos basados en modelos, como el enfoque de espacios de paridad estocástica se da en Hagenblad y cols. (2003). En (Li y cols., 2016), se argumenta que el Análisis en Componentes Principales Dinámico (*DPCA*), ya que el control del VANT es un sistema dinámico, permite reflejar perturbaciones desconocidas, mientras que los enfoques basados en modelos identificaban únicamente fallas conocidas previamente.

La propiedad de generalización de una máquina de aprendizaje, es decir su capacidad para emitir una respuesta correcta ante una nueva entrada semejante a aquellas con las que ha sido entrenada, es la característica principal que se busca en los sistemas conexionistas supervisados y sirve de justificación en la elección de los principios inductivos y el tipo de estructuras de aprendizaje para elaborar el presente trabajo.

2.5. Algoritmo de clasificación *k-NN*

El algoritmo k-NN es una extensión del método del vecino más cercano (por sus siglas en inglés NN, *Nearest Neighbor*). En este método se recolecta una serie de muestras etiquetadas que contienen mediciones de algunas características (del inglés *"features"*) de los objetos o eventos que se desea clasificar. Luego, para clasificar nuevas muestras se usan medidas de distancia para identificar cual es la muestra del conjunto inicial,denominado conjunto de entrenamiento, que más se parece a la muestra bajo prueba (la más cercana, en el espacio de características), y se asume que dada su similitud las dos muestras pertenecen a la misma clase (Santos-Ruiz y cols., 2019). El k-NN también se basa en esa suposición, pero considera un mayor número de vecinos, lo cual permite una clasificación más robusta, menos sensible a los valores atípicos y al ruido de medición.

En el contexto del aprendizaje automático (del inglés, *"machine learning"*), la clasificación por k-NN es un método de aprendizaje supervisado de tipo "perezoso" que sólo requiere un mínimo esfuerzo en la etapa de entrenamiento y difiere todo el cómputo para la etapa de clasificación. Esto quiere decir que inicialmente el k-NN no generaliza más allá de los datos de entrenamiento, y pospone esta acción hasta que el sistema deba clasificar nuevos datos de entrada (Santos-Ruiz y cols., 2019).

Dado un conjunto de entrenamiento de $m \times n$, donde m es el numero de muestras y n las características del sistema. La i-ésima muestra de una vector de $\mathbb{R}^n$ de la forma $\mathbf{x}_i = (x1, x2, ...x_n)^\top$ junto con una etiqueta que expresa su pertenencia a una de las q clases. Generalmente las muestras $\mathbf{x}_i$ se agrupan en una matriz $\mathbf{X} \in \mathbb{R}^{m \times n}$. Para las etiquetas de las clases se usan enteros positivos agrupados en un vector $\mathbf{y} \in \mathbb{R}^m$, donde $\mathbf{y}_i$ es la clase de la muestra $\mathbf{x}_i$. Se asume que las n características relevantes para la clasificación de las muestras han sido determinadas previamente mediante un proceso de selección o extracción. La fase de entrenamiento del algoritmo k-NN se limita a almacenar los vectores de características y las etiquetas de las clases. Posteriormente, en la fase de clasificación el algoritmo k-NN recibe nuevas muestras de clase desconocida y debe asignarles una de las q clases de los datos de entrenamiento, mediante algún proceso de inferencia. El algoritmo k-NN puede enmarcarse dentro de la teoría de decisión bayesiana, de modo que la clasificación de las nuevas observaciones se basa

en hallar la clase con la mayor probabilidad a *posteriori* $P(c_j \mid \mathbf{x})$, $j = 1, 2, ..., q$, donde $\mathbf{x}$ es el vector de características de la muestra a clasificar y c_j representa la j-ésima clase. Mediante el teorema de Bayes, se demuestra en Martinez y Martinez (2015), que la probabilidad a *posteriori* de cada clase está dada por

$$P(c_j \mid \mathbf{x}) = z = \frac{k_j}{k} \tag{2.33}$$

Donde k_j es el número de muestras del conjunto de entrenamiento que pertenecen a la j-ésima clase, considerando que sólo las k muestras más cercanas a la muestra bajo prueba. Luego, la clase asignada por el clasificador se determina hallando la máxima probabilidad a *posteriori*:

$$\widehat{y}(x) = \underset{j}{\operatorname{argmax}} P(c_j \mid \mathbf{x}) \tag{2.34}$$

La salida $\widehat{y}(\mathbf{x})$ del algoritmo k-NN es una estimación de la clase verdadera $y(\mathbf{x})$, la cual, considerando la ecuación (2.34), es el valor de y con la mayor frecuencia relativa entre los k vecinos más cercanos a $\mathbf{x}$.

Para cuantificar la similitud entre dos muestras basta con medir su cercanía en el espacio de características, lo cual está determinado por la distancia entre los puntos que las representan. Bajo el enfoque euclidiano, la distancia entre dos muestras x y x' está determinado por:

$$d(\mathbf{x}, \mathbf{x}') = \sqrt{(x_1 - x_1')^2 + (x_2 - x_2')^2 + ... + (x_n - x_n')^2} = \sqrt{(x - x')^\top (x - x')} \tag{2.35}$$

La región que encierra a las k muestras más cercanas al dato de prueba es una n-esfera (hiperesfera) en el espacio de características. En el caso bidimensional esta región es una circunferencia. En términos geométricos, el algoritmo k-NN consiste en calcular la densidad de puntos correspondiente a cada clase en la hiperesfera y seleccionar la clase con la mayor densidad. A pesar de que lo más frecuente para medir la similitud entre muestras es mediante los conceptos de distancia y norma usados en los espacios métricos, la medida usada para cuantificar la similitud no tiene que ser forzosamente una distancia y, en caso de serlo, esta no tiene por qué ser euclidiana. En los casos donde la similitud entre muestras no se mide mediante la distancia euclidiana, la región que encierra a los k vecinos más cercanos no tiene geometría esférica.

La ecuación (2.35) que representa la distancia euclidiana es un caso particular de la distancia de Minkowski, que se presenta a continuación:

$$dp(x,x') = \sqrt[p]{\sum_{j=1}^{n} \left|(x_j - x'_j)\right|^p} \tag{2.36}$$

Otras definiciones de distancia que particularizan (2.36) son la distancia Manhattan (p = 1) y la distancia de Chebychev (p = 1). Cualquiera de estas métricas de distancia puede ser usada para darle sentido a la expresión “vecino más cercano". Un aspecto a considerar cuando se usan la distancia euclidiana y otras distancias de Minkowski para medir la similitud entre muestras es que cada componente x_j del vector de características contribuye por igual al cuantificar la distancia. Esto puede dificultar la clasificación cuando existen algunas características x_j que son mucho más importantes que otras, o bien cuando se tienen muchas características irrelevantes. Por ejemplo, una característica relevante sería dominada por diez irrelevantes, de modo que resulta sensato asignar factores de ponderación a la contribución de $\left|(x_j - x'_j)\right|$ en el cálculo de la distancia. La necesidad de aplicar un escalamiento a cada término $\left|(x_j - x'_j)\right|$ en (2.36) es más evidente cuando las características x_j tienen diferentes escalas o unidades de medida. Una forma de escalamiento, que asigna las ponderaciones según la variabilidad de cada característica, es la distancia de Mahalanobis:

$$d_M(\mathbf{x},\mathbf{x}') = \sqrt{(x-x')^\top \mathbf{S}^{-1}(x-x')} \tag{2.37}$$

donde **S** es la matriz de covarianza de los datos, de modo que $s_{i,j}$= cov(x_i, x_j). Sin la ponderación **S** la distancia de Mahalanobis en (2.37) es la distancia euclidiana en (2.35).

Respecto a la elección de el k número de vecinos a utilizar, esta se hace basándose en los datos disponibles, optimizando alguna medida del error de clasificación. Una forma de medir este error es mediante el índice de pérdidas:

$$loss = 1 - \frac{m_s}{m} \tag{2.38}$$

donde m es el número total de muestras en los datos de prueba, y m_s es el número de estas

muestras que resultan correctamente clasificadas con un valor dado de k. Para mejorar la robustez del clasificador, en la determinación del valor óptimo de k se usa validación cruzada iterativa (*K-fold cross validation*), de modo que el error de clasificación se calcula para un conjunto de muestras distinto al de entrenamiento. Por lo general, los valores pequeños de k (i.e.$k \to 1$) producen clasificadores muy sensibles al ruido de medición y a otras incertidumbres en el vector de características.

2.5.1. Ejemplo de clasificación con *k-NN*

En la Fig. 2.12 se muestra una clasificación entre dos clases denominadas *versicolor* y *virginica* usando los 8 vecinos más cercanos. Se puede observar que 6 vecinos pertenecen a la clase *versicolor* y dos pertenecen a la clase *virginica*. De modo que $P(c_1 \mid x) = 0.75$ y $P(c_2 \mid x) = 0.25$. Por lo que se obtiene que la salida del algoritmo sería $\widehat{y}(x) = 2$ por lo que se deduce que el nuevo dato de entrada pertenece a la clase 2 denominada *versicolor*.

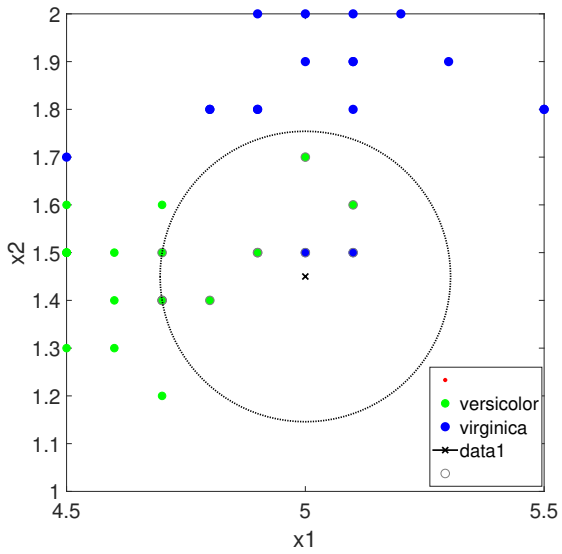


Figura 2.12: Clasificación 8-NN con distancia euclidiana, caso bidimensional.

2.6. Máquinas de vectores de soporte

El problema de clasificación puede restringirse a la consideración del problema de dos clases sin pérdida de generalidad. El objetivo es separar las dos clases por una función que se induce a partir de los ejemplos disponibles. El objetivo es producir un clasificación que Funcione bien en ejemplos que no sean perceptibles a simple vista, es decir, que se generalice bien.

En el ejemplo de la Fig.2.13 hay muchos clasificadores lineales que pueden separar los datos, pero solo hay una que maximiza el margen (maximiza la distancia entre él y el punto de datos más cercano de cada clase). Este clasificador lineal se denomina hiperplano se separación óptimo. Intuitivamente, esperaríamos que este límite se generalice bien en oposición a los otro límites posibles.

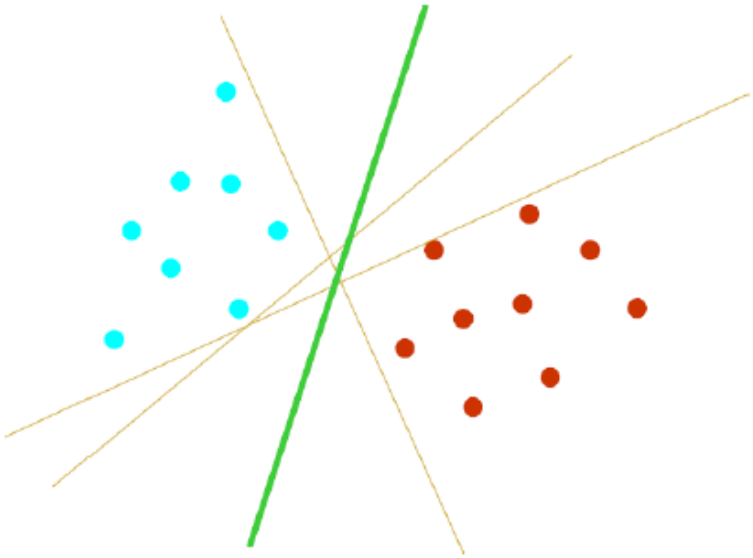


Figura 2.13: Hiperplano de separación óptima.

2.6.1. El hiperplano de separación óptimo

La teoría relacionada a las Máquinas de vectores de soporte (SVM, del inglés *Support Vector Machines*) es una técnica de clasificación de datos que se ha desarrollado también para resolver problemas en el área de la detección de fallas. Esta teoría se basa en el concepto

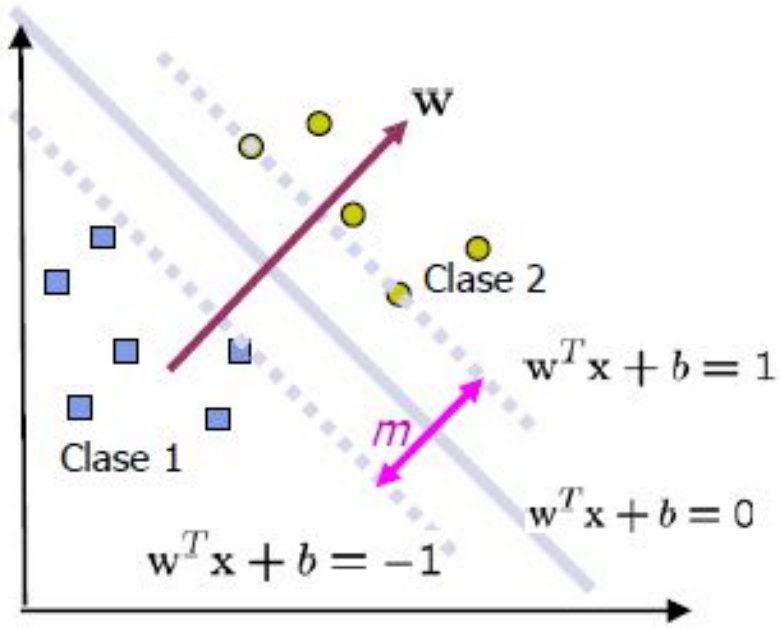


Figura 2.14: Margen m dividiendo a dos clases, con la mayor separación posible. (Betancourt, 2005)

de minimización de riesgo estructural, SRM (Vapnik, 1995). En ciertos casos, las SVM han obtenido un mejor desempeño que otros métodos de aprendizaje automatizado, como las redes neuronales artificiales y con un mejor potencial para la clasificación de datos (Burges, 1998). Una SVM primero mapea los puntos de entrada a un espacio de características mayor, es decir si los puntos dados están mapeados en un espacio de características de $\mathbb{R}^2$ entonces son mapeados a $\mathbb{R}^3$, y se encuentra un hiperplano que maximice el margen m entre las dos clases en el espacio (Fig. 2.14).

Se considera el problema de separar el conjunto de vectores de entrenamiento que consi-derados conjuntos de datos separados denominados clases,

$$D\{(x_1, y_1), ..., (x_l, y_l)\}, x \in \mathbb{R}, y \in \{-1, 1\} \tag{2.39}$$

En el hiperplano

$$< w, z > + b = 0 \tag{2.40}$$

Se dice que el conjunto de vectores está separado de manera óptima por el hiperplano si está separado sin error y la distancia entre el vector más cercano al hiperplano es máxima. Existe

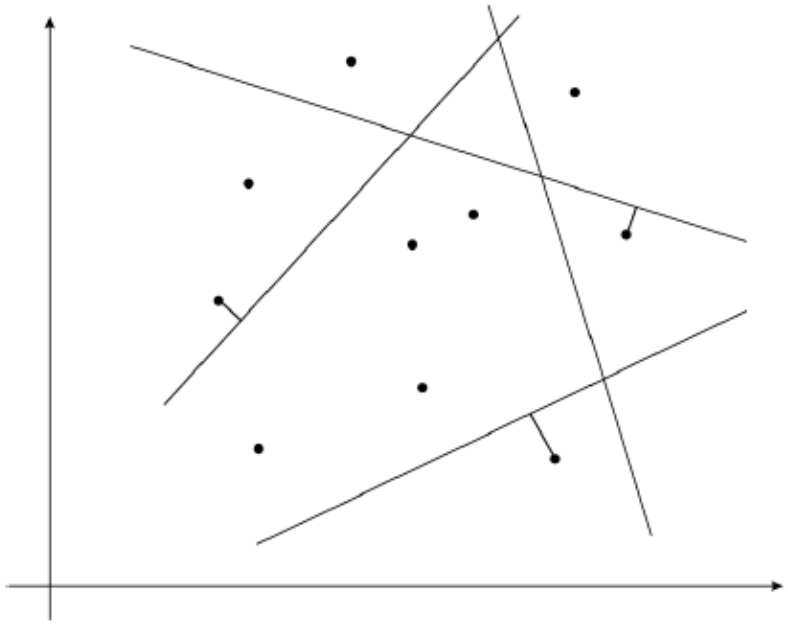


Figura 2.15: Hiperplano canónico

cierta redundancia en la Ecuación (2.40), y sin pérdida de generalidad es apropiado considerar un hiperplano canónico (Vapnik, 1995), donde los parámetros w, b están restringidos por,

$$\min_i |< w, z_i > +b| = 1 \tag{2.41}$$

Esta restricción incisiva en la parametrización es una alternativa preferible para simplificar la formulación del problema. Es decir, se establece que: la norma del vector de peso debe ser igual a la inversa de la distancia, del punto más cercano en el conjunto de datos al hiperplano. La idea se ilustra en la Figura 2.15, donde se muestra la distancia desde el punto más cercano a cada hiperplano. Un hiperplano de separación en forma canónica debe satisfacer las siguientes restricciones:

$$y_i = [< w, z_i > +b] \geq 1, i = 1, ..., l. \tag{2.42}$$

La distancia *d(w,b;z)* de un punto *x* desde el hiperplano *(w,b)* es,

$$d(w, b; z) = \frac{|< w, z_i > +b| = 1}{\| w \|} \tag{2.43}$$

El hiperplano óptimo se obtiene maximizando el margen ρ, sujeto a las restricciones de la ecuación (2.42). El margen viene dado por

$$\rho(w,b) = \frac{2}{\parallel w \parallel} \tag{2.44}$$

Por lo tanto, el hiperplano que separa de manera óptima los datos es el que minimiza

$$\Phi(w) = \frac{1}{2} \parallel w \parallel^2 \tag{2.45}$$

2.6.2. Conjunto linealmente separable

Dado el conjunto D de datos para la fase de entrenamiento, donde cada punto $x \in \mathbb{R}$, $y \in \{-1, 1\}$, lo que representa que a cada dato se le representa en alguna de las dos clases de y (Fig. 2.16). Una manera de encontrar un hiperplano adecuado es el de graficar el espacio de los datos de entrada en un espacio de una mayor dimensión. Sea $z = \varphi(x)$ la cual corresponde a el vector en el espacio de características. Se debe encontrar el hiperplano:

$$< w \cdot z > + b = 0 \tag{2.46}$$

de modo que sea posible separar el punto x_i de acuerdo a la función:

$$f(x_i) = sign(w \cdot z_i + b) = \begin{cases} 1 & \text{si } y_i = 1 \\ -1 & \text{si } y_i = -1 \end{cases} \tag{2.47}$$

Donde $w \in \mathbb{Z}$ y $b \in \mathbb{R}$. Se dice que el conjunto de datos de entrenamiento D es linealmente separable cuando se cumple:

$$\begin{cases} w \cdot z_i + b \geq 1 & \text{si } y_i = 1 \\ w \cdot z_i + b \leq 1 & \text{si } y_i = -1 \end{cases} \tag{2.48}$$

donde i=1,...,l. El hiperplano ótimo de separación en un conjunto D que es linealmente separable es único, y por tanto, el margen entre ambas clases es maximizado.

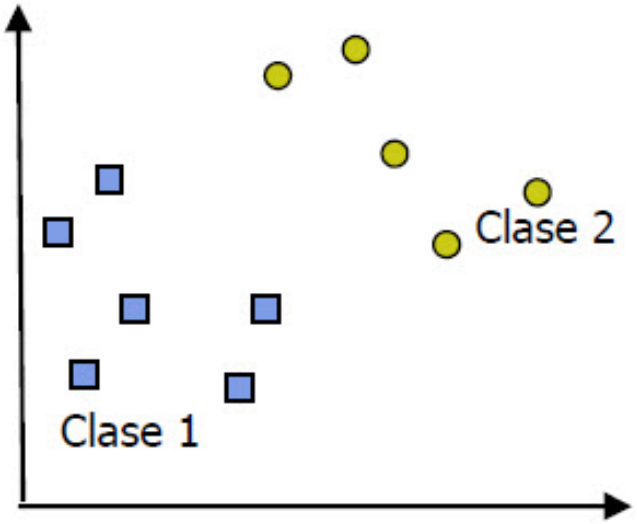


Figura 2.16: Ejemplo de datos linealmente separables (Betancourt, 2005)

2.6.3. Conjunto linealmente no separable

Cuando se presentan datos que son linealmente no separables, el algoritmo puede ser generalizado:

$$y_i(w_i \cdot z_i + b) \geq 1 - \xi_i; i = 1...l; \xi_i \geq 0 \tag{2.49}$$

Donde ξ_i son valores diferentes de 0 para los que el punto x_i no satisface. La solución para encontrar el hiperplano se define nuevamente como:

$$\text{mín}\left\{\frac{1}{2}w \cdot w + C\sum_{i=1}^{l}\xi_i\right\}, \tag{2.50}$$

donde C = constante y es un parámetro libre para el ajuste del margen de la SVM. El problema para encontrar el hiperplano en la ecuación anterior se resuelve a través de la solución del Lagrangiano y transformándolo al dual:

$$\text{máx}\, W(\alpha) = \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j z_i \cdot z_j \tag{2.51}$$

Dado que $\sum_{i=1}^{l} y_i\alpha_i = 0$, $0 \leq \alpha_i \leq C$, Donde $\alpha = \alpha_1...\alpha_l$ es un vector de multiplicadores de

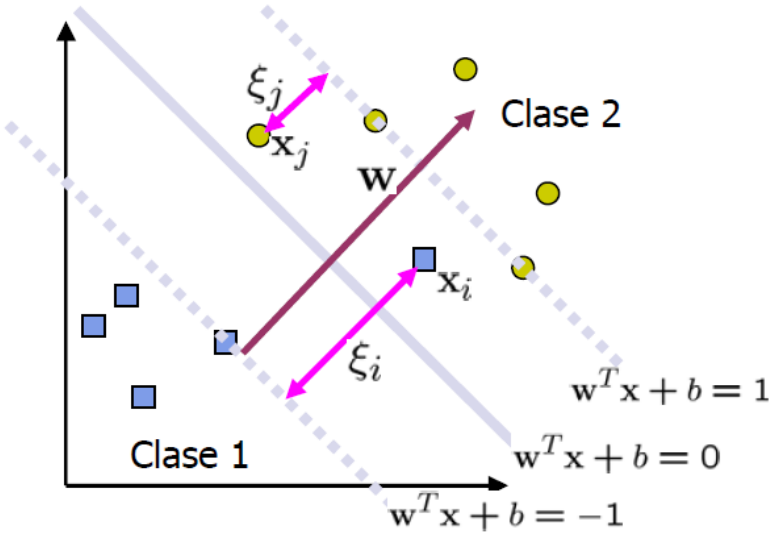


Figura 2.17: Ejemplo de datos linealmente no separables (Betancourt, 2005)

Lagrange positivos asociados con las constantes. De acuerdo al Teorema de Khun-Tucker (Betancourt, 2005) la solución $\bar{\alpha}_i$ al problema satisface:

$$\bar{\alpha}_i(y_i(\bar{w}\cdot z_i+\bar{b})-1+\bar{\xi}_i)=0, i=1,..,l \tag{2.52}$$

$$(C-\bar{\alpha}_i)\xi_i=0, i=1,..,l \tag{2.53}$$

El punto x_i correspondiente con $\bar{\alpha}_i \leq 0$ es el vector de soporte. Existen dos tipos de vectores de soporte en el conjunto linealmente no separable. Para $0<\bar{\alpha}_i<C$ el correspondiente vector de soporte x_i satisface las igualdades $y_i(\bar{w}\cdot z+\bar{b})=1$. Si $\bar{\alpha}_i=C$, donde $\xi_i \neq 0$ el correspondiente vector de soporte x_i no satisface a la ecuación (2.48). A estos vectores de soporte se le denominan vectores errores. El punto x_i correspondiente a $\bar{\alpha}_i$=0 se clasifica correctamente y como se observa en la Figura 2.17 , se encuentra alejado del borde de desición.

2.6.4. Kernel para conjunto linealmente no separable

Cuando no se conoce el vector en el espacio de características φ existe una propiedad de las Máquinas de Soporte Vectorial la cual es aplicable para este caso. Es necesario utilizar una

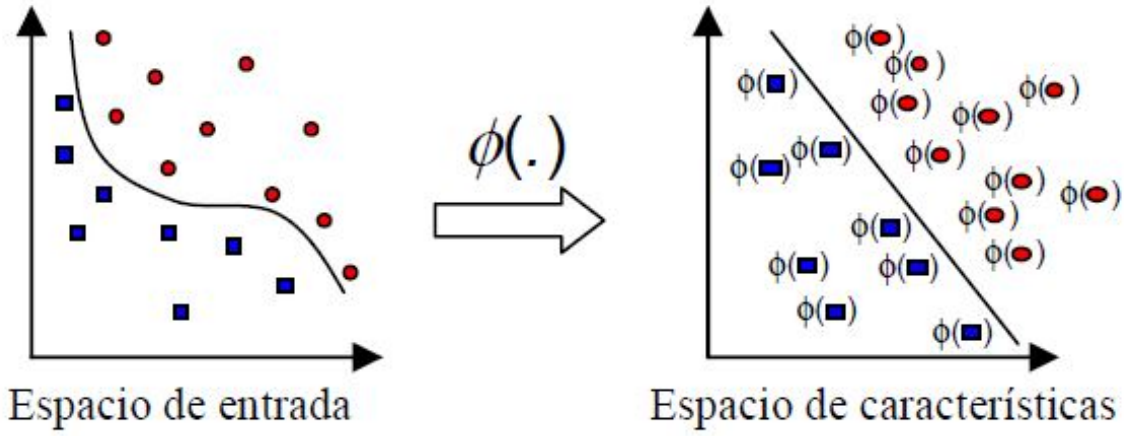


Figura 2.18: Kernel para transformación del espacio de los datos de entrada (Betancourt, 2005)

ecuación K, denominada también como **kernel** la cual realiza el cálculo del producto punto de los datos de entrada en el espacio de características Z (Fig. 2.18), lo que se representa en la siguiente ecuación:

$$z_i \cdot z_j = \varphi(x_i) \cdot \varphi(x_j) = K(x_i, x_j) \tag{2.54}$$

Existen funciones que satisfacen el teorema de Mercer y que pueden ser usados como kernels, dado que se puede aplicar el producto punto. Para este caso, es posible utilizar el kernel polinomial de grado d, que se muestra a continuación:

$$K(x_i, x_j) = (1 + x_i \cdot x_j)^d \tag{2.55}$$

Con el objetivo de poder realizar un clasificador SVM.
Para encontrar el hiperplano no lineal óptimo de separación de clases, se tiene que solucionar la siguiente ecuación:

$$Max.W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \tag{2.56}$$

Dado que $\sum_{i=1}^{l} y_i \alpha_i = 0$, $0 \leq \alpha_i \leq C$,e $i = 1, ..., l$.

Donde la función de decisión se define como:

$$f(x) = sign(w \cdot z + b) = sign[\sum_{i=1}^{l} \alpha_i y_i K(x_i, x_j) + b] \tag{2.57}$$

2.6.5. Ejemplo: caso linealmente separable

Supongamos que se tienen los siguientes puntos de datos etiquetados positivamente en $\mathbb{R}^2$:

$$\left\{\begin{pmatrix}3\\1\end{pmatrix}, \begin{pmatrix}3\\-1\end{pmatrix}, \begin{pmatrix}6\\1\end{pmatrix}, \begin{pmatrix}6\\-1\end{pmatrix}\right\}$$

y los siguientes datos, etiquetados negativamente en el espacio $\mathbb{R}^2$:

$$\left\{\begin{pmatrix}1\\0\end{pmatrix}, \begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}0\\-1\end{pmatrix}, \begin{pmatrix}-1\\0\end{pmatrix}\right\}$$

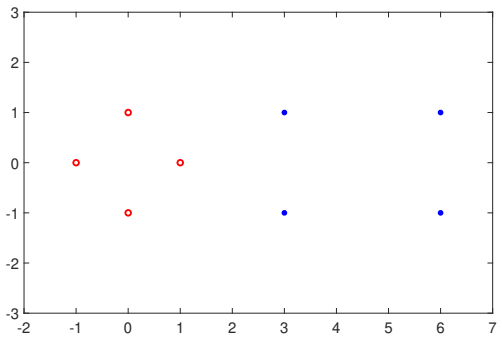


Figura 2.19: los puntos representados en el espacio $\mathbb{R}^2$
con azul los que representan al conjunto positivo y con rojo los que representan al conjunto negativo

La gráfica de ambos conjuntos en el plano se observa en la Figura 2.19 . Supongamos que se tiene la necesidad de encontrar un algoritmo *SVM* que sea capaz de clasificar con precisión ambas clases. Como los datos son linealmente separables, es posible utilizar un *SVM* lineal, es decir, una función de mapeo donde Φ es la función de identidad. A través de una simple inspección es posible determinar que se tienen 3 vectores de soporte (véase la Fig. 2.20) los cuales son:

$$\left\{ s_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, s_2 = \begin{pmatrix} 3 \\ 1 \end{pmatrix}, s_3 = \begin{pmatrix} 3 \\ -1 \end{pmatrix} \right\}$$

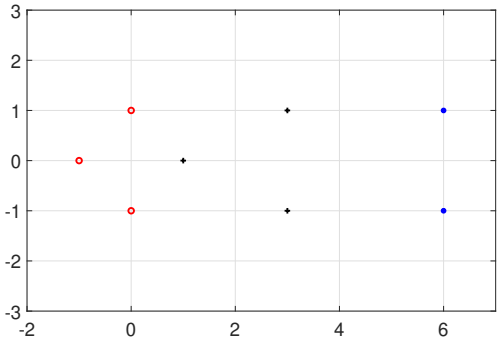


Figura 2.20: Los 3 puntos que representan a los vectores de soporte son marcados con una cruz en color negro

En lo consecuente, se utilizan los vectores aumentados con un 1 como sesgo de entrada, y para mayor claridad, se diferencian a éstos con una tilde por encima. Es decir, si $s_1 = (1,0)$ entonces $\tilde{s_1} = (1,0,1)$. En la Figura 2.21 se muestra la arquitectura SVM. Ahora el problema se basa en encontrar los valores para α_i tal que:

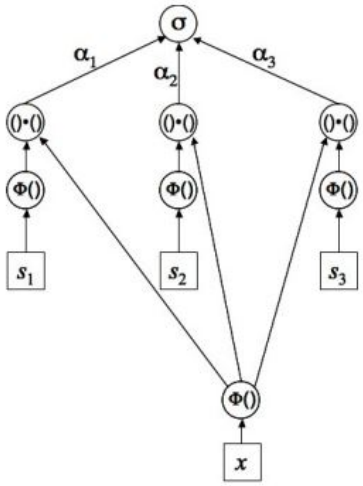


Figura 2.21: Arquitectura de la SVM

$$\alpha_1 \Phi(s_1) \cdot \Phi(s_1) + \alpha_2 \Phi(s_2) \cdot \Phi(s_1) + \alpha_3 \Phi(s_3) \cdot \Phi(s_1) = -1$$

$$\alpha_1 \Phi(s_1) \cdot \Phi(s_2) + \alpha_2 \Phi(s_2) \cdot \Phi(s_2) + \alpha_3 \Phi(s_3) \cdot \Phi(s_2) = +1$$

$$\alpha_1\Phi(s_1)\cdot\Phi(s_3)+\alpha_2\Phi(s_2)\cdot\Phi(s_3)+\alpha_3\Phi(s_3)\cdot\Phi(s_3)=+1$$

Dado que $\Phi()=I$, se reduce a:

$$\alpha_1(\bar{s_1})\cdot\bar{s_1}+\alpha_2(\bar{s_2})\cdot\bar{s_1}+\alpha_3(\bar{s_3})\cdot(\bar{s_1})=-1$$

$$\alpha_1(\bar{s_1})\cdot\bar{s_2}+\alpha_2(\bar{s_2})\cdot\bar{s_2}+\alpha_3(\bar{s_3})\cdot(\bar{s_2})=+1$$

$$\alpha_1(\bar{s_1})\cdot\bar{s_3}+\alpha_2(\bar{s_2})\cdot\bar{s_3}+\alpha_3(\bar{s_3})\cdot(\bar{s_3})=+1$$

Realizando los cálculos algebráicos por software, el resultado de los productos punto, se obtiene

$$2\alpha_1+4\alpha 2+4\alpha 3=-1$$

$$4\alpha_1+11\alpha 2+9\alpha 3=+1$$

$$4\alpha_1+9\alpha 2+11\alpha 3=+1$$

Resolviendo el sistema de ecuaciones, se obtienen los valores de $\alpha_1=-3.5$, $\alpha_2=0.75$ y $\alpha_3=0.75$. A continuación, es posible observar que a esos valores de α se relacionan con el hiperplano discriminante. En otras palabras, ahora con los valores de α_i podemos construir el hiperplano que clasifique a ambos conjuntos de datos:

$$\bar{\omega}=\sum_{i=1}^{n}\alpha_i\bar{s}_i \tag{2.58}$$

Si se sustituyen los valores de α_i y $\bar{s}_i$ se obtiene:

$$-3.5\begin{pmatrix}1\\0\\1\end{pmatrix}+0.75\begin{pmatrix}3\\1\\1\end{pmatrix}+0.75\begin{pmatrix}3\\-1\\1\end{pmatrix}=\begin{pmatrix}1\\0\\-2\end{pmatrix}$$

Finalmente, recordando que los vectores se encuentran aumentados con un sesgo de entrada, se puede igualar la entrada final en $\bar{\omega}$ como el sesgo b del hiperplano y escribir la ecuación del hiperplano de separación como $y=\omega x+b$ con $\omega=\binom{1}{0}$ y $b=-2$. En la Figura 2.22,

se traza la línea de la superficie de desición.

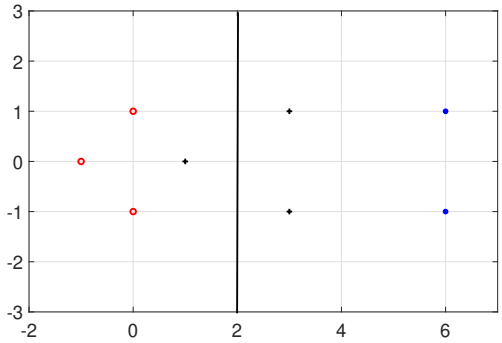


Figura 2.22: Hiperplano de separación que corresponde a los valores de $\alpha_1 = -3.5$, $\alpha_2 = 0.75$ y $\alpha_3 = 0.75$

2.6.6. Ejemplo: caso linealmente no separable

A continuación, se supone que tenemos el siguiente conjunto de datos etiquetados positivamente en el espacio $\mathbb{R}^2$

$$\left\{\begin{pmatrix}2\\2\end{pmatrix},\begin{pmatrix}2\\-2\end{pmatrix},\begin{pmatrix}-2\\-2\end{pmatrix},\begin{pmatrix}-2\\2\end{pmatrix}\right\}$$

y los siguientes datos, etiquetados negativamente en el espacio $\mathbb{R}^2$:

$$\left\{\begin{pmatrix}1\\1\end{pmatrix},\begin{pmatrix}1\\-1\end{pmatrix},\begin{pmatrix}-1\\-1\end{pmatrix},\begin{pmatrix}-1\\1\end{pmatrix}\right\}$$

Ambos conjuntos de datos se muestran gráficamente en la Figura 2.23.

Ahora se tiene como objetivo, el encontrar un hiperplano que sea capaz de discriminar entre las dos clases. Es obvio que este hiperplano no existe en el espacio de entrada, es decir, en el espacio en que se representan los datos de entrada originales. Por lo que es necesario hacer uno de la SVM no lineal, es decir, cuando el mapeo de la funcion Φ es una asignación

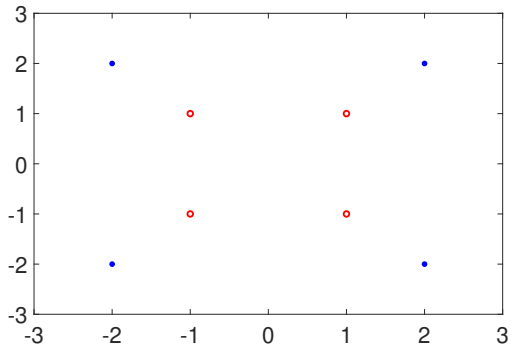


Figura 2.23: conjunto de datos no separable en el espacio $\mathbb{R}^2$. Los puntos azules corresponden al conjunto positivo, y los puntos rojos al conjunto negativo

desde el espacio de entrada no lineal a algún espacio de características. Se define:

$$\Phi\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{cases} \begin{pmatrix} 4 - x_2 + |x_1 - x_2| \\ 4 - x_1 + |x_1 - x_2| \end{pmatrix} & \text{si} \sqrt{x_1^2 + x_2^2} > 2 \\ \begin{pmatrix} x_2 \\ x_1 \end{pmatrix} & \text{en otro caso} \end{cases} \tag{2.59}$$

En la Figura 2.24 es posible apreciar como Φ transforma nuestros datos antes de realizar los productos punto. Por lo tanto podemos reescribir los datos etiquetados como positivos en el espacio de características como:

$$\left\{ \begin{pmatrix} 2 \\ 2 \end{pmatrix}, \begin{pmatrix} 6 \\ 2 \end{pmatrix}, \begin{pmatrix} 6 \\ 6 \end{pmatrix}, \begin{pmatrix} 2 \\ 6 \end{pmatrix} \right\}$$

y los datos etiquetados como negativos como:

$$\left\{ \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ 1 \end{pmatrix} \right\}$$

Ahora es mas fácil identificar cuales son los puntos que representan los vectores de soporte (Fig2.25):

$$\left\{ s_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, s_2 = \begin{pmatrix} 2 \\ 2 \end{pmatrix} \right\}$$

Nuevamente se hace uso de los vectores aumentados con un 1 como sesgo de entrada, y se diferencian como se hizo en el ejemplo anterior. Ahora que se tienen los vectores de soporte,

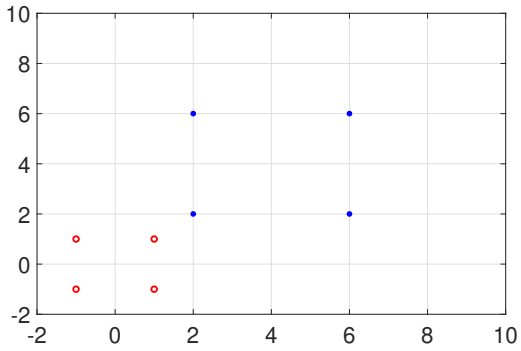


Figura 2.24: Los datos etiquetados positivos y negativos representados en el espacio de características

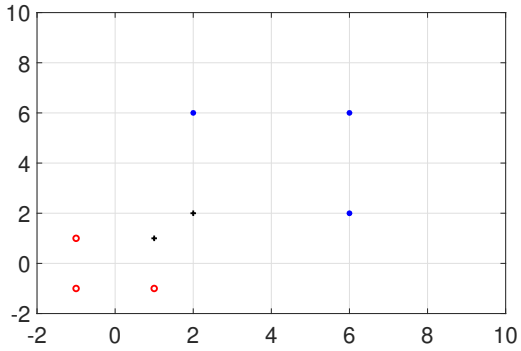


Figura 2.25: Los dos vectores de soporte en el espacio de características son marcados con una cruz en color negro

el problema se centra en encontrar los valores de α_i. Ahora se representa nuestro sistema como:

$$\alpha_1 \Phi(s_1) \cdot \Phi(s_1) + \alpha_2 \Phi(s_2) \cdot \Phi(s_1) = -1$$

$$\alpha_1 \Phi(s_1) \cdot \Phi(s_2) + \alpha_2 \Phi(s_2) \cdot \Phi(s_2) = +1$$

Esta ecuación se reduce a:

$$\alpha_1 \bar{s_1} \cdot \bar{s_1} + \alpha_2 \bar{s_2} \cdot \bar{s_1} = -1$$

$$\alpha_1 \bar{s_1} \cdot \bar{s_2} + \alpha_2 \bar{s_2} \cdot \bar{s_2} = -1$$

Se debe tener en cuenta que aunque Φ es una función no trivial, tanto s_1 como s_2 se mapean a ellos mismos, bajo Φ. Este no es el caso para todas las entradas, como se verá en el ejemplo

siguiente. A través de la realización del producto punto se obtiene:

$$\begin{pmatrix} 3\alpha_1 + 5\alpha_2 = -1 \\ 5\alpha_1 + 5\alpha_2 = +1 \end{pmatrix}$$

Al resolver el sistema de ecuaciones, se obtiene $\alpha_1 = -7$ y $\alpha_2 = 4$. Finalmente se obtiene el hiperplano de separación en el espacio de entradas correspondiente a esos valores de α:

$$\bar{\omega} = \sum_{i=1}^{n} \alpha_i \bar{s}_i \tag{2.60}$$

sustituyendo los valores de α, se obtiene:

$$-7\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + 4\begin{pmatrix} 2 \\ 2 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ -3 \end{pmatrix}$$

Dados estos valores, es posible separar la ecuación del hiperplano como $y = \omega x + b$ con $\omega = \binom{1}{1}$ y $b = -3$. En la Figura 2.26, se traza la línea de la superficie de desición.

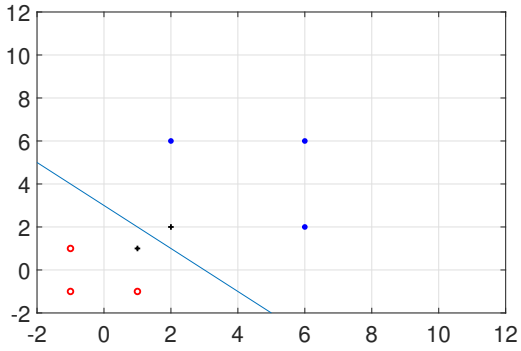


Figura 2.26: El hiperplano de separación correspondiente a los valores de α_1=7 y $\alpha_2 = 4$

2.6.7. Ejemplo: Uso del Kernel

La definición de la ecuación 2.59 Φ preserva el número de dimensiones. Es decir, nuestras entradas y el espacio de características son del mismo tamaño. Usualmente, es necesario que, para separar efectivamente los datos, debemos usar un espacio de características que es de

mayor dimensión que el espacio de salida. Ahora se considera una alternativa de la función de mapeo:

$$\Phi_2\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ \frac{(x_1^2+x_2^2)-5}{3} \end{pmatrix} \tag{2.61}$$

dicha ecuación transforma nuestros datos de un espacio de entrada de dos dimensiones a un espacio tridimensional. Usando esta transformación, con los datos utilizados en el ejemplo anterior, en términos del nuevo espacio de características se muestran como:

$$\left\{ \begin{pmatrix} 2 \\ 2 \\ 1 \end{pmatrix} \begin{pmatrix} 2 \\ -2 \\ 1 \end{pmatrix} \begin{pmatrix} -2 \\ -2 \\ 1 \end{pmatrix} \begin{pmatrix} -2 \\ 2 \\ 1 \end{pmatrix} \right\}$$

para los datos etiquetados positivos, y los datos etiquetados como negativos se muestran como:

$$\left\{ \begin{pmatrix} 1 \\ 1 \\ -1 \end{pmatrix} \begin{pmatrix} 1 \\ -1 \\ -1 \end{pmatrix} \begin{pmatrix} -1 \\ -1 \\ -1 \end{pmatrix} \begin{pmatrix} -1 \\ 1 \\ -1 \end{pmatrix} \right\}$$

Sin la necesidad de un análisis exhaustivo se observa que, para este caso, los 8 vectores de soporte corresponden a $\alpha_i = \frac{1}{46}$ para los vectores de soporte positivos y $\alpha_i = \frac{-7}{46}$ para los vectores de soporte negativos. Se observa que a consecuencia de esta transformación no es necesario aumentar los vectores dado que el hiperplano en el espacio de características pasa a través del origen, donde $y = \omega_x + b$, $\omega = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ y $b = 0$. Por lo tanto, la característica discriminante es x_3, y la ecuación se reduce a $f(x) = \sigma(x_3)$.

2.7. Sistema operativo para robots (ROS)

ROS es un conjunto de paquetes y herramientas de software para el desarrollo de aplicaciones de robótica. La unidad principal para la organización de software en ROS, se denomina paquete, el cual puede contener en su interior una biblioteca dependiente de ROS, conjuntos de datos, archivos de configuración, conjuntos de archivos o procesos, llamados nodos.
El ROS MASTER proporciona el registro de nombre para todos los nodos, y consulta el resto del gráfico de procesos, como se muestra en la Fig. 2.27. Sin el ROS MASTER, los nodos no serían capaces de encontrar al resto de ellos, intercambiar mensajes o invocar servicios. Para ejecutar este proceso es necesario del comando **roscore**, que es el primer paso para invocar a los nodos o visualizar los tópicos activos. La única excepción es si se ejecuta un archivo con la extensión **.launch**, que realiza automáticamente este proceso.

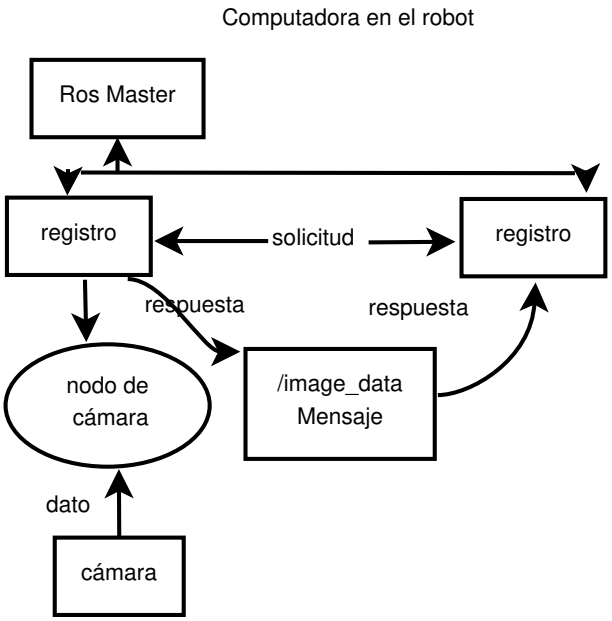


Figura 2.27: Ejemplo de funcionamiento de ROS por medio del intercambio de datos en un tópico y dos nodos

Un nodo es un archivo ejecutable dentro de un paquete de ROS. Utiliza una biblioteca de cliente de ROS para comunicarse con otros nodos. Puede publicar o suscribirse a un tópico, lo cual es configurado directamente desde la codificación. Un nodo de ROS funciona con el uso de bibliotecas de cliente de ROS, roscpp (C++) y rospy (Phyton). Para poder ver la lista de nodos activos se utiliza el comando **rosnode list**, donde siempre estará activo el **rosout**, el

cual siempre se ejecuta a medida que recoge y registra la salida de depuración de los nodos. Ros tiene herramientas para manejar los nodos y dar información acerca de ellos, a través de **rosnode**:

- rosnode info *"nodo"*: muestra la información sobre el nodo.
- rosnode kill*"nodo"*: detiene o elimina las funciones del nodo, a través del envío de una señal.
- rosnode list: muestra una lista con los nodos activos

Los nodos se comunican a través de mensajes. Un mensaje es una estructura de datos compuestos y comprende una combinación de tipos primitivos y mensajes (válgase la redundancia). Los tipos primitivos de datos son entero, punto flotante y boleano. Los mensajes pueden incluir estructuras y arreglos (como en lenguaje C). cada tópico contiene un tipo específico de dato.
Los tópicos se conocen como buses sobre los cuales los nodos intercambian mensajes. Los tópicos tienen una semántica de publicación/suscripción como se muestra en la Figura 2.28 . En general, los nodos no tienen conocimiento sobre con quién se están comunicando. Pero se pueden suscribir al tópico de interés. Posteriormente puede haber varios editores y suscriptores a un tópico. Esto significa que los tópicos pueden ser transmitidos sin comunicación directa entre nodos, lo cual conlleva a una producción y consumo de datos desacoplada. ROS tiene una herramienta para trabajar con los tópicos llamada **rostopic** en línea de comandos que proporciona información sobre el tópico o publica datos directamente sobre la red:

- rostopic bw /"tópico": muestra el ancho de banda utilizado por un tópico
- rostopic echo /"tópico": muestra el mensaje con la salida estándar
- rostopic find "tipo de mensaje": busca tópicos que usen el tipo de mensaje especificado
- rostopic hz /"tópico": muestra la tasa de publicación del tópico
- rostopic info /"tópico": muestra la información sobre el tópico, el tópico publicado, los suscriptores y los servicios

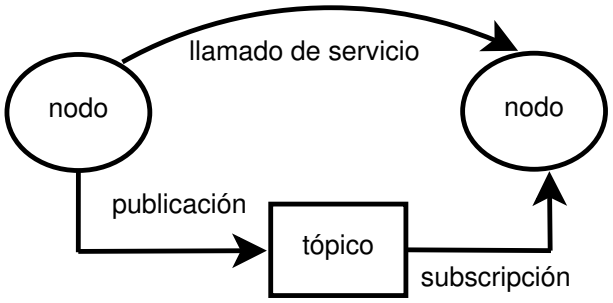


Figura 2.28: Concepto de publicador y suscriptor en ROS

En resumen los tópicos son la vía por donde se van a transmitir los datos de interés. Sin embargo cuando se necesita comunicarse con nodos y recibir una respuesta, no se puede realizar con los tópicos, sino que se utilizan los servicios, los cuales son desarrollados por el usuario. Con **rosservice** se pueden enlistar y enviar peticiones:

- rosservice call /"servicio": llama al servicio con los argumentos apropiados
- rosservice find "tipo de mensaje": Busca los servicios por el tipo de mensaje
- rosservice info /"servicio": Muestra información sobre el servicio
- rosservice list: muestra la lista de los servicios activos

Para almacenar los datos mediante una localización central, existe el servidor de parámetros, que se encuentra inmerso dentro del ROS MASTER. Es prácticamente un diccionario compartido multivariable accesible a través de la red. Los nodos usan el servidor de parámetros para almacenar y recibir parámetros en tiempo de ejecución. La herramienta proporcionada para trabajar con dicho servidor es denominada **rosparam**:

- rosparam list: muestra la lista de todos los parámetros del servidor
- rosparam get "parámetro": devuelve el valor de un parámetro
- rosparam "parámetro": establece el valor de un parámetro
- rosparam delete "parámetro": elimina un parámetro
- rosparam load “archivo”: Carga un fichero con parámetros en el servidor de parámetros.

Capítulo 3

Materiales y métodos

Este capítulo presenta la metodología para realizar el algoritmo de detección y localización de fallas, bajo dos métodos experimentales. El primer método es a través de las señales de rotación angular provenientes de la plataforma experimental para simulación de vuelo, y el segundo, corresponde a los datos obtenidos por la IMU dentro de la tarjeta controladora del dron.

3.1. Metodología para la detección y localización de fallas

El arreglo para la FDI utiliza las señales de las vibraciones correspondientes a las posiciones angulares obtenidas por los sensores (encoders) de una plataforma mecánica de tipo giroscopio de la empresa Eureka Dynamics, referida como FFT-GYRO (Valencia-Palomo y cols., 2018). El VANT se monta sobre el FFT-GYRO y el arreglo experimental se muestra en la Fig. 3.1. Esta plataforma se encuentra instrumentada con un sistema de cardanes que permiten el desplazamiento rotacional del vehículo en los tres ejes. En cada eje se encuentra un encoder (E_1, E_2 y E_3) para el monitoreo de los ángulos, a través de las vibraciones producidas por el desplazamiento del vehículo. Estos sensores se encuentran conectados a una tarjeta de adquisición de datos, que se enlaza al ordenador para el registro de las señales. Dichas señales se obtienen y se procesan en el ordenador mediante **Matlab** y **Simulink**. El diagrama a bloques utilizado en simulink se muestra en la Fig. 3.2

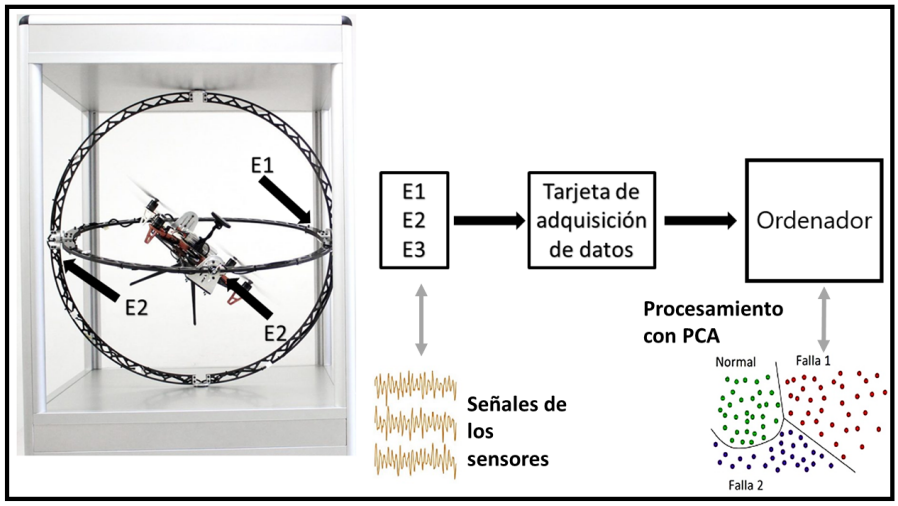


Figura 3.1: Arreglo experimental propuesto.

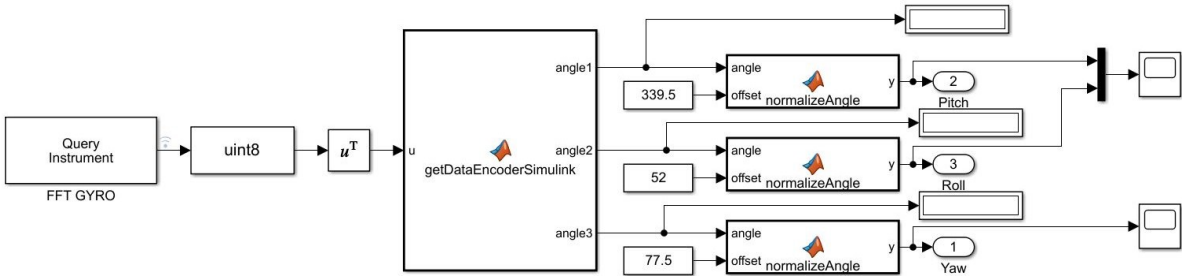


Figura 3.2: Programación a bloques en Simulink para la obtención de señales angulares

Dado que la falla más común en este tipo de vehículos es la pérdida de total o parcial de la hélice, se realizó la caracterización de fuerza de empuje del motor Emax2212, con el cual se estuvo trabajando. Para ello se realizaron 3 experimentos: cuando la hélice estaba completa, cuando presentaba una pérdida de 1cm, y cuando presentaba una pérdida de 2 cm. El banco experimental propuesto se observa en la Figura 3.3, donde, a través de una NIDAQ USB-6002, se contabilizaron los pulsos por segundo del motor, para cuantificar la velocidad en rps. En el otro extremo se colocó una báscula digital para cuantificar la fuerza de empuje. Los datos obtenidos con la NIDAQ se visualizaron mediante labVIEW gráficamente, como se muestra en la figura , donde se presenta la velocidad angular con respecto al tiempo. Para cuantificar la fuerza de empuje de los motores, se obtuvieron las fuerzas que marcaba la báscula bajo diferentes velocidades angulares y con ello se hizo una comparación de estos datos con la hélice completa, y con el desprendimiento de 1cm y el desprendimiento de 2 cm. Estos datos se compran en la Figura 3.5, donde acertadamente se muestra que al disminuir la dimensión

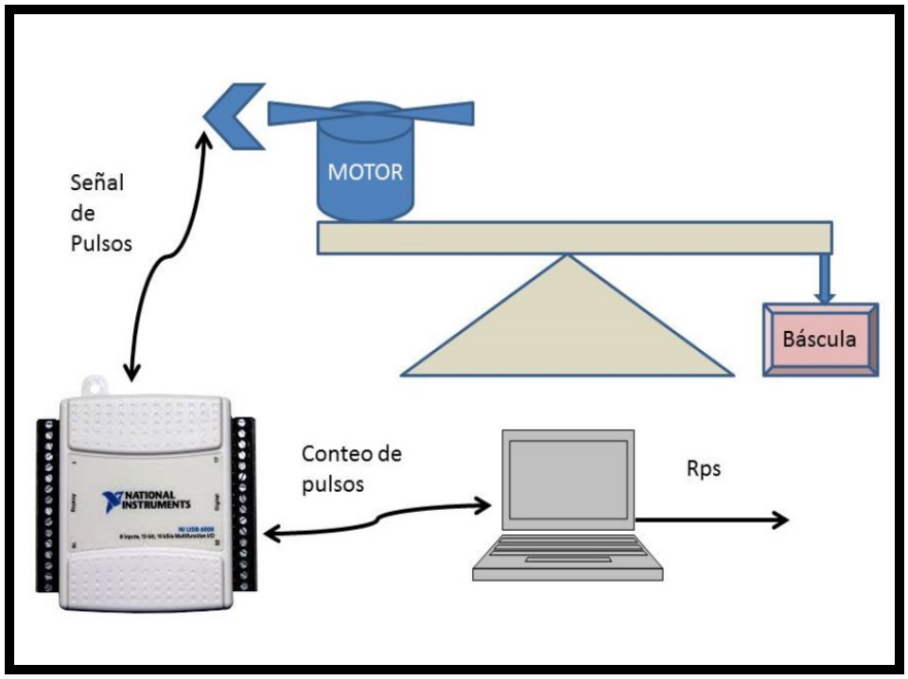


Figura 3.3: Banco experimental propuesto para caracterización de fuerza de los motores.

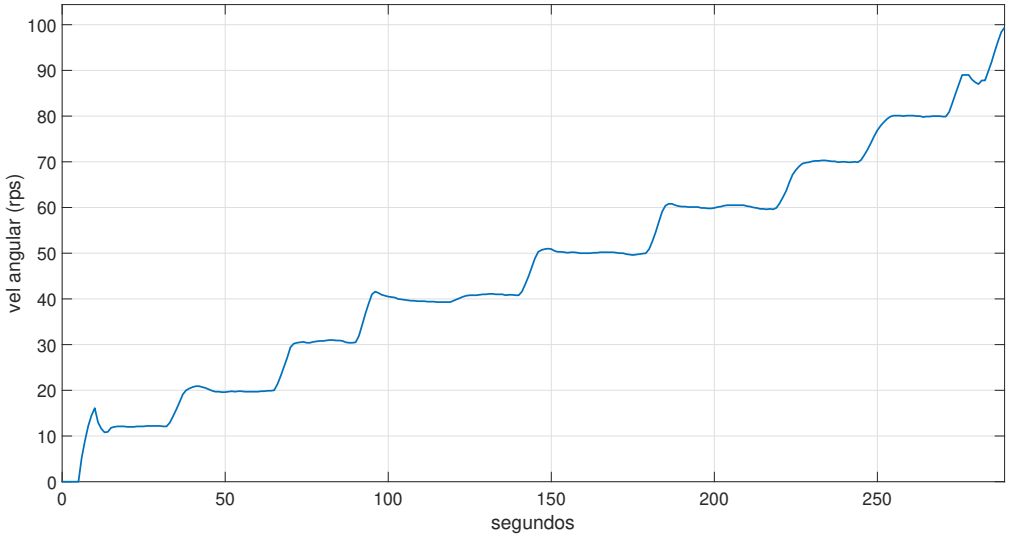


Figura 3.4: Grafica de los datos obtenidos con la NIDAQ de la velocidad angular (rps) con respecto al tiempo (s)

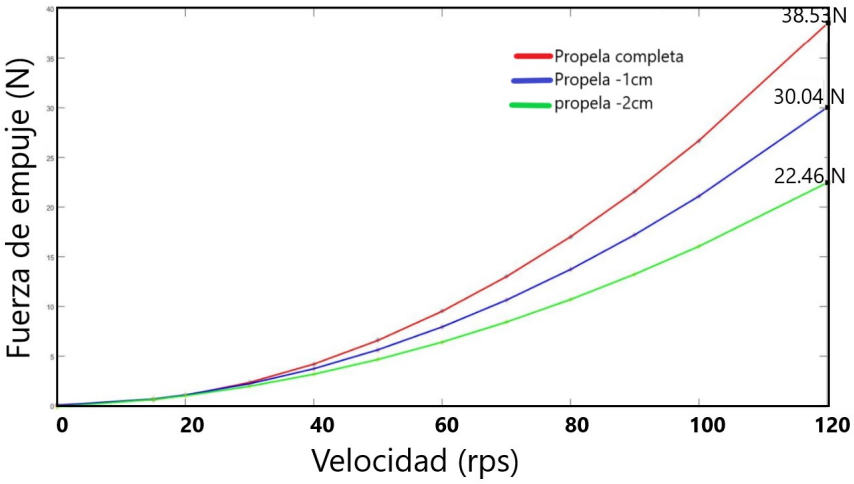


Figura 3.5: Pérdida de fuerza de empuje en el motor al disminuir la dimensión de la propela

de la hélice, disminuye la fuerza de empuje, y con ello se infiere la existencia de falla para el control del vehículo.

3.2. Detección de fallas mediante señales de rotación angular

El procedimiento para realizar la detección de fallas mediante las señales de rotación angular fue el siguiente. Primero se obtuvieron m cantidad de datos de los 3 sensores correspondientes a la posición angular (alabeo, cabeceo y guiñada) del VANT operando en condiciones *nominales*, es decir *libre de fallas*, en estado de vuelo estacionario (cuando los cuatro rotores presentan la misma fuerza de empuje), donde los ejes de cabeceo y alabeo se ubicaron en 0° como punto de referencia al inicio del experimento y el eje de guiñada en un punto arbitrario. Se inició la puesta en marcha del vehículo, y las vibraciones fueron captadas por los sensores tipo encoder's en cada eje cardián. Con estos datos se construyó la matriz de mediciones $\mathbf{X}=\begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2 & \mathbf{x}_3\end{bmatrix} \in \mathbb{R}^{m\times 3}$, donde cada vector representa la información de uno de los 3 sensores en condiciones nominales. La matriz de datos $\mathbf{X}$ nominal, se procesa mediante PCA para extraer las características principales del sistema y la reducción de dimensionalidad de los datos. Una vez obtenidos las componentes principales, y caracterizada la región de operación nominal, se toman a estos como la "nueva base". Posteriormente, se provoca una falla en un el

motor R1 a través de la propela, mediante un desgaste de 2 cm. Sin quebrar la propela se coloca nuevamente en su posición y se simula el vuelo del VANT en modo de vuelo estacionario hasta el desprendimiento de la parte desgastada de la propela como se muestra en la Fig.3.6.

Figura 3.6: Hélice sin fractura y hélice con dos centímetros de desprendimiento.

El desprendimiento de una parte de la propela provoca que se reduzca el empuje del motor lo cual se traduce a una falla en el ala rotativa y además provoca vibraciones y desplazamientos en los ejes de referencia del vehículo.

Es importante mencionar que las lecturas de estas vibraciones contienen codificadas la información de la fallas, las cuales se decodifican mediante la descomposición en sus componentes principales, y se grafica en términos de la nueva base, teniendo como referencia la media y la desviación estándar de los datos sin fallas, para su comparación y discernir entre las regiones de operación del vehículo. Este procedimiento se repite para los rotores restantes 2, 3 y 4 sucesivamente para la obtención de datos con fallas, como se muestra en los casos a,b,c y d de la Fig. 3.7. Un esquema del procedimiento completo para el diagnóstico de fallas en el VANT se muestra en la Figura 3.8. Los bloques del lado izquierdo corresponden a la operación que es realizada para que el sistema caracterice las condiciones de operación nominal a través de la media y a desviación estándar y obtener las componentes principales. Los bloques del lado derecho corresponden a la operación cuando el sistema obtiene los nuevos datos de entrada y con ello se normalizan y se dejan en términos de la nueva base. Se obtiene el estadístico T^2 para las nuevas muestras, y estos valores permiten discernir entre la existencia o la ausencia de la falla.

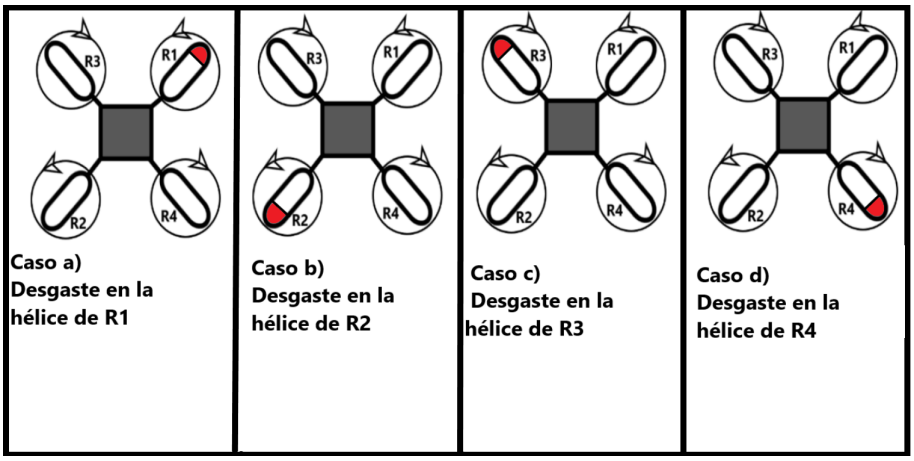


Figura 3.7: Casos de estudio de fallas en los rotores del vehículo a través del desgaste en las propelas.

3.3. Detección de fallas mediante señales de la unidad de medición inercial

El procedimiento para realizar la detección de fallas mediante las aceleraciones es el siguiente. Primero se obtienen m cantidad de datos correspondientes a la orientación (x,y,z,w), velocidad angular(x,y,z) y la aceleración lineal (x,y,z) del VANT operando en condiciones *nominales* en estado de vuelo estacionario. Para realizar esta tarea fue necesario obtener los datos proporcionados por la IMU, inmersa en tarjeta de control del VANT. Este proceso se observa en la Fig.3.9 , donde por medio de la red wifi que crea el VANT, permite la conexión inalámbrica con el ordenador. A través del ordenador mediante programación es posible obtener conectarse al **ROSMASTER** de ErleBrain3, y a través de la publicación de sus tópicos, obtener los datos de la imu a través de ROSTOPIC /MAVROS/IMU/DATA. La publicación de este tópico permite obtener los datos de orientación, velocidad angular y aceleración lineal, siendo esta última la más usual para este tipo análisis en vehículos. Con estos datos se construye la matriz de mediciones $\mathbf{X}=\begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_{10}\end{bmatrix} \in \mathbb{R}^{m\times 10}$, donde cada vector representa la información de los datos anteriormente especificados en sus ejes x , y, z. En la etapa de entrenamiento la matriz de datos $\mathbf{X}$, se procesa mediante PCA para extraer las características principales del sistema y la reducción de dimensionalidad de los datos. Una vez obtenidos las componentes principales, y caracterizada la región de operación nominal, se toman a estas como la “nueva base” de los datos extraídos. De igual manera se sigue con el procedimiento mostrado en la sección

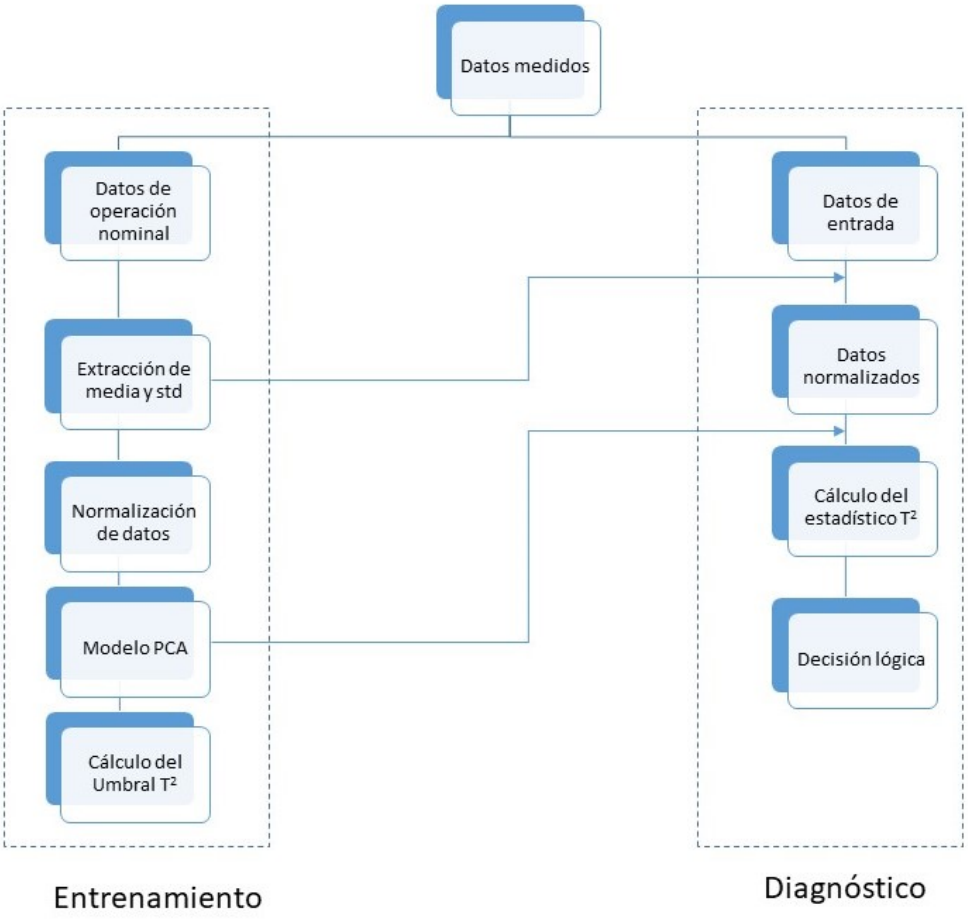


Figura 3.8: Diagrama esquemático del procedimiento de entrenamiento y diagnóstico mediante las señales de rotación angular.

anterior donde se induce falla en cada uno de los motores a través de la propela, mediante un desgaste de 2 cm. Sin quebrar la propela se coloca nuevamente en su posición y se simula el vuelo del VANT en modo estacionario montado sobre la plataforma, hasta el desprendimiento de la parte desgastada de la propela como se muestra en la Fig.3.6. Con ello se crea una base de datos con las lecturas obtenidas de las fallas en cada una de las propelas. Se repiten los casos de estudio de la Figura 3.7. A cada caso se le aplica el PCA para la extracción de características en términos de la nueva base, considerando la media y la desviación estándar de los datos nominales. Con ello se crea una matriz de datos que contiene los datos nominales, y los datos con fallas, que fueron procesados con PCA.

Los datos de la matriz que contiene la extracción de características serán ahora las nuevas

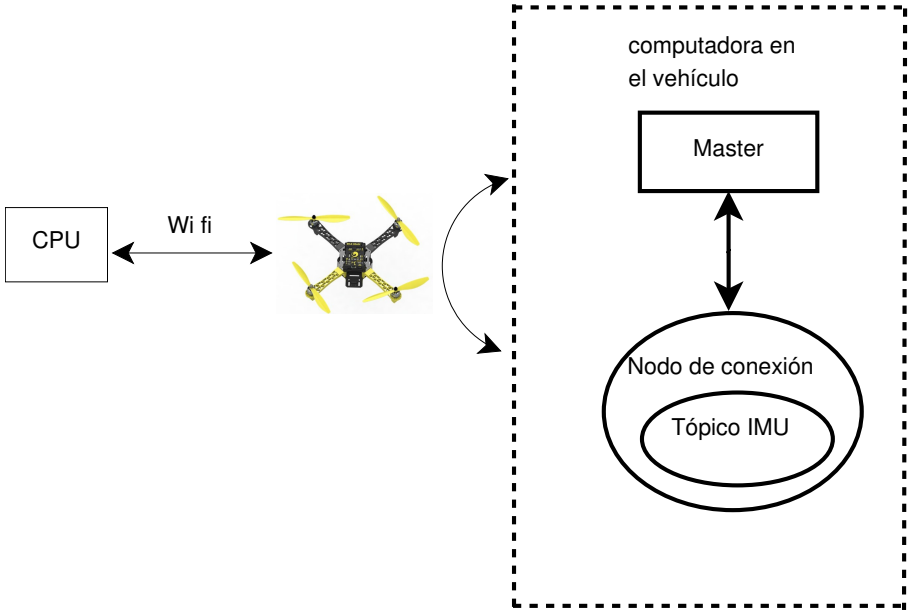


Figura 3.9: Esquema de comunicación con ROS vía wifi para obtención de datos de la IMU

entradas para el clasificador en la etapa de entrenamiento, que se aplicará fuera de línea, dado que ambos pertenecen a algoritmos de aprendizaje de máquina supervisados, deben ser previamente etiquetados. Para la etapa de entrenamiento se trabajó con el 80 % de los datos obtenidos y 20 % para la etapa de diagnóstico.

Con el modelo FDI definido, se podrá poner a prueba la efectividad del sistema de detección y aislamiento de fallas, con los nuevos datos de entrada, correspondientes al 20 % de los datos obtenidos, y se validará la robustez del sistema cuando se complementa con el clasificador SVM y *k-NN*, para verificar cuál clasificador tiene mejor desempeño en la etapa de diagnóstico, como se muestra en la Figura 3.10, la cual representa un bosquejo del procedimiento completo para el diagnóstico de fallas. Los bloques en el lado izquierdo corresponden a las operaciones que se llevan a cabo en la etapa de entrenamiento, para que el sistema pueda estadísticamente caracterizar la condición de operación normal por medio de sus componentes principales. Los bloques del lado derecho corresponden a las operaciones que ejecuta el sistema en la etapa de diagnóstico, donde las entradas a los clasificadores son los datos basados en el modelo definido previamente y a la salida del clasificador se obtiene la decisión de la clase a la que pertenece el dato. El que alguno de los algoritmos de clasificación tenga un

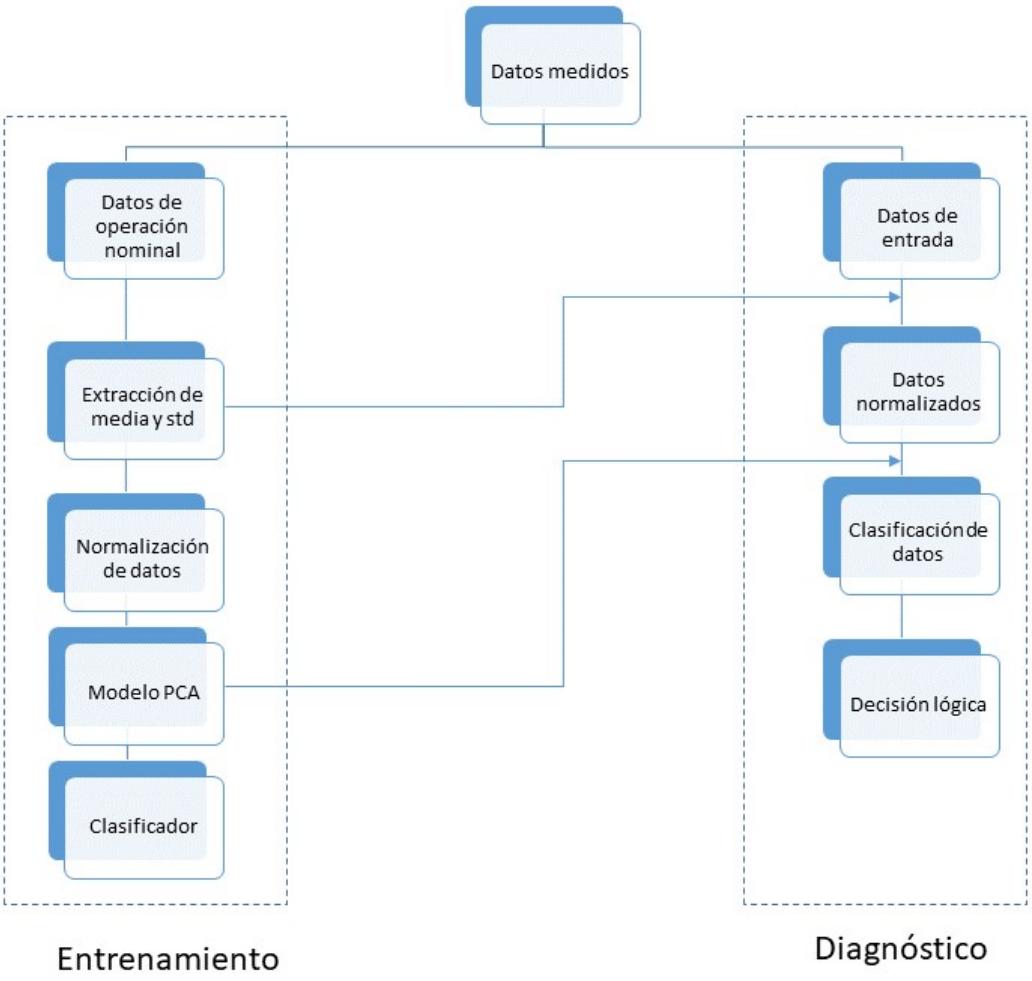


Figura 3.10: Diagrama esquemático del procedimiento de entrenamiento y diagnóstico mediante las señales obtenidas de la IMU.

mejor desempeño en la etapa de entrenamiento, no implicaría que también tenga un buen desempeño en la etapa de diagnóstico, por lo que se debe comprobar ambos clasificadores tanto en la etapa de entrenamiento como en la etapa de diagnóstico.

Capítulo 4

Resultados y discusión

4.1. Resultados obtenidos mediante datos de rotación angular

Una vez obtenido los datos del VANT cuatrimotor en condiciones normales (Fig.4.1) y en condiciones de fallas en los rotores r_1, r_2, r_3 y r_4 (figuras 4.3a,4.3b,4.3c, 4.3d) se extraen los datos y se normalizan, es decir, se dejan en una distribución normal con media cero y desviacón estandar en 1, para posteriormente obtener la matriz de puntuaciones estandarizadas y se grafican en términos de la nueva base correspondiente a las componentes principales. Para este caso en particular, resultó que el sistema se define en términos de 3 componentes principales (P_1, P_2 y P_3), debido a que la varianza total del conjunto de datos es $\lambda_1+\lambda_2+\lambda_3$=1.4509+0.9677+0.5814=3. Estos datos corresponden a la varianza explicada por cada componente y se muestran de manera gráfica en el diagrama de Pareto de la Figura 4.2. Es decir con las dos primeras componentes se obtiene un 80.6194 % de la varianza del sistema, por lo que si se desea obtener un porcentaje mayor de explicación de la dinámica en el sistema, es necesario que éste quede explicado en términos de las 3 componentes principales. El comportamiento del sistema en condiciones normales se observa en la Fig. 4.4 a partir de los datos de medición en vuelo estacionario sin fallas, en términos de los 3 componentes principales, y en la Fig. 4.5 una proyección ortonormal de la misma gráfica en términos de $P1$ y $P2$. El propósito de elaborar

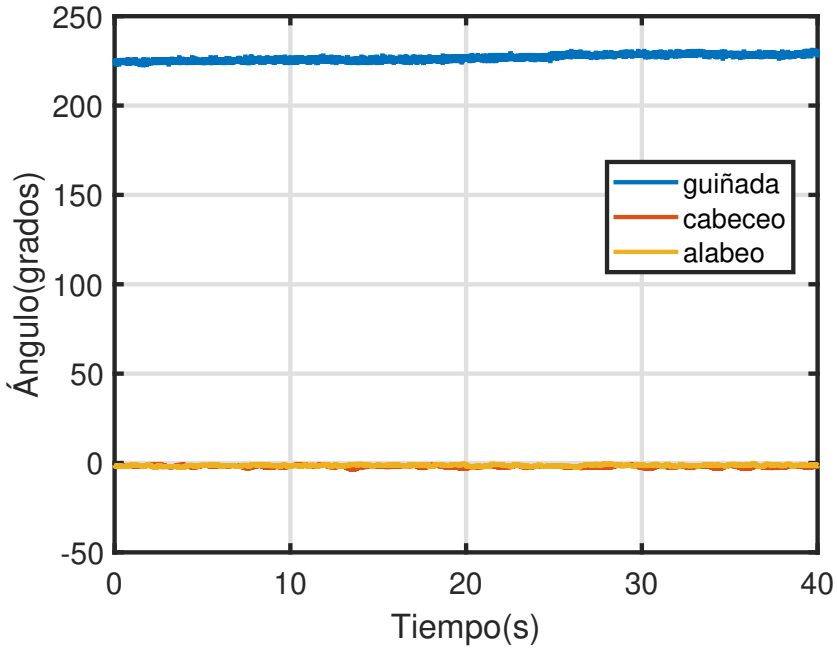


Figura 4.1: Datos del VANT en vuelo estacionario sin falla.

este gráfico es delimitar con una elipse que se extiende en ambas direcciones la región de operación nominal sin fallas con una aceptación de $\pm 3\sigma$.

Con el modelo PCA definido por ($\mathbf{P}$, Λ, $\boldsymbol{\mu}, \boldsymbol{\sigma}$) es posible inferir si cualquier medición $\mathbf{x_{new}} = [\mathbf{x_1}, \mathbf{x_2}, ...\mathbf{x_n}]^\top$ es compatible con el “comportamiento normal"del proceso. Para ello se estandariza $\mathbf{x}_{new}$ con las medias y desviaciones estándar del modelo. Luego la muestra estandarizada $\mathbf{z}_{new}$ puede ser expresada en términos de la base ortonormal $\mathbf{P}$ con las coordenadas.

$$\mathbf{t_{new}} = \mathbf{z}_{new}^\top \mathbf{P}. \tag{4.1}$$

Siguiendo el procedimiento de la ecuación(4.1) para las mediciones obtenidas ($\mathbf{x}_{new}$) con las fallas en los actuadores de los rotores 1, 2, 3 y 4, se normalizan ($\mathbf{z}_{new}$) y se proyectan en el subespacio vectorial de las Componentes Principales. Esta proyección se observa en la Fig. 4.6 en términos de las 3 componentes principales y la Fig. 4.7 una proyección ortonormal de la misma en términos de P1 y P2, cuyo objetivo es verificar la separabilidad de los datos con fallas y sin fallas (dentro de la zona de operación nominal). Una manera de determinar la existencia de fallas es comparar la nueva muestra transformada ($\mathbf{t}_{new}$) con los datos proyec-

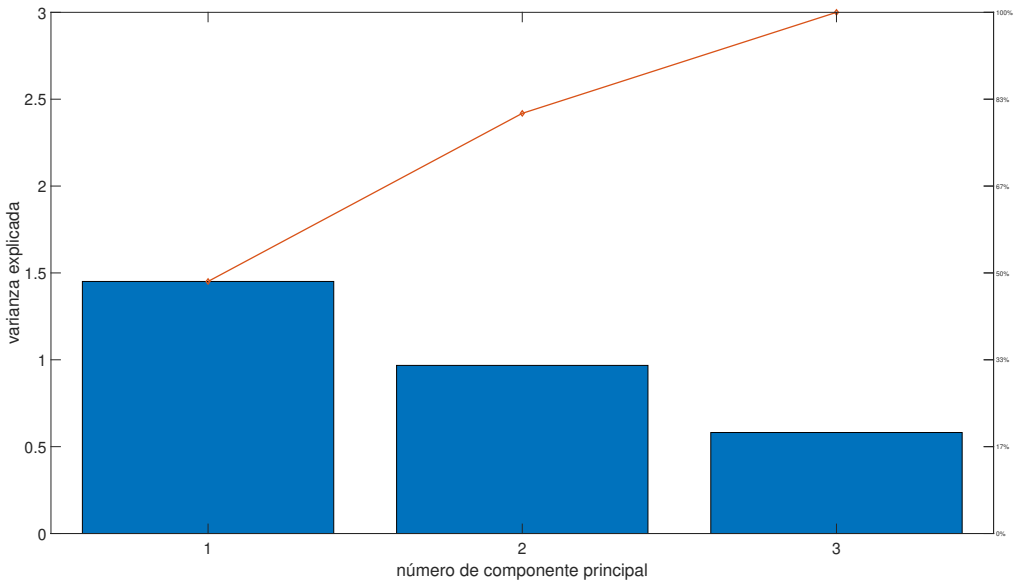


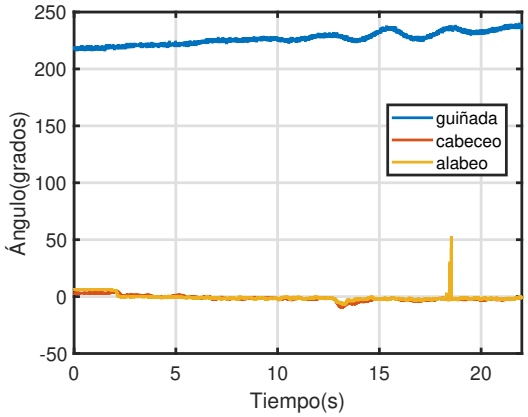
Figura 4.2: Diagrama de Pareto que muestra la contribución de cada componente principal en la explicación de la variabilidad de los datos.

tados correspondientes a la región nominal(Fig. 4.4 y 4.5). Así los datos que caen fuera de la región nominal, representan parámetros anormales (Fig.4.6 y 4.7). Sin embargo, existen otros métodos para la detección de fallas, como el uso de estadísticos de prueba, por ejemplo, el T^2 de Hotelling. Este estadístico se puede calcular a partir de la ecuación (4.2).

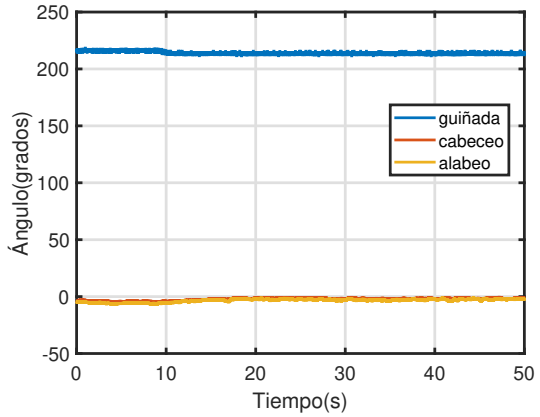
$$T^2(\mathbf{z}_{new}) = \sum_{K=1}^{q} \frac{{t_k}^2}{\lambda_k} = \mathbf{t}_q^\top \Lambda_q \mathbf{t}_q = \mathbf{z}_{new}^\top \overline{\mathbf{P}} \Lambda_q^{-1} \overline{\mathbf{P}}^\top \mathbf{z}_{new}, \tag{4.2}$$

Donde Λ_q= diag$[\lambda_1, \lambda_2, ...\lambda_n]$ contiene los eigenvalores asociados al subespacio principal. El argumento ($\mathbf{z}_{new}$) en la ec. 4.2, indica que T^2 se calcula para la muestra de prueba, aunque también es posible calcularlos para cada reglón de la matriz $\mathbf{Z}$, pero estos pueden ser estimados mediante distribuciones de probabilidad sin necesidad de un cálculo exhaustivo. Cuando el proceso opera en condiciones nominales, el estadístico T^2 tiene un valor pequeño y se infiere la existencia de fallas cuando sobrepasan determinado umbral U_{T^2}. En (Russell y cols., 2012), se establece la ec.4.3 para determinar el U_{T^2}.

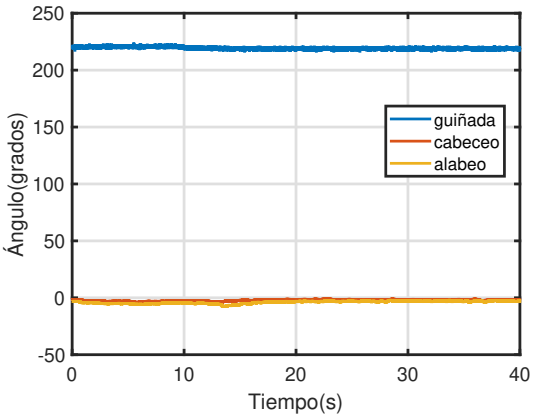
$$U_{T^2} = \frac{q(m^2-1)}{m(m-q)} F_\alpha(q, m-q). \tag{4.3}$$



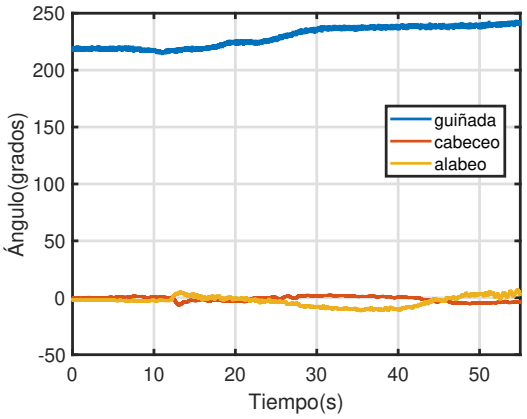
(a) Datos del VANT en vuelo estacionario cuando presenta falla en la hélice del rotor 1.



(b) Datos del VANT en estado de vuelo estacionario cuando presenta falla en la hélice del rotor 2.



(c) Datos del VANT en vuelo estacionario cuando presenta falla en la hélice del rotor 3.



(d) Datos del VANT en vuelo estacionario cuando presenta falla en la hélice del rotor 4.

Figura 4.3: Gráficas de rotación angular del vehículo en vuelo estacionario

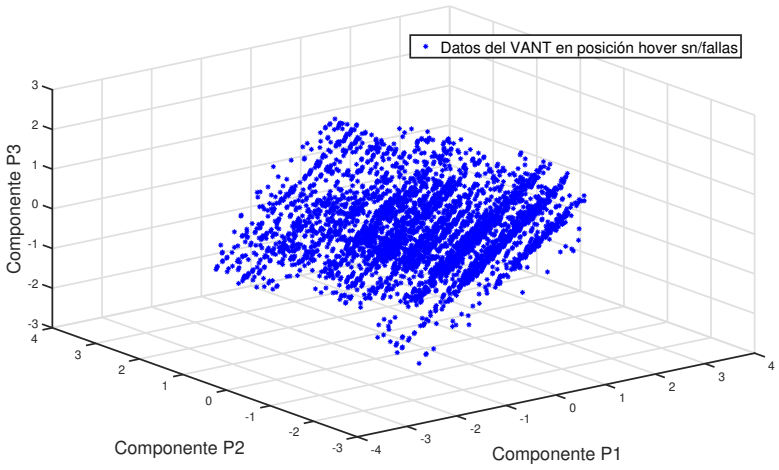


Figura 4.4: Mediciones sin fallas proyectadas dentro del subespacio PCA tridimensional

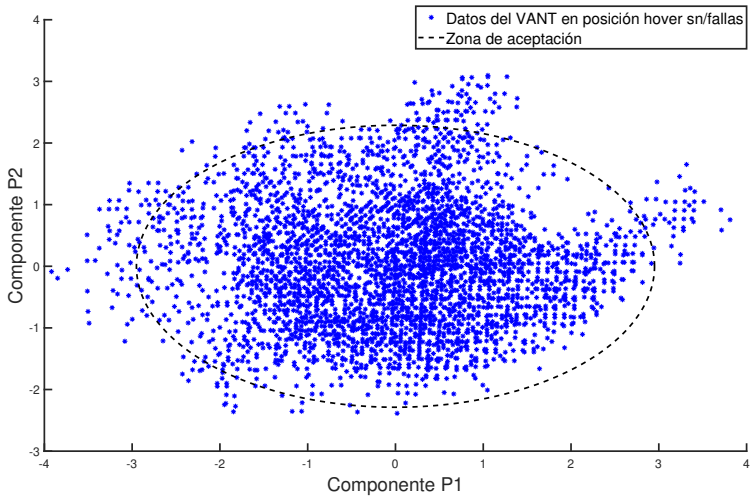


Figura 4.5: Mediciones sin fallas proyectadas dentro del subespacio PCA bidimensional

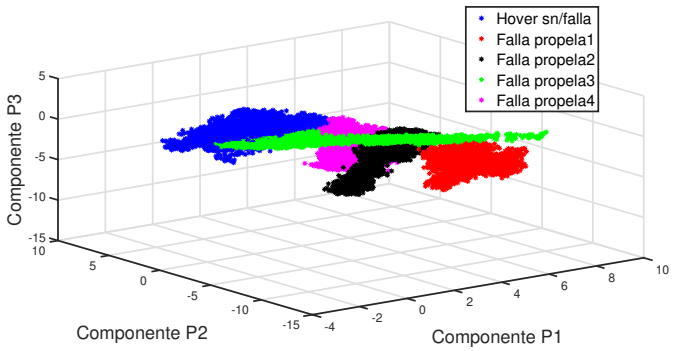


Figura 4.6: Mediciones sin fallas y con fallas proyectadas dentro del subespacio PCA tridimensional

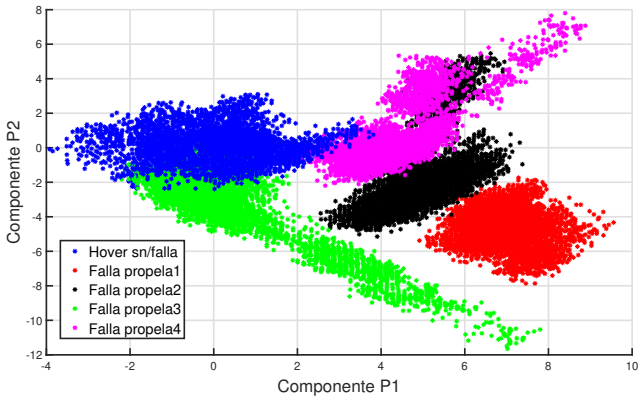


Figura 4.7: Mediciones proyectadas dentro del subespacio PCA bidimensional

Donde $F_\alpha(q, m-q)$ es el punto crítico superior de $100\alpha\,\%$ en la distribución F de Fisher con q y m-q grados de libertad. Para este trabajo se eligió una confianza de α=95 %. A partir del cálculo de U_{T^2} se obtiene que el valor de dicho umbral es de 11.35 para las condiciones nominales. Por lo que se infiere una falla cuando el valor de $T^2 > U_{T^2}$, aunque el umbral puede sobrepasarse ocasionalmente durante la operación normal, como se muestra en la Figura 4.8, dependiendo de la confiabilidad y sensibilidad deseada. Para comprobar la funcionalidad del método se seleccionó arbitrariamente el conjunto de datos de falla en la propela uno. Se observa en la Fig.4.9 que en el instante de ruptura de la propela, el dato sobrepasa el U_{T^2}. Esta figura se encuentra en una escala logarítmica en el eje 10, con la finalidad de visualizar mejor la magnitud y el instante de la falla. Después de la ruptura, se observa que el conjunto de datos regresa a condiciones nominales respecto al estadístico T^2, esto es debido a la acción del controlador interno del VANT, que compensa este tipo de fallas y perturbaciones externas.

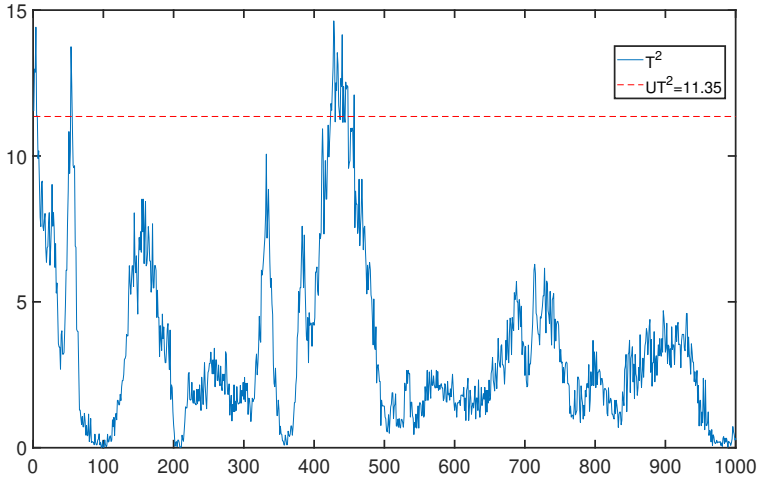


Figura 4.8: Monitoreo del proceso mediante el índice T^2 en condiciones nominales.

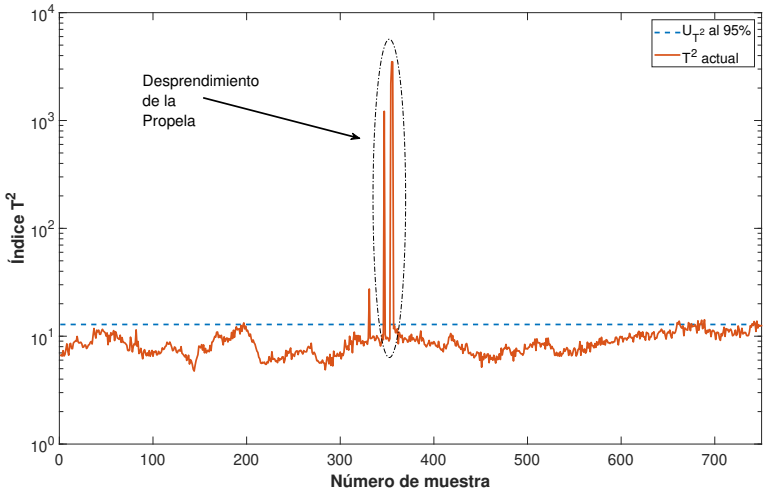


Figura 4.9: Monitoreo del proceso mediante el índice T^2 en condiciones de fallas.

4.2. Resultados obtenidos mediante datos de la unidad de medición inercial

Como primer paso se obtienen los datos de la IMU del controlador interno del VANT cuatrimotor. Para este enfoque las más representativas para el análisis resultaron ser las velocidades angulares y las aceleraciones lineales. Los datos obtenidos del vuelo en estado de suspensión sin falla se muestran en la Figura 4.10 . Los datos obtenidos cuando se presentó la falla en lo rotores r_1, r_2, r_3 y r_4 se muestran en las figuras 4.12, 4.13, 4.14, 4.15 consecutivamente. La varianza total del conjunto de datos es $\lambda_1+\lambda_2+\lambda_3+\lambda_4+\lambda_5+\lambda_5+\lambda_7+\lambda_8+\lambda_9+\lambda_{10}=2.44+1.94+1.23+1.09+1.04+0.91+0.77+0.36+0.18+0.001=10$. Estos datos corresponden a la varianza explicada por cada componente y se muestra de manera gráfica en el diagrama de Pareto de la Figura 4.11. Si se quisiera reducir la dimensionalidad de los datos, como ejemplo, podría despreciarse las ultimas 3 columnas y con ello se tendría el 94.47 % de la variabilidad de los datos.

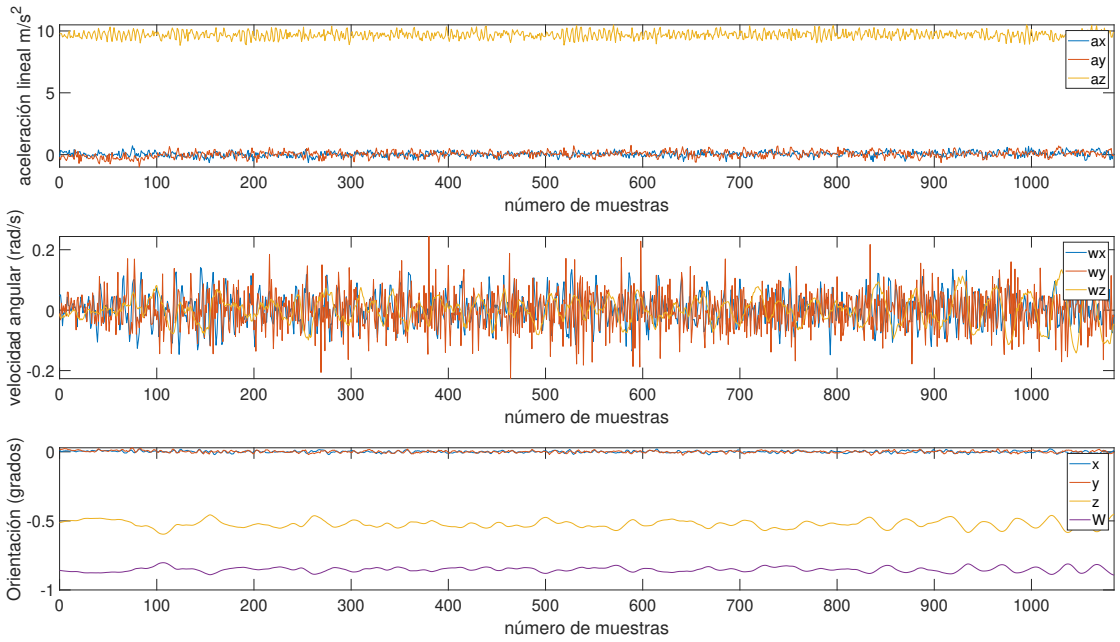


Figura 4.10: Datos del VANT en vuelo estacionario sin falla.

Con el modelo PCA definido por ($\mathbf{P}$, Λ, $\boldsymbol{\mu},\boldsymbol{\sigma}$) es posible inferir si cualquier medición

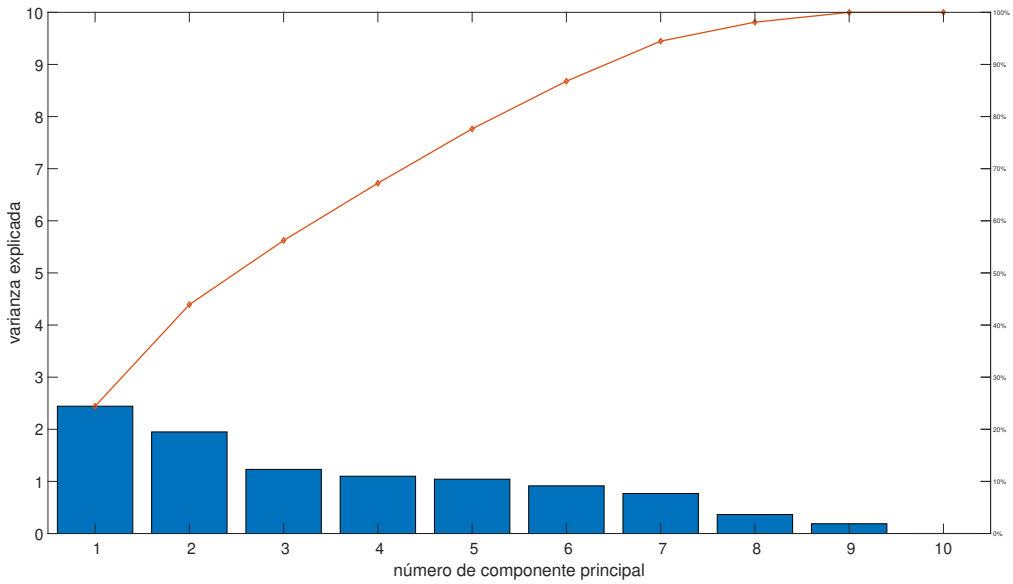


Figura 4.11: Diagrama de Pareto que muestra la contribución de cada componente principal en la explicación de la variabilidad de los datos.

$\mathbf{x_{new}} = [\mathbf{x_1}, \mathbf{x_2}, ...\mathbf{x_n}]^\top$ es compatible con el “comportamiento normal"del proceso. Para ello se estandariza $\mathbf{x}_{new}$ con las medias y desviaciones estándar del modelo nominal. Luego la muestra estandarizada $\mathbf{z}_{new}$ puede ser expresada en términos de la base ortonormal $\mathbf{P}$ con las coordenadas de la ecuación (4.1). Para las mediciones obtenidas ($\mathbf{x}_{new}$) con las fallas en los actuadores de los rotores 1, 2, 3 y 4, se normalizan ($\mathbf{z}_{new}$) y se proyectan en el subespacio vectorial de las Componentes Principales. Esta proyección se observa en la Fig.4.16 en términos de las 3 componentes principales y la Fig. 4.17 una proyección ortonormal de la misma en términos de P1 y P2, cuyo objetivo es verificar la separabilidad de los datos con fallas y sin fallas (dentro de la zona de operación nominal). Una manera de determinar la existencia de fallas es comparar la nueva muestra transformada ($\mathbf{t}_{new}$) con los datos proyectados correspondientes a la región nominal. Así los datos que caen fuera de la región nominal, representan parámetros anormales (véase Fig. 4.17).

A partir de la extracción de características (*feature extraction*, en inglés) de los datos, estos serán ahora las entradas para el entrenamiento de los clasificadores. Para este trabajo se utilizaron dos clasificadores, k-NN y SVM, para evaluar cuál ofrecía un mejor desempeño. Los datos de entrenamiento para la evaluación en ambos algoritmos se organizaron como se

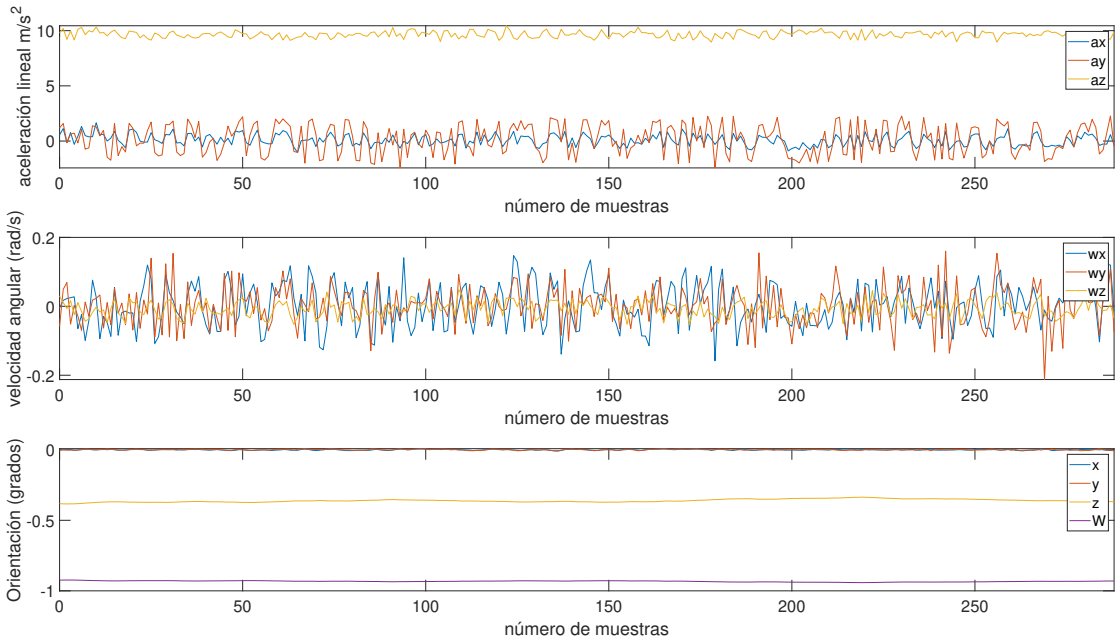


Figura 4.12: Datos del VANT en vuelo estacionario con falla en la hélice del rotor 1

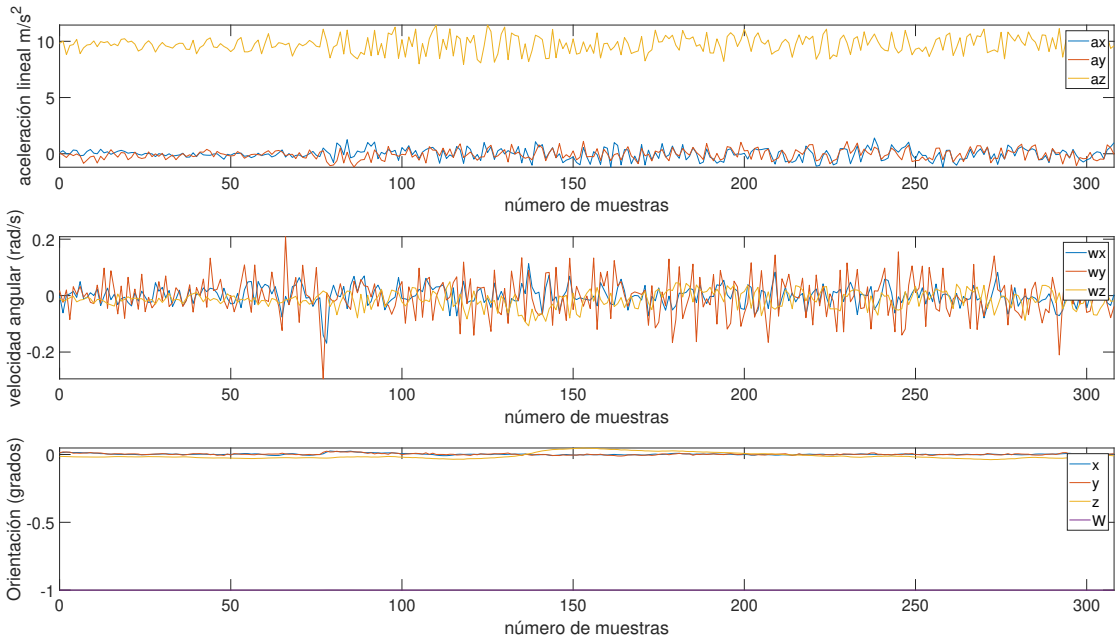


Figura 4.13: Datos del VANT en vuelo estacionario con falla en la hélice del rotor 2

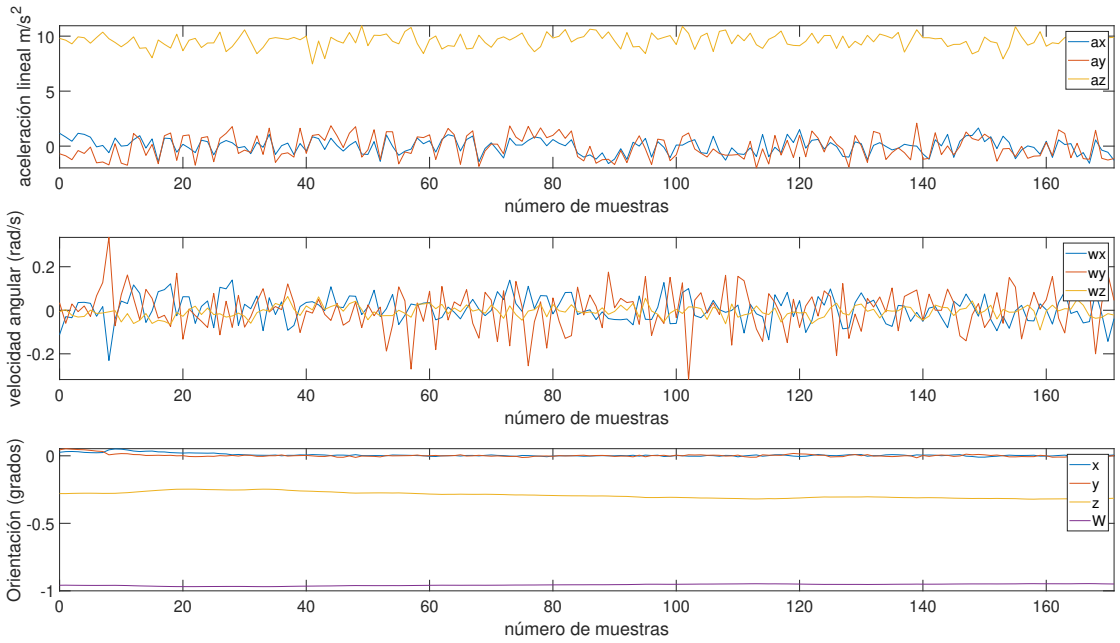


Figura 4.14: Datos del VANT en vuelo estacionario con falla en la hélice del rotor 3

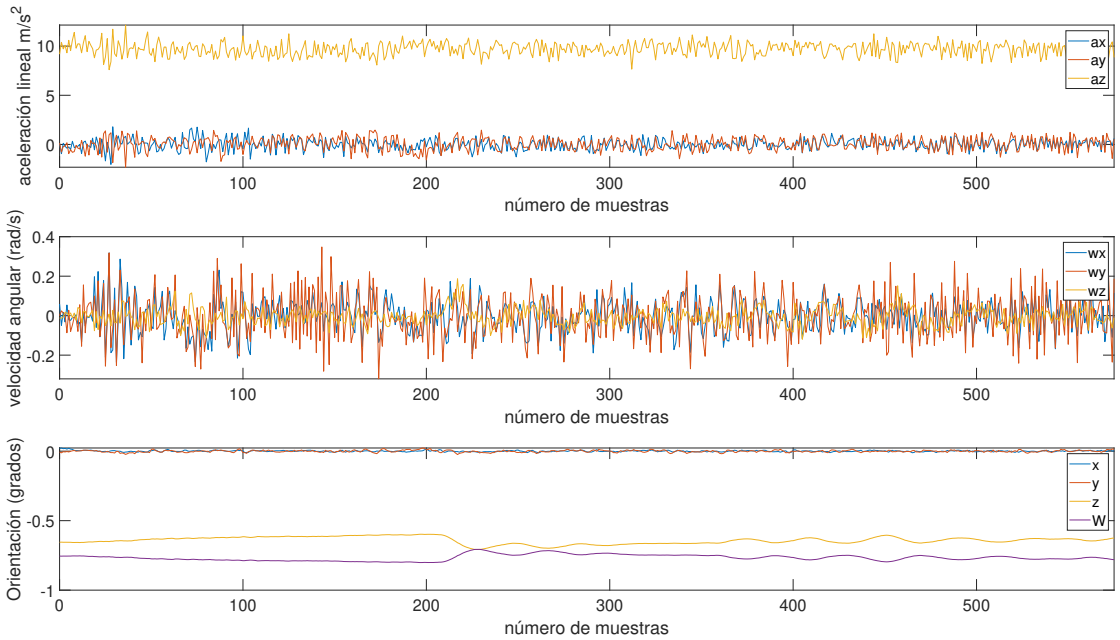


Figura 4.15: Datos del VANT en vuelo estacionario con falla en la hélice del rotor 4

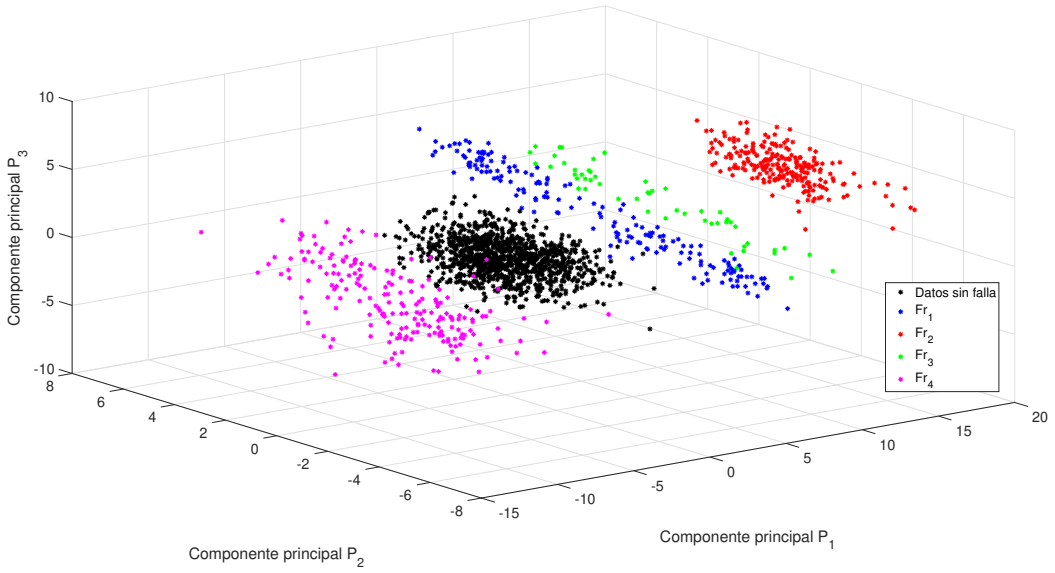


Figura 4.16: Mediciones proyectadas dentro del subespacio PCA tridimensional

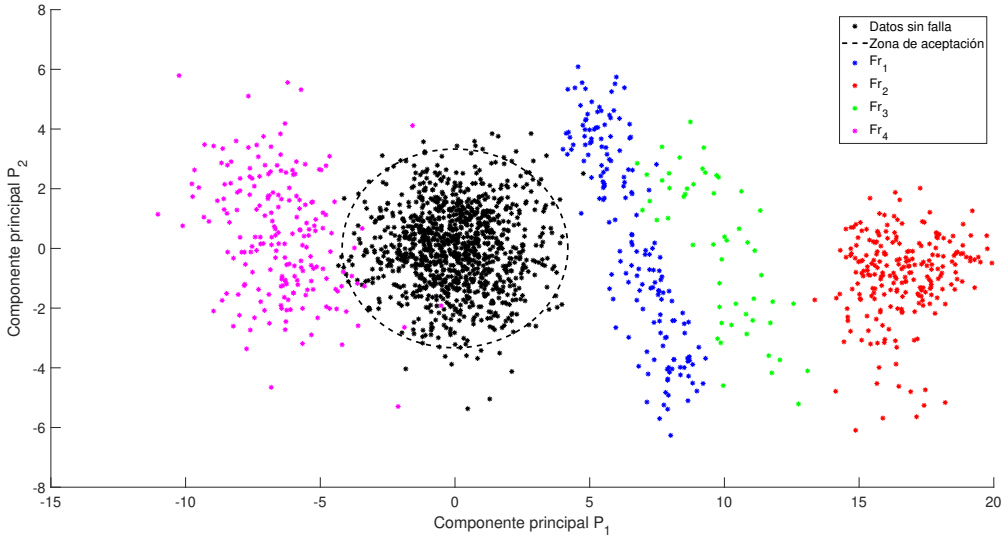


Figura 4.17: Mediciones proyectadas dentro del subespacio PCA bidimensional

Tabla 4.1: Clasificación de datos para la fase de entrenamiento de los algoritmo *k-NN* y *SVM*

Muestra	Vector de características	Clase
1	(**x1** = x1,1, x1,2...x1,n)	y1
2	(**x2** = x2,1, x2,2...x2,n)	y2
3	(**x3** = x3,1, x3,2...x3,n)	y3
4	(**x4** = x4,1, x4,2...x4,n)	y4
5	(**x5** = x5,1, x5,2...x5,n)	y5

Tabla 4.2: Número de datos para la fase de entrenamiento y diagnóstico

Clase	Descripsión	Total de datos	Datos de entrenamiento	Datos de prueba
1	Datos nominales	1085	868	217
2	Falla en r1	276	221	55
3	Falla en r2	231	185	46
4	Falla en r3	151	121	30
5	Falla en r4	376	301	75

muestra en la Tabla 4.1 donde los vectores **x1**, **x2**,..., **x5**. Contienen las características de los datos analizados mediante PCA, de la región nominal (**x1**), y de las regiones con fallas en los rotores r_1, r_2, r_3 y r_4 (**x2**,..., **x5**). Las etiquetas y_i son enteros positivos que determinan la clase a la que pertenece el dato. La clase 1 pertenece a los datos sin falla, y de la clase 2,...,clase 5, pertenecen a los datos con falla en r_1,..., r_4. En la fase de entrenamiento se eligieron un 80 % de los datos, y en la fase de prueba un 20 % como se observa en la Tabla 4.2.

Al evaluar los algoritmos utilizando el software MATLAB® se calcularon los índices de pérdidas por resustitución ($\boldsymbol{r-loss}$) el cual mide la incapacidad de los clasificadores para clasificar correctamente las fallas conocidas dentro del conjunto de entrenamiento. Para el algoritmo *k*-NN se encontró que se obtenía una mejor respuesta con un número de vecinos más cercanos, cuando $k = 3$. Con ello se obtuvo un índice $\boldsymbol{r-loss} = 0.0012$. Este resultado también se puede determinar mediante la Matriz de confusión de la Tabla 4.3a, donde cada columna de la matriz representa el número de predicciones de cada clase, mientras que cada fila representa a las instancias en la clase real. Es decir, la diagonal principal representa el número de instancias correctamente clasificadas en cada clase, y fuera de ella, las que no se clasificaron debidamente.

El algoritmo SVM obtuvo un mejor desempeño en el entrenamiento para la clasificación de los datos, logrando clasificar con éxito todas las muestras. Por lo que se obtuvo un índice

(a) Matriz de confusión del algoritmo *k-NN* en la etapa de entrenamiento

clase	**1**	**2**	**3**	**4**	**5**
1	868	0	0	0	0
2	0	221	0	0	0
3	0	0	185	0	0
4	0	2	0	119	0
5	0	0	0	0	301

(b) Matriz de confusión del algoritmo *SVM* en la etapa de entrenamiento

clase	**1**	**2**	**3**	**4**	**5**
1	868	0	0	0	0
2	0	221	0	0	0
3	0	0	185	0	0
4	0	0	0	121	0
5	0	0	0	0	301

Tabla 4.3: Tabla comparativa de los resultado obtenidos por los algoritmos de clasificación en la etapa de entrenamiento

Clase	**1**	**2**	**3**	**4**	**5**
1	183	0	0	0	44
2	0	55	0	0	0
3	0	0	46	0	0
4	0	1	0	29	0
5	0	0	0	0	75

(a) Matriz de confusión del algoritmo *k-NN* en la etapa de prueba

Clase	**1**	**2**	**3**	**4**	**5**
1	217	0	0	0	0
2	0	55	0	0	0
3	11	0	35	0	0
4	1	0	0	26	3
5	17	0	0	0	58

(b) Matriz de confusión del algoritmo *SVM* en la etapa de prueba

Tabla 4.4: Tabla comparativa de los resultados obtenidos por los algoritmos de clasificación en la etapa de prueba

$\boldsymbol{r - loss} = 0$. La matriz de confusión de este algoritmo está representado en la Tabla 4.3b. De igual manera se evaluaron los desempeños de ambos algoritmos en la etapa de prueba. Para el algoritmo *k-NN* este desempeño se observa en la Tabla 4.4a, donde se clasificaron el 92.435 % de los datos correctamente. Para el algoritmo *SVM* el desempeño se observa en la Tabla 4.4b, donde se clasificaron el 89.361 % de los datos correctamente.

Capítulo 5

Conclusiones y Perspectivas

En este trabajo se abordó el problema de la detección y clasificación de fallas en un VANT del tipo cuatrimotor, bajo un enfoque basado en el análisis de datos. Se optó por realizar este enfoque debido a que facilitan el análisis multivariable, dado que no se necesita de un conocimiento previo de la dinámica del sistema. El argumento central de este enfoque fue el de seleccionar los datos más adecuados para que, a través del análisis en componentes principales se realizaran dos tareas fundamentales: extraer las características mas importantes de los datos y reducir las dimensiones de los vectores de datos. Con esto fue posible obtener *a posteriori* una clasificación adecuada. Es por ello que se analizaron, en una primera instancia, los datos obtenidos de las señales de rotación angular a través de sensores tipo encoder colocados en los ejes de cabeceo, alabeo y guiñada del la plataforma para simulación de vuelo denominada FTT-GYRO, cuando el vehículo se encontraba en vuelo estacionario. Como se mostró previamente, al realizar el análisis en componentes principales, se observa que la separación entre las regiones de operación no es la adecuada. Esto puede deberse a que ante perturbaciones externas, los ángulos varíen y se interprete dicha desviación como falla, o viceversa, ocasionando la presencia de falsos positivos o falsos negativos, es decir, que el sistema de diagnóstico detecte una falla cuando ésta no existe o que no se diagnostique la falla cuando ésta se encuentra presente en el vehículo. Sin embargo, este enfoque es útil para la detección de fallas bajo el uso de estadísticos como el T^2. En un entorno de vuelo real sería muy difícil la aplicación de este enfoque, dado que el sistema del VANT es muy variante y propenso a

perturbaciones externas, como el viento, lo que implicaría un diagnóstico incorrecto.

En la segunda prueba experimental se analizaron los datos obtenidos de las aceleraciones lineales, aprovechando la unidad de medición inercial que se encontraba dentro de la tarjeta controladora de vuelo Erle Brain, cuando el vehículo se encontraba en condiciones de vuelo estacionario. Dicha tarjeta contiene dentro de su sistema operativo, el entorno de programación para robots, denominado ROS, lo que hizo posible la obtención de las señales de orientación, velocidad angular y velocidad lineal a través del tópico de datos de la IMU. Este enfoque es el más usado por los trabajos citados, dado que las aceleraciones se mantienen en un rango más estable sin importar la orientación del vehículo. Al realizar el análisis PCA se observa una mejor separación entre las regiones de operación nominal, y las que presentan fallas en las hélices de los rotores. Es por ello que esta metodología permitió complementarse con algoritmos de clasificación de aprendizaje de máquina, y se evaluó su desempeño. Una vez obtenidas la extracción de características con PCA, es decir, la mayor variabilidad del sistema en términos de las nuevas componentes, se tomaron a estos datos como las entradas de los algoritmos de clasificación. Para la etapa de diagnóstico se optó por trabajar con las máquinas de vectores de soportes y el algoritmo de vecinos más cercanos. El algoritmo *SVM*, para este sistema en específico, obtuvo un mejor desempeño que el *k-NN* tanto en la etapa de entrenamiento, como en la etapa de prueba. A partir de estos resultados es posible llegar a la conclusión de que el enfoque basado en datos con el algoritmo que combina la técnica de estadística PCA con el clasificador *SVM* es el más adecuado para afrontar el problema de la detección y diagnóstico de fallas en un VANT. También es posible que se pueda complementar al algoritmo con mas datos y observar su desempeño, con nuevas variables que pudieran ser importantes, por ejemplo, la corriente demandada por los rotores o la temperatura en cada rotor.

En el trabajo a futuro se recomienda realizar experimentos con datos obtenidos a partir de acelerómetros externos acoplados a cada uno de los rotores, y a partir de estos datos implementar el algoritmo propuesto. También se deben explorar otras metodologías del aprendizaje automático, como en Jiang y cols. (2015) donde se aplica redes neuronales artificiales para la

identificación de diversas fallas, como una hélice dañada, fracturada o distorsionada, con base en las señales de vibraciones, que se obtuvieron de una IMU incorporada en un *smartphone*, colocado en el centro del marco del vehículo. Otro aspecto a evaluar es el desempeño de la metodología propuesta ante otras situaciones que afecten el desempeño, como en Yap (2014), donde además de la fractura de la hélice, se analizan las fallas provocadas por los tornillos sueltos, o el rotor dañado en su totalidad. Es necesario analizar las tres etapas propuestas por Ghalamchi y Mueller (2018), que se muestran en la Figura 5.2. Así como las trayectorias de vuelo en el VANT (Fig.5.1). Se establece que para la detección de la falla en una de las hélices, se realizan cuatro trayectorias de vuelo (S1, S2, S3 y S4). Para las fallas en dos o más hélices se realizan las etapas anteriores más otras cuatro etapas de vuelo diagonal (S5, S6, S7 y S8). También se puede mejorar este procedimiento acoplando **ROS** con **MATLAB** ®y haciendo el experimento en tiempo real, con el objetivo de que el análisis sea funcional en un vuelo bajo condiciones no controladas, donde el vehículo sea suceptible a perturbaciones externas y no en un ambiente de laboratorio controlado. Una vez logrado en su totalidad el objetivo, y con base en esta investigación, se abre la oportunidad de combinar el enfoque basado en datos con sistemas de control tolerante a fallas, que permitan realizar un vuelo plenamente autónomo y con la capacidad de reaccionar ante la presencia de fallas o perturbaciones.

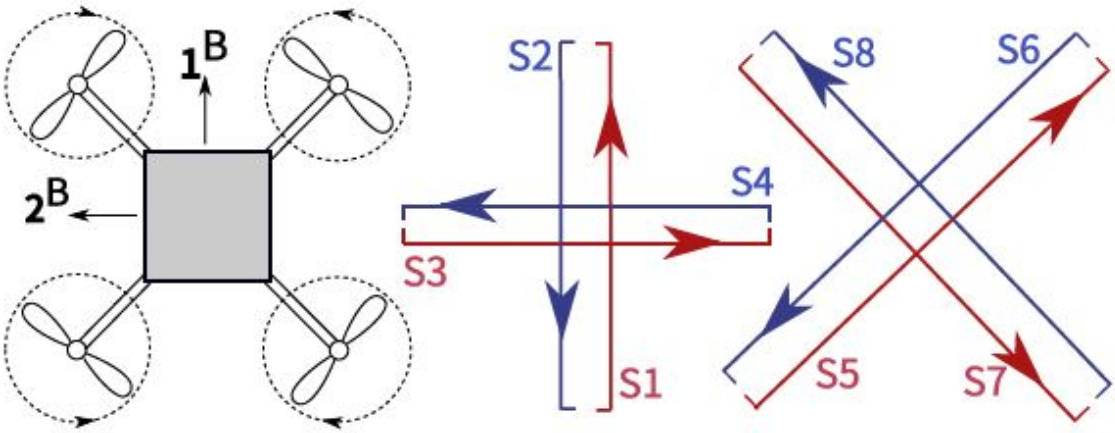


Figura 5.1: Etapas de vuelo propuestas para la identificación de fallas (Ghalamchi y Mueller, 2018)

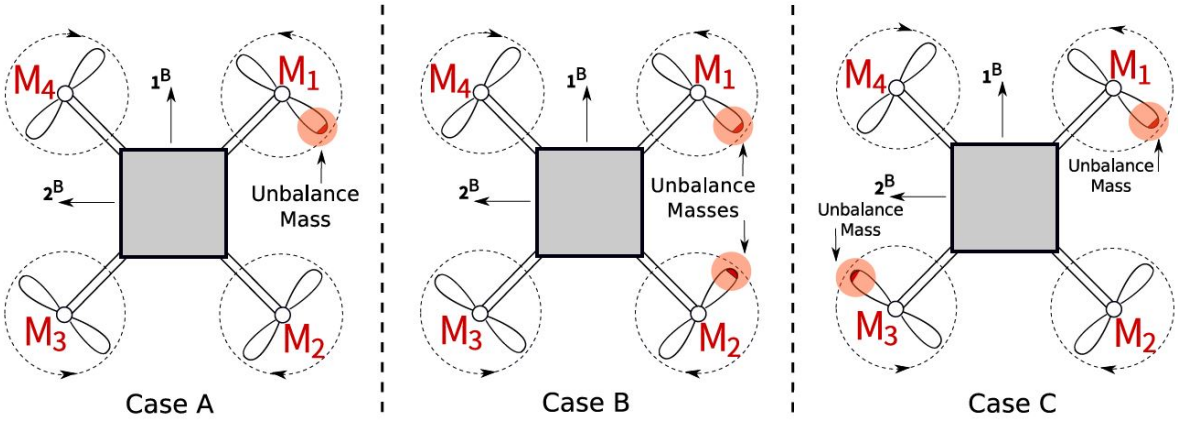


Figura 5.2: Casos de estudio propuestos en Ghalamchi y Mueller (2018): Caso A, una sola hélice dañada, Caso B, dos hélices dañadas sobre el mismo extremo, Caso C, dos hélices dañadas en extremos opuestos.

Bibliografía

Baskaya, E., Bronz, M., y Delahaye, D. (2017). Fault detection & diagnosis for small uavs via machine learning. En *Digital avionics systems conference (dasc), 2017 ieee/aiaa 36th* (pp. 1–6).

Betancourt, G. A. (2005). Las máquinas de soporte vectorial (svms). *Scientia et technica*, *1*(27).

Blanke, M., Kinnaert, M., Lunze, J., Staroswiecki, M., y Schröder, J. (2006). *Diagnosis and fault-tolerant control* (Vol. 2). Springer.

Burges, C. J. (1998). A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, *2*(2), 121–167.

Cavoukian, A. (2012). *Privacy and drones: Unmanned aerial vehicles*. Information and Privacy Commissioner of Ontario, Canada Ontario.

Ding, S., Zhang, P., Ding, E., Naik, A., Deng, P., y Gui, W. (2010). On the application of pca technique to fault diagnosis. *Tsinghua Science and Technology*, *15*(2), 138–144.

Freeman, P., Pandita, R., Srivastava, N., y Balas, G. J. (2013). Model-based and data-driven fault detection performance for a small uav. *IEEE/ASME Transactions on mechatronics*, *18*(4), 1300–1309.

Gertler, J. (2013). *Fault detection and diagnosis*. Springer.

Gertler, J. (2015). Fault detection and diagnosis. *Encyclopedia of Systems and Control*, 417–422.

Ghalamchi, B., y Mueller, M. (2018). Vibration-based propeller fault diagnosis for multicopters. En *2018 international conference on unmanned aircraft systems (icuas)* (pp. 1041–1047).

GUI, W.-h., y LIU, X.-y. (2002). Fault diagnosis technologies based on artificial intelligence for complex process [j]. *Basic Automation*, *4*, 000.

Guzmán-Rabasa, J., Lóopez-Estrada, F., González-Contreras, B., Valencia-Palomo, G., Chadli, M., y Pérez-Patricio, M. (2019). Actuator fault detection and isolation ona quadrotor uav modeled as a lpv system. *Measurement and Control*, In Press. DOI: 10.1177/0020294018824764.

Hagenblad, A., Gustafsson, F., y Klein, I. (2003). A comparison of two methods for stochastic fault detection: the parity space approach and principal components analysis. *IFAC Proceedings Volumes*, *36*(16), 1053–1058.

Iannace, G., Ciaburro, G., y Trematerra, A. (2019). Fault diagnosis for uav blades using artificial neural network. *Robotics*, *8*(3), 59.

Jiang, Y., Zhiyao, Z., Haoxiang, L., y Quan, Q. (2015). Fault detection and identification for quadrotor based on airframe vibration signals: a data-driven method. En *2015 34th chinese control conference (ccc)* (pp. 6356–6361).

Jolliffe, I. (2011). *Principal component analysis*. Springer.

Li, M., Li, G., y Zhong, M. (2016). A data driven fault detection and isolation scheme for uav flight control system. En *Control conference (ccc), 2016 35th chinese* (pp. 6778–6783).

López Bautista, M. (2018). El salto cualitativo de deep learning en problemas de clasificación.

Martinez, W. L., y Martinez, A. R. (2015). *Computational statistics handbook with matlab*. Chapman and Hall/CRC.

Maurya, M. R., Rengaswamy, R., y Venkatasubramanian, V. (2005). Fault diagnosis by qualitative trend analysis of the principal components. *Chemical Engineering Research and Design*, *83*(9), 1122–1132.

Maurya, M. R., Rengaswamy, R., y Venkatasubramanian, V. (2007). Fault diagnosis using dynamic trend analysis: A review and recent developments. *Engineering Applications of artificial intelligence*, *20*(2), 133–146.

Mina, J., y Verde, C. (2004). Detección de fallas usando análisis de componentes principales. En *Instituto de ingeniera, unam congreso anual de la amca.*

Mouloua, M., Gilson, R., Kring, J., y Hancock, P. (2001). Workload, situation awareness, and teaming issues for uav/ucav operations. En *Proceedings of the human factors and ergonomics society annual meeting* (Vol. 45, pp. 162–165).

Mueller, M. W., y D'Andrea, R. (2014). Stability and control of a quadrocopter despite the complete loss of one, two, or three propellers. En *2014 ieee international conference on robotics and automation (icra)* (pp. 45–52).

Nomikos, P., y MacGregor, J. F. (1995). Multivariate spc charts for monitoring batch processes. *Technometrics*, *37*(1), 41–59.

Nonami, K., Kendoul, F., Suzuki, S., Wang, W., y Nakazawa, D. (2010). Introduction. En *Autonomous flying robots* (pp. 1–29). Springer.

Puig, V., Quevedo, J., Escobet, T., Morcego, B., y Ocampo, C. (2004). Control tolerante a fallos (parte i): Fundamentos y diagnóstico de fallos. *Revista Iberoamericana de automática e informática industrial*, *1*(1), 15–31.

Qin, S. J. (2009). Data-driven fault detection and diagnosis for complex industrial processes. *IFAC Proceedings Volumes*, *42*(8), 1115–1125.

Qin, S. J. (2012). Survey on data-driven industrial process monitoring and diagnosis. *Annual reviews in control*, *36*(2), 220–234.

Russell, E. L., Chiang, L. H., y Braatz, R. D. (2012). *Data-driven methods for fault detection and diagnosis in chemical processes.* Springer Science & Business Media.

Saied, M., Lussier, B., Fantoni, I., Francis, C., Shraim, H., y Sanahuja, G. (2015). Fault diagnosis and fault-tolerant control strategy for rotor failure in an octorotor. En *Ieee international conference on robotics and automation* (pp. 5266–5271).

Santos-Ruiz, I., López-Estrada, F.-R., Puig, V., Blesa, J., y Javadiha, M. (2019). Localización de fugas en redes de distribución de agua mediante k-nn con distancia cosenoidal. *Asociación de México de Control Automático*.

Schlechtingen, M., y Santos, I. F. (2011). Comparative analysis of neural network and regression based condition monitoring approaches for wind turbine fault detection. *Mechanical systems and signal processing*, *25*(5), 1849–1875.

Sharifi, F., Mirzaei, M., Gordon, B. W., y Zhang, Y. (2010). Fault tolerant control of a quadrotor uav using sliding mode control. En *2010 conference on control and fault-tolerant systems (systol)* (pp. 239–244).

Strang, G., Strang, G., Strang, G., y Strang, G. (2016). *Introduction to linear algebra* (Vol. 3). Wellesley-Cambridge Press Wellesley, MA.

Sun, R., Cheng, Q., Wang, G., y Ochieng, W. (2017). A novel online data-driven algorithm for detecting uav navigation sensor faults. *Sensors*, *17*(10), 2243.

Tamura, M., y Tsujita, S. (2007). A study on the number of principal components and sensitivity of fault detection using pca. *Computers & Chemical Engineering*, *31*(9), 1035–1046.

Valencia-Palomo, G., Villanueva-Grijalba, O., y Robles-Ríos, R. (2018). *Device for the pose measurement and test of control algoritms for unmanned aerial vehicles.* Mexican Institute of Intellectual Protection. (Mexican Patent Pending App. MX/a/2017/005377)

Vapnik, V. (1995). The nature of statistical learning theory· 6·[mj new york. *Springer-Verlag*, *1*, 995.

Vey, D., y Lunze, J. (2016). Experimental evaluation of an active fault-tolerant control scheme for multirotor uavs. En *2016 3rd conference on control and fault-tolerant systems (systol)* (pp. 125–132).

Wang, B., Chen, Y., Liu, D., y Peng, X. (2018). An embedded intelligent system for on-line anomaly detection of unmanned aerial vehicle. *Journal of Intelligent & Fuzzy Systems*, *34*(6), 3535–3545.

Wang, H., Chai, T.-Y., Ding, J.-L., y Brown, M. (2009). Data driven fault diagnosis and fault tolerant control: some advances and possible new directions. *Acta Automatica Sinica*, *35*(6), 739–747.

Willsky, A. S. (1976). A survey of design methods for failure detection in dynamic systems. *Automatica*, *12*(6), 601–611.

Wold, S. (1978). Cross-validatory estimation of the number of components in factor and principal components models. *Technometrics*, *20*(4), 397–405.

Xian, B., y Hao, W. (2019). Nonlinear robust fault-tolerant control of the tilt trirotor uav under rear servo's stuck fault: Theory and experiments. *IEEE Transactions on Industrial Informatics*, *15*(4), 2158–2166.

Yap, Y. K. (2014). Structural health monitoring for unmanned aerial systems. *EECS., UNC, BerNley, Rep. UCB/EECS-2014-70*.

Zhang, Y., y Jiang, J. (2008). Bibliographical review on reconfigurable fault-tolerant control systems. *Annual reviews in control*, *32*(2), 229–252.

Anexo A

Programación en MATLAB

Análisis PCA

Los pasos que se realizaron en la programación del algoritmo se muestran en la Figura A.1, donde en una primera instancia se obtuvieron los datos de las aceleraciones en el vehículo, para después realizar el PCA en Matlab. Con la extracción de las características, se tomaron a éstas como las nuevas entradas para los clasificadores de datos, y con ello obtener como salida la decisión del sistema para la FDI.

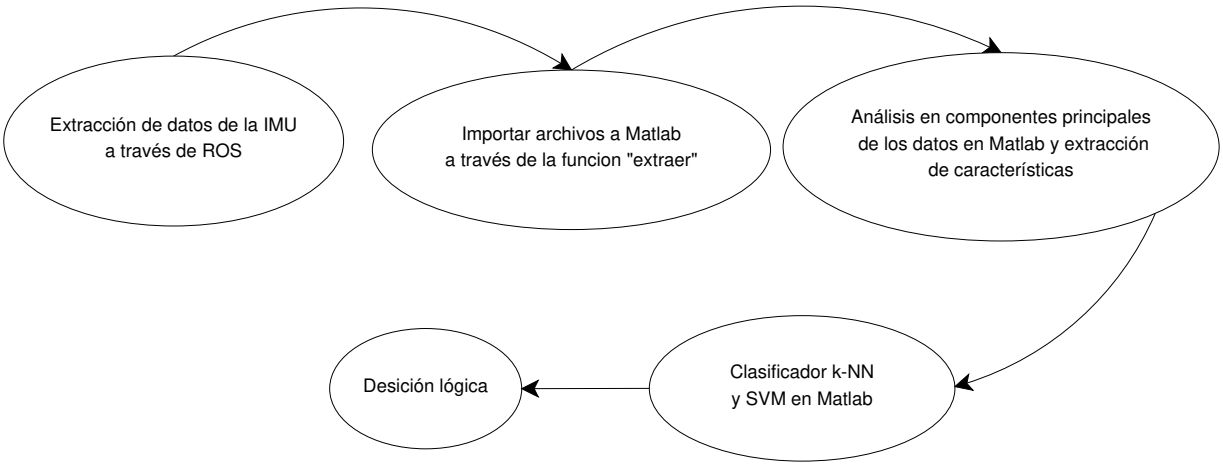


Figura A.1: Diagrama del proceso para la programación del algoritmo FDI propuesto

```
1 %codigo para el analisis y graficacion con PCA
```

```
[t,x,y,z,w,wx,wy,wz,ax,ay,az] = extraer('Datos_nominales.txt');
t= 0:1:size(t)-1;
[Z,mu,sigma] = zscore([x,y,z,w,wx,wy,wz,ax,ay,az]);
Matriz_covarianza = cov(Z);
[P,T,lambda,T2,explained] = pca(Z);
scatter3(T(:,1),T(:,2),T(:,3),'*k')
hold on
theta=linspace(0,2*pi,361);
xx=mean(T(:,2))+3*std(T(:,2))*cos(theta);
yy=mean(T(:,3))+3*std(T(:,3))*sin(theta);
plot(xx,yy,'k--')
%
[t1,x1,y1,z1,w1,wx1,wy1,wz1,ax1,ay1,az1] = extraer('Fr1.txt');%
    fallarotor1
xnew1=[x1,y1,z1,w1,wx1,wy1,wz1,ax1,ay1,az1];
    znew1 = zeros(size(xnew1));
    tnew1 = zeros(size(xnew1));
        for i=1:length(tnew1)
            xprueba1= xnew1(i,:);
             znew1 = (xnew1-mu)./sigma;
         end
tnew1 = znew1*P;%datos normalizados en terminos de la nueva base
scatter3(tnew1(:,1),tnew1(:,2),tnew1(:,3),'*b')%graficacion en terminos
    de la nueva base
%
[t2,x2,y2,z2,w2,wx2,wy2,wz2,ax2,ay2,az2] = extraer('Fr2.txt');%
    fallarotor2
xnew2=[x2,y2,z2,w2,wx2,wy2,wz2,ax2,ay2,az2];
    znew2 = zeros(size(xnew2));
    tnew2 = zeros(size(xnew2));
```

```
29          for i=1:length(tnew2)
30              xprueba2= xnew1(i,:);
31               znew2 = (xnew2-mu)./sigma;
32           end
33 tnew2 = znew2*P;
34 scatter3(tnew2(:,1),tnew2(:,2),tnew2(:,3),'*b')
35 %
36 [t3,x3,y3,z3,w3,wx3,wy3,wz3,ax3,ay3,az3] = extraer('Fr3.txt');%
       fallarotor3
37 xnew3=[x3,y3,z3,w3,wx3,wy3,wz3,ax3,ay3,az3];
38     znew3 = zeros(size(xnew3));
39     tnew3 = zeros(size(xnew3));
40          for i=1:length(tnew3)
41              xprueba3= xnew3(i,:);
42               znew3 = (xnew3-mu)./sigma;
43           end
44 tnew3 = znew3*P;
45 scatter3(tnew3(:,1),tnew3(:,2),tnew3(:,3),'*b')
46 %
47 [t4,x4,y4,z4,w4,wx4,wy4,wz4,ax4,ay4,az4] = extraer('Fr3.txt');%
       fallarotor4
48 xnew4=[x4,y4,z4,w4,wx4,wy4,wz4,ax4,ay4,az4];
49     znew4 = zeros(size(xnew4));
50     tnew4 = zeros(size(xnew4));
51          for i=1:length(tnew4)
52              xprueba4= xnew4(i,:);
53               znew4 = (xnew4-mu)./sigma;
54           end
55 tnew4 = znew4*P;
56 scatter3(tnew4(:,1),tnew4(:,2),tnew4(:,3),'*b')
```

```
57
58 xlabel('Componente principal P_1')
59 ylabel('Componente principal P_2')
60 zlabel('Componente principal P_3')
```

Función extraer

```
1  %codigo para extraer los datos de la imu.txt
2  function [t,x,y,z,w,wx,wy,wz,ax,ay,az] = extraer(filename)
3  %    Author: Ildeberto de los Santos Ruiz
4  %        idelossantos@ittg.edu.mx
5  fid = fopen(filename);
6  t = [];
7  x = [];
8  y = [];
9  z = [];
10 w = [];
11 wx = [];
12 wy = [];
13 wz = [];
14 ax = [];
15 ay = [];
16 az = [];
17 s_old = fgetl(fid);
18 while ~feof(fid)
19     s = fgetl(fid);
20     if length(s) > 6
21         if strcmp(s(1:6),'  seq:')
22             t = [t;str2num(s(7:end))];
23         end
```

```
24          if strcmp(s(1:4),'  x:') && strcmp(s_old,'orientation: ')
25              x = [x;str2num(s(5:end))];
26              s = fgetl(fid);
27              y = [y;str2num(s(5:end))];
28              s = fgetl(fid);
29              z = [z;str2num(s(5:end))];
30              s = fgetl(fid);
31              w = [w;str2num(s(5:end))];
32          end
33          if strcmp(s(1:4),'  x:') && strcmp(s_old,'angular_velocity: ')
34              wx = [wx;str2num(s(5:end))];
35              s = fgetl(fid);
36              wy = [wy;str2num(s(5:end))];
37              s = fgetl(fid);
38              wz = [wz;str2num(s(5:end))];
39          end
40          if strcmp(s(1:4),'  x:') && strcmp(s_old,'linear_acceleration: '
                )
41              ax = [ax;str2num(s(5:end))];
42              s = fgetl(fid);
43              ay = [ay;str2num(s(5:end))];
44              s = fgetl(fid);
45              az = [az;str2num(s(5:end))];
46          end
47      end
48      s_old = s;
49  end
```

Clasificador knn, svm

```
[t,x,y,z,w,wx,wy,wz,ax,ay,az] = extraer('Datos sin falla.txt');
xa=[x,y,z,w,wx,wy,wz,ax,ay,az];
m1 = zeros(size(wx))*5;
[z,mu,sigma] = zscore(xa);
P = pca(z);
%
[t1,x1,y1,z1,w1,wx1,wy1,wz1,ax1,ay1,az1] = extraer('Fr1.txt');
xnew2=[x1,y1,z1,w1,wx1,wy1,wz1,ax1,ay1,az1];
m2=ones(size(wx1));
%
[t3,x3,y3,z3,w3,wx3,wy3,wz3,ax3,ay3,az3] = extraer('Fr2.txt');
xnew3=[x3,y3,z3,w3,wx3,wy3,wz3,ax3,ay3,az3];
m3=ones(size(wx3))*2;

[t4,x4,y4,z4,w4,wx4,wy4,wz4,ax4,ay4,az4] = extraer('Fr3.txt');
xnew4=[x4,y4,z4,w4,wx4,wy4,wz4,ax4,ay4,az4];
m4=ones(size(wx4))*3;
%
[t5,x5,y5,z5,w5,wx5,wy5,wz5,ax5,ay5,az5] = extraer('Fr4.txt');
xnew5=[x5,y5,z5,w5,wx5,wy5,wz5,ax5,ay5,az5];
m5=ones(size(wx5))*4;
%
x = [xa;xnew2;xnew3;xnew4;xnew5];
y = [m1;m2;m3;m4;m5];
z = (x-mu)./sigma;
T = z*P;
%%
modelo = fitcknn(T(:,1:5),y,'NumNeighbors',3);
%modelo = fitcecoc(T(:,1:5),y,'Learner',templateSVM('KernelFunction','
    gaussian'));
```

```
30
31 modelo.resubLoss()
32 yhat = modelo.predict(T(:,1:5));
33 ConfM=confusionmat(y,yhat)
34 %%
35 xnew6=[x6,y6,z6,w6,wx6,wy6,wz6,ax6,ay6,az6];
36       znew = (xnew6-mu)./sigma;
37     Tnew = znew*P;
38 for i=1:length(Tnew)
39                 xprueba= Tnew(i,:);
40                 ynew(i) = modelo.predict(xprueba(1:5));
41 end
```

Programación del Ejemplo de clasificación de vecinos más cercanos

```
%Esta programacion se encuentra en la pagina oficial de MATLAB en la direccion:https://la.mathworks.com/help/stats/classification-using-nearest-neighbors.html

load fisheriris
x = meas(:,3:4);
gscatter(x(:,1),x(:,2),species);
newpoint = [5 1.45]; line(newpoint(1),newpoint(2),'marker','x','color','k','markersize',10,'linewidth',2)
Mdl = KDTreeSearcher(x);
[n,d] = knnsearch(Mdl,newpoint,'k',10); line(x(n,1),x(n,2),'color',[.5 .5 .5],'marker','o','linestyle','none','markersize',10)
xlim([4.5 5.5]); ylim([1 2]);
axis square
ctr = newpoint - d(end); diameter = 2*d(end);
 h = rectangle('position',[ctr,diameter,diameter],'curvature',[1 1]); h.LineStyle = ':';
 legend({'','versicolor','virginica','dato prueba','zona de aceptacion'})
```

Anexo B

Obtención de datos de la imu en Erle-brain3 a través de ROS

Montar la imagen del S.O frambuesa en Windows

1. Descargar e instalarlos siguientes softwares en windows: SD Formatter y rufus.
2. Abrir el programa SD Formatter e introducir una memoria micro-SD clase 10 en la PC.
3. Elegir y formatear la unidad de memoria asignada a la tarjeta SD. Una vez completado el proceso, cerrar el programa.
4. Abrir rufus, seleccionar la unidad de memoria asignada a la tarjeta SD y buscar el disco virtual (imagen ISO) que contiene el sistema operativo solicitado.
5. Grabar el sistema operativo en la memoria SD y esperar a que el proceso finalice (suele demorarse unos minutos).
6. Introducir la tarjeta SD con el S.O Frambuesa en la raspberry pi3

Conexión por SSH con la tarjeta Erle-Brain3

Es necesario alimentar la tarjeta Erle-brain3 con 5v. La computadora se conecta a la red wifi de la tarjeta denominada ”erle-brain” y por medio de un cmd en Windows o en caso de Ubuntu

una terminal, nos conectamos vía ssh mediante los siguientes comandos:

- **ssh erle@erle-brain**

La contraseña es *holaerle*.

Instalacion APM planner

Ardupilot (APM): Es el software (firmware) que se ejecuta en nuestra controladora de vuelo. Es el encargado de gestionar toda la información que recibe del exterior (sensores físicos y radio) y actuar en consecuencia sobre los actuadores de la aeronave. Actualmente se puede descargar desde su página web oficial o incluso contribuir a su mejora (o crear versiones propias) mediante GitHub y su repositorio oficial. Este firmware nativo sólo puede emplearse en plataformas soportadas y mantenidas oficialmente por sus creadores (pese a que son Open Source y Open Hardware respectivamente) como son las controladoras de vuelo APM (en sus distintas versiones), PX4 FMU o Pixhawk. Erle Robotics se encuentran entre los principales contribuyentes del código fuente de Ardupilot. Al Arrancar la raspberry solo conectando la alimentación, una vez cargado el sistema operativo, se realiza la actualización del sistema operativo, para ello, se abre la terminal (consola de comandos), una vez disponible la terminal se escribe y ejecutan los siguientes comandos:

- **sudo apt-get update**
- **sudo apt-get upgrade**

Este proceso tiene un tiempo de duración de algunos minutos. Para que este proceso se realice exitosamente tambíen hay que realizar el proceso de actualización de las llaves para ello se ejecuta el siguiente comando:

- **curl packages.erlerobotics.com/keys/deploy | sudo bash**

Terminado este proceso tendremos instalado correctamente el APM copter dentro de la erle brain. El APT ayuda a construir varios vehículos autónomos (Copters, Planos, Rovers y

HexaCopters). Para buscar estos controladores, podemos escribir el siguiente código en la ventana de comandos:

- **sudo apt-cache search erle-brain**

Y se debe desplegar una lista de los controladores para los diferentes tipos de controladores de vehículos que APM tiene para la tarjeta Erle-brain3. Para instalar el controlador de vuelo del vehículo cuatrimotor se escribe en la ventana de comandos:

- **sudo apt-get install apm-copter-erlebrain**

Cuando se haya finalizado el proceso, se tiene que reiniciar el sistema:

- **sudo reboot**

Cuando se haya reiniciado, es posible iniciar la puesta en marcha del cuatrimotor mediante el control del vehículo.

Obtención de datos

Es posible acceder a los datos que proporciona la IMU que tiene integrada la tarjeta Erle-brain3, a través de **ROS** (*Robotic System Operating*). ROS provee librerías y herramientas para ayudar a los desarrolladores de software a crear aplicaciones para robots. Además ROS proporciona abstracción de hardware, controladores de dispositivos, librerías, herramientas de visualización, comunicación por mensajes, administración de paquetes y más. ROS está bajo la licencia open source, BSD.

Para acceder a los datos de la IMU primero debemos conectarnos a la red wifi de la tarjeta Erle-brain3. Y entablar la comunicación por ssh desde el cmd. Una vez que esto se halla realizado, se tiene que establecer en el cliente la URI del master (el erle brain) como variable de entorno. Si se ha conectado al wifi hotspot del brain, su IP por defecto será 10.0.0.1:

- **export ROS_MASTER_URI=http://10.0.0.1:11311**

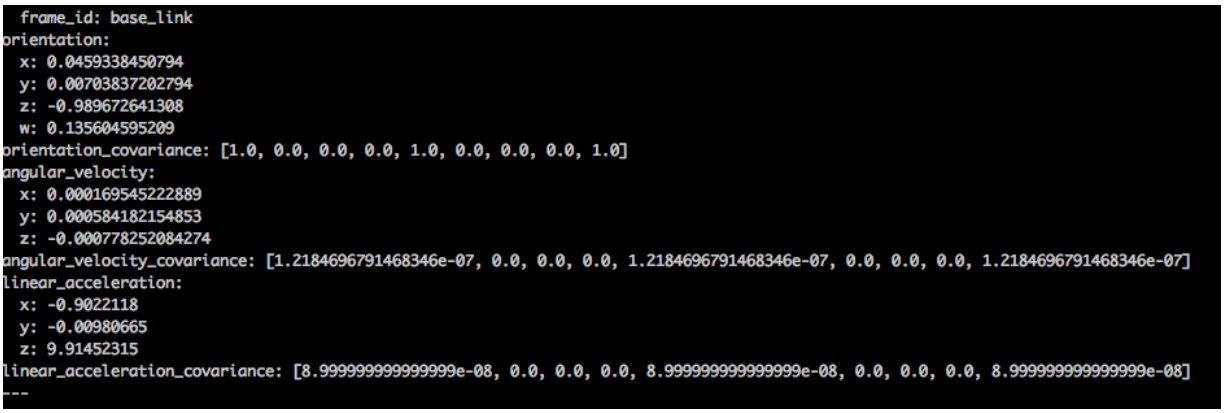


Figura B.1: Datos obtenidos de la IMU

Para conectarse al nodo de ROS que nos permite visualizar los datos de la IMU, se tiene que habilitar la publicación de estos mensajes, para ello tienes que llamar al servicio de mavros de set_stream_rate:

- **rosservice call /mavros/set_stream_rate 0 10 1**

Con este paso se habilitará (el 1) el stream rate de todos los tipos de topico (el 0), a 10 hercios (el 10). Después de esto se visualizan los datos de la IMU a través de la instrucción:

- **rostopic echo /mavros/imu/data**

Este nodo proporciona datos de posición, velocidad angular y aceleración lineal B.1, datos que fueron de interés para el análisis propuesto en este trabajo.

B.1. Componentes básicos del vehículo Erle-copter

El vehículo utilizado para este experimento es el erlecopter, un VANT cuatrimotor que tiene como tarjeta de control la Erle Brain 3, que se muestra en la Fig. B.2.

Dicha tarjeta está basado en Linux y da soporte al dron con *Robot Operating System*, o comúnmente conocido como **ROS**. Incluye sensores, periféricos y la electrónica necesaria para facilitar autonomía a los robots. Las principales especificaciones técnicas del Erle-Brain se muestran en la Tabla B.1.

Figura B.2: Vehículo cuatrimotor con tarjeta Erle-brain3

Para poder armar el vehículo Erle-copter, es inherente la necesidad de utilizar ciertos componentes mecánicos y electrónicos que se mencionan a continuación.

El *frame* o marco del vehículo, es la estructura que da soporte al resto de elementos que formarán el vehículo en conjunto. Éste se encuentra compuesto por cuatro brazos y dos placas metálicas. En los extremos de cada brazo se ubican los motores. Las dimensiones estándar son de una longitud de motor a motor de 450mm. La estructura pesa 280 gramos.

Los motores son los encargados de convertir la energía eléctrica en mecánica y transmitir

Figura B.3: Marco del vehículo

dicha energía a las hélices. Para este vehículo se utilizaron los motores Emax2212. Este tipo de motores son conocidos como *brushless DC*, es decir, motores de corriente continua sin escobillas. Están formados por un rotor donde se encuentran los imanes permanentes por un estator, la parte fija donde se encuentran los bobinados del hilo conductor. Las dimensiones

Tabla B.1: Especificaciones técnicas Erle Brain 3

Característica	**Valor**
Procesador	1.2 GHz 64-bit quad-core ARMv8
Gráficos	VideoCore IV 3D gráficos core/S
Memoria SDRAM	1 GB RAM
Fuente de alimentación	MicroUSB
Puertos Host	4x USB 2.0
Eternet	10/100, RJ45
WiFi	802.11 n
Bluetooth	Bluetooth 4.1
Conector SD/MMC	microSD, 3.3 V
Salida de vídeo	Puerto Full HDMI
Audio	3.5 mm audio jack

del estator son de 22x13mm. La ventaja de los motores brushless respecto a los motores con escobillas es que tienen una mayor eficiencia, mayor vida útil, menos peso y menos interferencia electromagnética.

Las hélices son las encargadas de convertir el movimiento rotatorio proveniente, de los

Figura B.4: Motores brushles Emax2212

motores eléctricos en una fuerza propulsora. Las hélices son asimétricas dos a dos. Se diferencian por su sentido de giro según al motor al que se aplican. Dado que la configuración del cuatrimotor es en "X" el sentido de giro en las hélices se muestra en la Figura B.5.

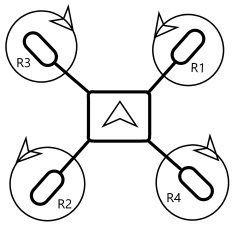


Figura B.5: Sentido de giro en las hélices de los motores en un cuatrimotor en configuración "X"

Los controles electrónicos de velocidad (ESC), también conocidos como variadores elec-

trónicos, proporcionan una fuente trifásica. Se trata de un circuito electrónico cuya funcion es variar y controlar la rotación de los motores. Cada ESC dispone de dos terminales de alimentación a la batería, tres terminales de aplicación al BLDC y un conector para aplicación de señales PWM proporcionadas por la tarjeta Erle-brain.

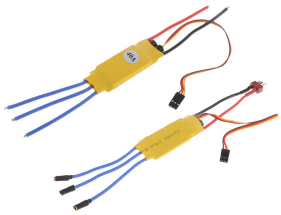


Figura B.6: controladores electrónicos de velocidad (ESC)

La batería que se utiliza para alimentar al vehículo es de Polímero Litio de 500 mAh y 11.1 V. la cual dota al vehículo de una autonomía de aproximadamente 20 minutos. El mando de radiocontrol, RC, permite controlar el vuelo del don de manera remota. Para nuestro caso se dispone de un radiocontrol modelo Turningy TGY-i6 el cual transmite a 2.4 GHz y dispone de seis canales. Con estos seis canales se controlan a los tres principales movimientos de rotación en el vehículo denominados alabeo, cabeceo y guiñada; además de controlar el acelerador. Este viene acompañado de un receptor que se conecta al vehículo.

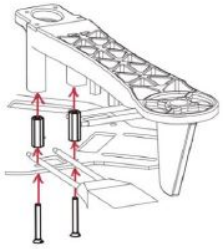
(a) radiocontrol **(b)** receptor

Figura B.7: Elementos para la comunicación transmisor-receptor de un VANT

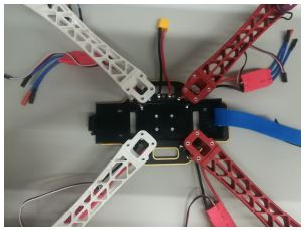
B.2. Montaje del vehículo Erle-copter

Los elementos que conforman la arquitectura del VANT son los que se mencionaron en la sección anterior. Además de ellos son necesarios elementos como: tornillos, embellecedores,

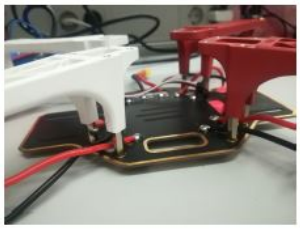
tuercas, arandelas, bridas y cinta adhesiva doble cara. Como primer paso se fijan los cuatro brazos a la parte metálica inferior del chasis mediante ocho tornillos embellecedores. Los brazos de color rojo indicarán la parte delantera del vehículo, mientras que los brazos blancos indicarán la parte trasera.



(a) Montaje de los brazos



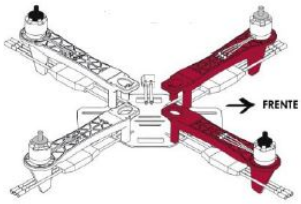
(b) Vista superior del montaje de los brazos



(c) Ensamble de los brazos con los embellecedores

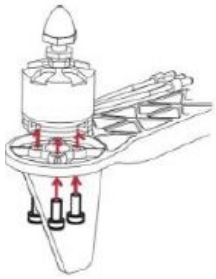
Figura B.8: Descripción y montaje de los brazos del vehículo

Como segundo paso se colocan los cuatro rotores, uno por cada brazo, y se fijan a cada uno con cuatro tornillos, tal y como se muestra en la Figura B.9.



(a) Esquema de montaje de rotores

(b) Vista superior del montaje de rotores



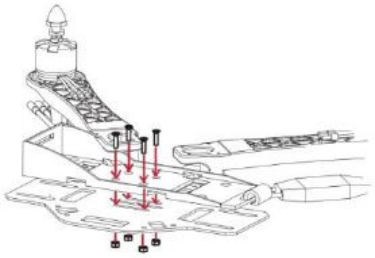
(c) fijación de los tornillos

Figura B.9: Descripción y montaje de los rotores del vehículo

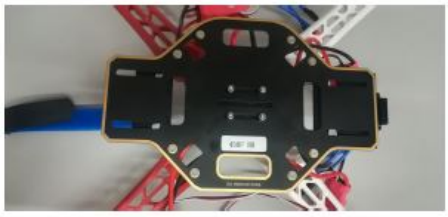
En seguida se monta el soporte para la batería en la parte inferior metálica del chasis con cuatro tornillos y cuatro tuercas como se muestra en la Fig. B.10a. Se atornilla la parte superior metálica del chasis tal y como se aprecia en la Figura B.10. Esta estructura servirá como base al Erle-brain.

A continuación, con la cinta adhesiva doble cara se adhiere la Erle-brain a la parte superior del chasis dentro del marco del vehículo, fijándose completamente.

Como paso siguiente se conectan los ESC a los motores, teniendo cuidado ya que esta

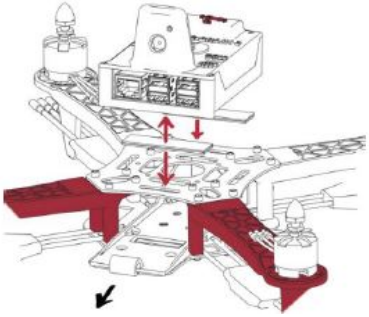


(a) Esquema de montaje de soporte



(b) Vista superior del montaje de soporte

Figura B.10: Montaje del soporte de la batería

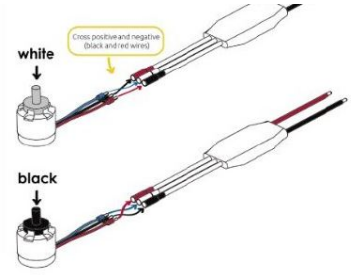


(a) Esquema de montaje de Erle-brain al chasis

(b) colocación de Erle-brain

Figura B.11: Montaje de Erle-brain en la parte superior del chasis, dentro del marco del vehículo

conexión varía dependiendo el tipo de motor que se tenga y que el sentido de giro en cada motor sea el correcto. La conexión adecuada se muestra en la Figura B.12. Una vez que se han conectado los variadores a los motores, se fijan a los brazos del chasis mediante bridas para evitar daños durante el vuelo.



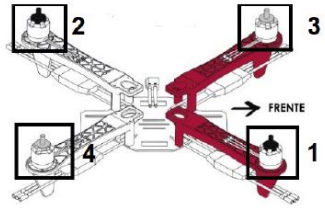
(a) Esquema de conexión de los ESC

(b) Conexión de los motores al ESC

Figura B.12: Conexión entre los ESC y los motores del vehículo

Posteriormente, se conectan los ESC a las salidas PWM del Erle-brain, según el esquema de la Figura B.13a, siendo el ESC con el número 1 conectado a la entrada PWM situada a

la derecha y el ESC con el número 4 a la izquierda. Los conectores deben quedar en orden ascendente y los cables colocados con la señal en la parte superior, positivo en el medio y negativo en la parte inferior (Figura B.13b).



(a) Numeración para la posición de los ESC

(b) Conexión de los ESC a la trajeta Erle-brain

Figura B.13: Conexión de los controladores electrónicos a la tarjeta Erle-brain

Se conecta el regulador de voltaje al chasis del vehículo mediante el conector XT-60 y al Erle-brain mediante el cable DF13 de 6 posiciones, como se muestra en la Figura B.14.

Se fija el receptor del mando de radiocontrol al lado del Erle-brain mediante una brida. Se conecta en el canal 14 de sus pines de manera que el cable naranja del conector quede en la parte superior. En la Figura B.15 se aprecia dicho montaje. A continuación se colocan las cuatro hélices. Para ello se deben hacer girar las hélices 1 y 2 en sentido antihorario y la hélices 3 y 4 en sentido horario (Figura B.16).

Para concluir con el montaje del vehículo se posiciona la batería con el soporte y se conecta ésta al modulador de voltaje a través del XT-60 como se muestra en la Figura B.17.

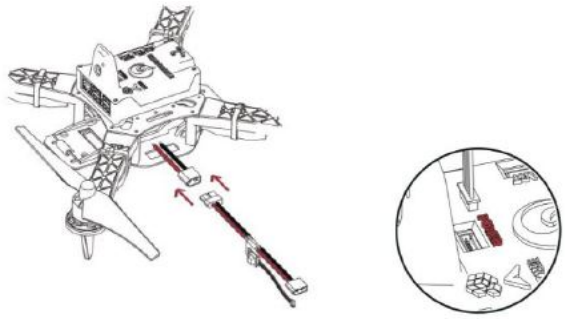


Figura B.14: Conexión del módulo regulador de voltaje a la tarjeta Erle-brain

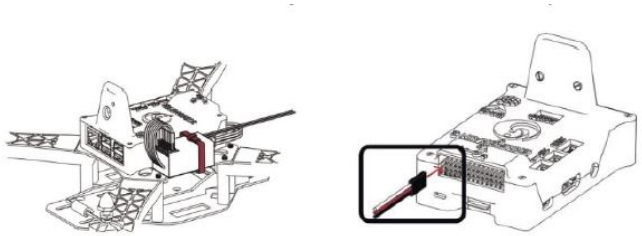


Figura B.15: Conexión del receptor de mando de radiocontrol

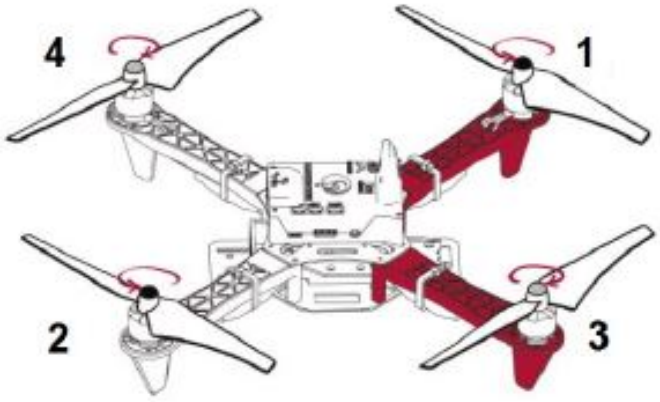


Figura B.16: Colocación de las hélices en el vehículo

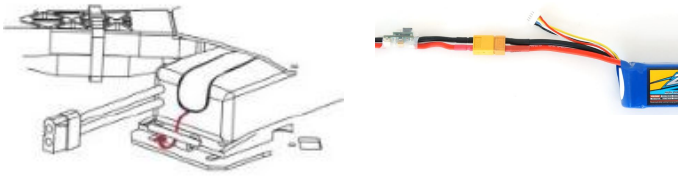


Figura B.17: Montaje de la batería al vehículo