



UNIVERSIDAD DE GUANAJUATO

CAMPUS IRAPUATO-SALAMANCA
DIVISION DE INGENIERÍAS

*“Análisis de Señales Mediante Cuaterniones: del Algoritmo
a la Implementación en Hardware”*

TESIS

QUE PARA OBTENER EL GRADO DE:
DOCTOR EN INGENIERÍA ELÉCTRICA

PRESENTA:

José Luis Contreras Hernández

ASESORES:

Dr. Mario Alberto Ibarra Manzano

Dra. Dora Luz Almanza Ojeda

SALAMANCA, GTO.

2020

Salamanca, Gto., a 20 de abril del 2020.

M. en I. H ERIBERTO GUTIÉRREZ MARTÍN
JEFE DE LA UNIDAD DE ADMINISTRACIÓN ESCOLAR
P R E S E N T E-

Por medio de la presente, se otorga autorización para proceder a los trámites de impresión, empastado de tesis y titulación al alumno(a) M. C. José Luis Contreras Hernández del *Programa de Doctorado en Ingeniería Eléctrica* y cuyo número de *NUA* es: 143233 del cual soy director. El título de la tesis es: Análisis de Señales mediante Cuaterniones: del algoritmo a la implementación en Hardware.

Hago constar que he revisado dicho trabajo y he tenido comunicación con los sinodales asignados para la revisión de la tesis, por lo que no hay impedimento alguno para fijar la fecha de examen de titulación .

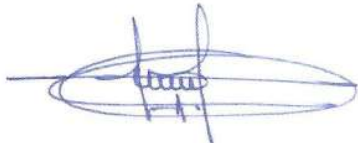
A T E N T A M E N T E



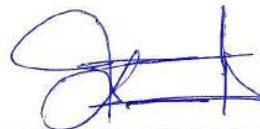
Dr. Mario Alberto Ibarra Manzano
DIRECTOR DE TESIS
SECRETARIO



Dra. Dora Luz Almanza Ojeda
DIRECTOR DE TESIS



Dr. Juan Gabriel Aviña Cervantes
PRESIDENTE



Dr. Carlos Hugo García Capulín
VOCAL



Dr. Juan Carlos Gómez Carranza
VOCAL



Dr. Martín Valtierra Rodríguez
VOCAL

A Martha y a nuestro futuro.

Afortunadamente tenemos algo por
aprender.

José Luis Contreras Hernández

Agradecimientos Personales

A Dios, por permitirme concluir una etapa más.

A Martha. Mi esposa, mi novia, mi mejor amiga, mi razón, mi fuerza y mi apoyo. No tengo palabras para agradecerte todo lo que ha hecho por mí. Por todo tu amor, paciencia, tiempo y esfuerzo. Gracias por ayudarme a ser mejor cada día y por ser mi compañera de vida.

A mis padres por su apoyo y crianza, sin ellos no hubiera llegado hasta acá (a los dos).

A mi hermana por su apoyo y ocurrencias, eres la mejor. Estoy orgulloso y feliz de ir caminando a tu lado. Me has inspirado mucho.

A mis abuelos, por ser parte de mi crianza, gracias de corazón.

A mis familiares que siempre han estado para apoyarme.

A mis suegros, por su apoyo y recibirme en su casa. He aprendido mucho de ustedes.

A mi familia política, que me recibieron con los brazos abiertos.

A mis asesores y padrinos, Dr. Mario y Dra. Dora. Por aguantarme, por tenerme paciencia y por toda la ayuda y confianza que me han brindado. No tengo como pagarles.

A Enrique, por darme la confianza de ser parte de la Estudiantina y por todo lo que me ha ayudado desde que estaba en la licenciatura.

A todos mis amigos y hermanos, por su confianza y ánimos.

A mis hermanos de la Estudiantina, los que están y los que estuvieron en alguna etapa. Por compartir grandes experiencias y escenarios de todo tipo. *Y mientras el cuerpo aguante seré tuno hasta morir.*

A todos mis compañeros de Laboratorio, que de todos he aprendido y a todos los llevo en el corazón.

A mis alumnos, que el aprender no es solamente de mí hacía ellos.

Agradecimientos Institucionales

Agradezco a la Universidad de Guanajuato por el apoyo durante el desarrollo de este trabajo de tesis. También por haberme permitido formarme en ella como alumno y como profesor. Termine este trabajo con la satisfacción de aportar un granito de arena para que siga su nombre en alto y del cual me siento orgulloso de portar.

Agradezco a todo el personal de la DICIS que de una u otra manera me ayudaron en el desarrollo de este trabajo de tesis. Mi División.

Gracias al Laboratorio de DSP por proporcionarme un espacio y equipo adecuado para trabajar de manera adecuada.

Agradezco al Consejo de Ciencia y Tecnología (CONACYT) por otorgarme la beca (487660) para realizar mis estudios de Doctorado en Ingeniería de la Universidad de Guanajuato.

Resumen

En el procesamiento digital de señales, el uso de cuaterniones ha incrementado debido a que ofrecen soluciones matemáticas manejables con pocas restricciones y con un número de parámetros reducido. Por lo cual, los cuaterniones son aplicados en procesamientos de imágenes, señales, cifrados y transformaciones espaciales.

En este trabajo de tesis se desarrolla el algoritmo de análisis de señales por medio de cuaterniones, el cual es implementado en hardware. De esta forma se obtiene un modelo de comportamiento mediante el análisis de la rotación de tres o cuatro señales simultáneas en el dominio de tiempo.

Con el fin de comprobar la estructura y funcionamiento del algoritmo, se clasifican tres estados de un motor de inducción mediante los valores estadísticos obtenidos de una ventana de al menos dos muestras de cuaterniones, los cuales están constituidos por la señal de corriente y tres señales producidas por los ejes de un acelerómetro colocado de forma no invasiva al motor. Los resultados de las estadísticas son clasificados mediante un árbol de decisión para obtener resultados de eficiencia.

El análisis favorable de los resultados de precisión obtenidos es realizado de manera general, comparando los resultados de los distintos estados del motor, cambiando características en el algoritmo de implementación y comparando resultados con trabajos que desarrollan sus propios algoritmos para clasificar estados de los motores de inducción.

Índice general

Resumen	VI
1. Introducción	5
1.1. Antecedentes	6
1.2. Objetivos	7
1.2.1. Objetivo General	7
1.2.2. Objetivos Particulares	7
1.3. Hipótesis	7
1.4. Justificación	8
1.5. Contribuciones	8
1.6. Organización del Documento	9
2. Revisión de Literatura	10
2.1. Estado del Arte	10
2.2. Marco Teórico	14
2.2.1. Cuaternión	14
2.2.2. Estadísticas	18
2.2.3. Árbol de Clasificación	18
2.2.4. Lógica Programable	20
3. Metodología	22
3.1. Algoritmo	22
3.2. Implementación	24
3.2.1. Normalización	25
3.2.2. Rotación de Cuaternión	28

<i>ÍNDICE GENERAL</i>	1
3.2.3. Extracción de Características Estadísticas	29
3.2.4. Clasificación	30
4. Pruebas y Resultados	32
4.1. Configuración Experimental	32
4.2. Resultados	36
4.2.1. Árboles de Decisión	39
4.2.2. Implementación en Matlab	40
4.2.3. Implementación en FPGA	44
5. Conclusiones y Perspectivas	50
5.1. Conclusiones	50
5.2. Perspectivas	51
A. Artículo IEEE Transactions on Industrial Electronics	58
B. Artículo Measurement	67
C. Artículo CONIMI	77

Índice de figuras

2.1. Rotación de cuaterniones.	16
2.2. Estructura del árbol de clasificación.	19
2.3. Proceso de árbol de clasificación.	20
2.4. Conformación a bloques de un FPGA	21
3.1. Algoritmo del análisis de señal cuaternión.	23
3.2. Diagrama general del sistema propuesto.	24
3.3. Representación interna del sistema.	25
3.4. Bloque normalización del cuaternión. a) Estructura general. b) Estructura interna del bloque de norma. c) Estructura interna del bloque raíz cuadrada. d) Estructura del bloque división.	28
3.5. Estructura de la norma de tres componentes.	29
3.6. Estructura interna del bloque estadístico.	29
3.7. Estructura interna del árbol de decisión.	30
4.1. Configuración experimental.	33
4.2. Configuración de fallas. a) Falla de balero. b) Falla de desbalance.	33
4.3. Prueba de señales. a) Señales de corriente. b) Señales de vibración para eje X. c) Señales de vibración para eje Y. d) Señales de vibración para eje Z.	34
4.4. Distribución de clasificación usando ventana de diez muestras.	37
4.5. Distribución de clasificación usando ventana de quince muestras.	37
4.6. Distribución de clasificación usando ventana de treinta muestras.	38
4.7. Distribución de clasificación usando ventana de cien muestras.	38
4.8. Distribución de clasificación usando ventana de doscientas muestras	39
4.9. Árbol de clasificación generado con ventana de cincuenta muestras.	40

4.10. Árbol de clasificación generado con ventana de doscientas muestras.	40
4.11. Exactitud general de clasificación.	41
4.12. Exactitud en la clasificación por estado.	42
4.13. Comparación de exactitud de clasificación general del QSA con baja frecuencia	44
4.14. Comparación de exactitudes de clasificación <i>BAL</i>	45
4.15. Comparación de exactitudes de clasificación <i>BRN</i>	45
4.16. Comparación de exactitudes de clasificación <i>HLT</i>	46
4.17. Tiempo computacional.	47
4.18. Recursos de memoria y el número de nodos del árbol de decisión en la implementación en FPGA.	49

Índice de tablas

4.1. Muestra de matriz de confusión utilizando ventana de 15 muestras.	35
4.2. Exactitud promedio, mínima y máxima de la clasificación.	43
4.3. Comparación de métodos en trabajos relacionados con el QSA en Matlab. . .	47
4.4. Comparación de métodos en trabajos relacionados con el QSA implementado en FPGA.	48
4.5. Comparación de recursos consumidos.	48

Introducción

Los motores de inducción son importantes en los procesos industriales. La detección temprana de fallas en motores es esencial para asegurar el tiempo de producción y planear el mantenimiento en la línea de manufactura cuando lo requiera, lo cual implica una pérdida menor en los costos. Las fallas en los motores de inducción pueden ser generadas por fuerzas electrodinámicas, aislamiento de la bobina, grandes tensiones, envejecimiento térmico y vibraciones mecánicas de fuentes internas o externas. Las fallas mecánicas más típicas de los motores de inducción son fallas en el estátor, de baleros, de barras y de rotor. [1].

Para la identificación de distintos tipos de fallas se han desarrollado métodos en los que se hacen uso de señales adquiridas de los motores. Los métodos presentan una amplia variedad en el espacio temporal y de frecuencia, con los cuales se detectan distintos tipos de fallas. Entre estos métodos, podemos encontrar los que están basados en propiedades estadísticas y que mediante algoritmos de clasificación logran detectar fallas en los motores. Se requiere de un corto tiempo de operación en los métodos debido a que los resultados necesitan detectarse en ocasiones sin desmontar los motores y se cuenta con poco tiempo para la prueba. Es por esto que un método que realice el procesamiento en tiempo real es una mejor opción.

En este proyecto de tesis se propone el análisis estadístico de cuatro señales provenientes del motor de inducción, las cuales son adaptadas al espacio de cuaterniones y a las que se le aplica un procesamiento geométrico para obtener un modelo de su comportamiento en el dominio del tiempo. Este análisis puede ser implementado en hardware lo que implica un bajo consumo de energía, bajo consumo de recursos y portabilidad a diferencia de los desarrollos en software.

1.1. Antecedentes

Distintos tipos de señales son producidas en la presencia de un fenómeno físico, las cuales cuentan con propiedades que permiten describir el fenómeno y sus características.

Generalmente, estas señales son obtenidas en un formato digital debido al gran auge de los sensores digitales, los cuales presentan costos bajos y manejo fácil. El procesamiento digital de señales (*Digital Signal Processing*, DSP) es las matemáticas, algoritmos y técnicas usadas para manipular las señales con el fin de obtener características que permitan describir sus propiedades. El *DSP* es una herramienta muy poderosa por los avances que ha permitido desarrollar en múltiples áreas de investigación como la ingeniería, la medicina, la industria y la milicia[2]. Mediante el *DSP* se comenzaron a desarrollar métodos que permiten la identificación de fallas en motores de inducción en los que se analiza el espacio espectral de señales de corriente y vibración, ya sean por separado o juntas, por medio de la transformada de Fourier para detectar fallas en balero [3], la presencia de asimetría [4] y la excentricidad en el entrehierro [5].

Entre las matemáticas que se emplean para el procesamiento digital de señales se encuentran los cuaterniones, los cuales son una representación extendida de los números reales muy similar a los números complejos denominados números hipercomplejos. Este tipo de representaciones cuenta con cuatro elementos y se encuentran relacionados con un espacio real de cuatro dimensiones formando un espacio vectorial euclídeo. Este espacio permite una representación no singular, reducida y rápida para la obtención de rotaciones con soluciones matemáticas manejables [6]. Este método ha sido aplicado en el análisis de señales por cuaterniones (*Quaternion Signal Analysis*, QSA), con el fin de detectar pensamientos de movimiento en señales electroencefalográficas. Se demuestra que el *QSA* es una técnica buena en la clasificación de elementos [7].

Por otro lado, los sistemas de lógica programable como el arreglo de compuertas programables (*Field-Programmable Gate Array*, FPGA), tienen la capacidad de implementar cualquier función Boleana para realizar operaciones matemáticas, lo que permite realizar procesamientos de señales. El *FPGA* presenta alta densidad de compuertas, alto rendimiento y bajo consumo de potencia. Además, su estructura y la forma en que se pueden implementar funciones, permiten un cálculo paralelo y con esto mejorar los tiempos de procesamiento [8].

1.2. Objetivos

1.2.1. Objetivo General

Desarrollar un sistema basado en el procesamiento de señales con cuaterniones capaz de clasificar correctamente tres estados de un motor de inducción y con la capacidad de ser implementado en *FPGA*.

1.2.2. Objetivos Particulares

- Analizar, diseñar e implementar en software el algoritmo *QSA* para detectar el estado en los motores de inducción.
- Comparar el desempeño del algoritmo mediante la exactitud, el recall, la especificidad, la cantidad de muestras y el tiempo de muestreo.
- Comparar el desempeño del algoritmo con distintos métodos presentados en el estado del arte para la detección de estados de motores, lo cual permitió la publicación en revista indizada JCR.
- Diseñar e implementar la arquitectura *QSA* en *FPGA*.
- Validar la arquitectura del algoritmo mediante la evaluación del consumo de recursos, el tiempo de procesamiento y exactitud.
- Comparativa de la arquitectura con distintos métodos presentados en el estado del arte, lo cual permitió la publicación en revista indizada JCR.

1.3. Hipótesis

El procesamiento estadístico de señales con cuaterniones permite la clasificación correcta de múltiples fallas en motores de inducción.

1.4. Justificación

La detección temprana de fallas en motores de inducción de manera no invasiva permite la programación y reestructuración en líneas de producción industriales, por lo que se benefician las empresas al tener una menor pérdida económica en las reparaciones. El *QSA* permite la implementación de una metodología para la detección temprana de fallas en motores de inducción en espacio temporal, presentando gran exactitud con una menor cantidad de muestras en comparación a los trabajos mostrados en el estado del arte. Además, la implementación del método en una tarjeta *FPGA* permite la creación de un sistema portable, de alta velocidad y con un bajo consumo de recursos por el método de árbol de decisión.

1.5. Contribuciones

- Obtención del método *QSA* en software capaz de detectar los estados de motores mediante las señales adquiridas de la corriente de alimentación y los ejes de un acelerómetro.
- Implementación del método *QSA* en un sistema hardware para la detección de estados de motores.
- Publicación de los resultados del desempeño del algoritmo en software en revista IEEE Transactions on Industrial Electronics, Editorial IEEE, ISSN: 0278-0046 (IF 7.503), "Quaternion Signal Analysis Algorithm for Induction Motor Fault Detection", vol. 66, no. 11, pp. 8843-8850, Nov. 2019. doi: 10.1109/TIE.2019.2891468
- Publicación del diseño en hardware y resultados del desempeño del procesamiento en revista Measurement, Editorial Elsevier, ISSN: 0263-2241, (IF 2.791), "Motor fault detection using Quaternion Signal Analysis on FPGA", vol. 138, pp. 416-424, 2019. doi: 10.1016/j.measurement.2019.01.088.

1.6. Organización del Documento

El documento se encuentra dividido en cinco capítulos. El primer capítulo es dedicado a la introducción, en este se realiza una revisión de los antecedentes, son presentados los objetivos del trabajo de tesis, así como su hipótesis y justificación. Al final son presentadas cada una de las contribuciones del trabajo.

En el segundo capítulo se desarrolla la revisión de literatura, el cual contiene el estado del arte y la metodología de la tesis. En el estado del arte se presentan los trabajos previos que han desarrollado otros autores y que serán comparados con los resultados obtenidos en este trabajo. Por otro lado, el marco teórico presenta los conceptos y definiciones utilizadas para el desarrollo de los algoritmos implementados en el trabajo de tesis.

En el tercer capítulo se presenta la metodología seguida para el desarrollo en software del *QSA* y posteriormente los diagramas a bloques y algoritmos de cada una de las partes que conforman su implementación en hardware.

La configuración experimental, las pruebas y resultados del desarrollo en software y hardware son presentados en el capítulo cuatro de manera gráfica y numérica. En este capítulo son presentadas tablas de comparación de exactitud entre los estados evaluados del motor, tablas de comparación con trabajos relacionados y tablas de los recursos obtenidos.

En el quinto capítulo se presentan las conclusiones y el trabajo a futuro del sistema desarrollado, así como las áreas de oportunidad.

Revisión de Literatura

2.1. Estado del Arte

Los motores de inducción son importantes en los procesos industriales. La detección temprana de fallas en motores es esencial para asegurar el tiempo de producción y planear el mantenimiento en la línea de manufactura que lo requiera, lo cual implica una pérdida menor en los costos. Las fallas en los motores de inducción pueden ser generadas por fuerzas electrodinámicas, aislamiento de la bobina, grandes tensiones, envejecimiento térmico y vibraciones mecánicas de fuentes internas o externas. Las fallas mecánicas más comunes de los motores de inducción son fallas en el estator, de baleros, de barras y de rotor [1].

La importancia de la detección temprana de fallas en los motores de inducción ha motivado el desarrollo de algoritmos [9]. Las fallas en baleros y rotor son los tipos más comunes de análisis en fallas de motores. Un ejemplo es el trabajo presentado en [10], en el cual son detectadas las fallas en el devanado del estator mediante un algoritmo que analiza la impedancia del motor. El autor en el artículo mencionado, calcula la impedancia mediante la teoría de función de embobinado (*winding function theory*) y posteriormente compara sus resultados con una base de datos para detectar el tipo de falla.

El desarrollo de métodos que utilizan el procesamiento de señales es una herramienta muy aplicada para la detección de fallas en los motores. Entre los métodos se encuentran los basados en el análisis estadísticos, los cuales proporcionan propiedades numéricas de las señales que son usadas para revisar las condiciones de un sistema. Mediante esto, el trabajo presentado en [11], el autor desarrolla un clasificador de frecuencias moduladas usando promedios de la relación señal ruido (*Signal-to-Noise Ratio*, SNR). Este método presenta una gran efectividad. Existen trabajos que se centran en la predicción y clasificación de vibraciones

en la transmisión por medio del calculo de la media y la covarianza de señales de torque. El porcentaje en la clasificación alcanza el 100 % de exactitud [12]. El método estadístico de mínimos cuadrados se usa para detectar la falla en los baleros por medio del análisis de las señales de vibración [13]. De forma similar, existen algoritmos basados en curtosis espectral que se utilizan para encontrar fallas en elementos del balero por medio de la corriente del estator o señales de vibración para analizar las frecuencias características [14]- [15].

Entre los métodos desarrollados para la identificación de fallas también se encuentran los métodos basados en la transformación de espacio o métodos espectrales como el presentado en [16], donde algunas transformaciones de emisión acústica en el tiempo y frecuencia se utilizan para calcular las características estadísticas. De esta manera, cuatro estados de baleros se clasifican a través de máquinas de vectores de soporte multiclase (*Multiclass Support Vector Machines*, MSVM). Este algoritmo se implementa en el *FPGA* Xilinx Virtex-7 con 99.36 % de precisión, usando 90.1 % de *RAM*, 0.8 % de bloque *LUT* y 0.07 % de bloque lógico configurable. La frecuencia de muestreo de este método es de 100 kHz, lo que permite el procesamiento en línea.

De los métodos más comunes usados en el análisis de motores se encuentran la transformada rápida de Fourier (*Fast Fourier Transform*, FFT) y la transformada wavelet (*Wavelet Transform*, WT). Tales son los casos de los trabajos presentados en [17] y [18], los cuales utilizan la *FFT* para calcular el espectro de frecuencia de las corrientes del estator. Las magnitudes relativas de la señal son usadas para calcular el número de barras rotas en el rotor. De manera similar, en [19] los autores aplican la transformada de Fourier a señales acústicas y de corriente con el fin de estimar fallas de excentricidad. La *FFT* también es utilizada para aplicar el método acortado de selección de frecuencias múltiples en señales acústicas con lo que se diagnostica las fallas en el balero, el estator y el rotor como se presenta en [20]. Continuando con los métodos basados en transformaciones espaciales para la detección de fallas, existen los métodos basado en la transformada discreta wavelet (*Discrete Wavelet Transform*, DWT). El tabajo presentado en [21] utiliza las señales de vibración para detectar el motor saludable y una o dos barras rotas. Así mismo, los autores en [22] proponen este método como una solución viable para la detección de fallas en la jaula exterior de los motores de doble jaula por medio del análisis del espectro de la corriente del estator. Este tipo de métodos de transformación a frecuencia implican costos altos de operación y generalmente

producen un retraso en el diagnóstico, poca resolución, pérdidas de espectro, ineficiencia para proporcionar una buena relación tiempo-frecuencia, etc. [23]. Para contrarrestar estas desventajas de las transformaciones de espacio, se implementan algoritmos como zoom *FFT*, *Espirit* y *MUSIC* para aumentar la resolución. Sin embargo, las pérdidas de espectro son un reto debido a los costos de computación altos y las técnicas muy sensibles a los parámetros usados, dando como resultado una mala identificación de fallas [24]. Por otro lado, los métodos de estimación también son utilizados para la detección de fallas, tal como lo presentan los autores en [25], dónde plantean el uso de un estimador de frecuencia usando la corriente del estator, un estimador de amplitud y un módulo de decisión de falla para la detección de barra rota en el rotor.

El aprendizaje de máquina también es utilizado en la identificación de fallas, como se plantea en [26], en donde se usan los árboles de decisión como método de detección de fallas en los baleros y engranes. A las señales de vibración y corriente se les aplica la *FFT* y la densidad espectral de potencia para ser clasificadas. Este trabajo es desarrollado en una PC y muestra una tasa de diagnósticos correctos de 98.8%. Otro método de aprendizaje utilizado es el clasificador de vecinos cercanos (*nearest neighbor classifier*), como se muestra en [27], en el cual se desarrolla un diagnóstico acústico para detectar cuatro estados de motores. Las redes neuronales artificiales (*Artificial Neural Network*, ANN) aplicadas a características estadísticas son utilizadas en la identificación de fallas como los defectos en baleros, 1/2 barra rotas del rotor, 1 barra rota del rotor, desequilibrio y desalineación en el eje, tal como se presenta en [28]. En este trabajo se descomponen las señales de vibración para calcular quince características estadísticas con el fin de optimizar, seleccionar y extraer las características significativas de las fallas. Este método presenta una efectividad del 92.59%. Otro trabajo que aplica el mismo procedimiento de diagnóstico de falla en baleros es el presentado en [29]. Los métodos de ANN e inteligencia artificial se encuentran limitados por el tiempo de entrenamiento requerido por la red y su limitada confiabilidad en la identificación de datos no entrenados [30].

Los cuaterniones han incrementado su popularidad en el procesamiento de señales debido a la reducción en el número de parámetros y a que ofrecen soluciones matemáticas manejables con pocas restricciones [27]. Los cuaterniones son aplicados en procesamiento de señales digitales para aplicaciones en imágenes [31, 32], cifrado [33] o transformación

espacial [34]–[35]. En el mismo sentido, los conceptos de algoritmos ya existentes se han expandido al dominio de cuaterniones, tal es el caso del filtrado distribuido de Kalman en cuaterniones (*distributed quaternion Kalman filtering*) usado en aplicaciones de estimación como el seguimiento de objetivos [21], o en la estimación adaptativa modelo múltiple para señales en $3D$ y $4D$ [36]. Así mismo, los cuaterniones se han aplicado en el algoritmo de aprendizaje basado en el nuevo cálculo generalizado de HR (*Generalized HR*, GHR) [26], como el cuaternión de mínima curtosis (*Quaternion Least Mean Kurtosis*, QLMK) [19], el filtrado adaptivo no lineal [37] y las redes neuronales recurrentes [20].

El método *QSA* obtiene un modelo de comportamiento mediante un análisis de tres o cuatro señales simultáneamente en el dominio de tiempo. Este método considera cada muestra como un punto, el cual modela trayectorias en tiempo, proporcionando niveles altos de precisión haciendo uso de una cantidad pequeña de muestras. Así mismo, la aplicación de cuaterniones proporciona una buena compensación entre la complejidad computacional y la liberación del bloqueo de cardán, que consiste en evitar la pérdida de un grado de libertad cuando dos de los tres ejes se colocan en paralelo (*Gimbal lock*). Como se muestra en [7], el *QSA* se utiliza para el reconocimiento de patrones en el dominio del tiempo de las señales electroencefalográficas (*Electroencephalographic*, EEG). Estas señales representan la actividad cerebral y que es relacionada con las imágenes motoras. Cuatro de las señales *EEG* son utilizadas para formar una señal de cuaterniones y con los que se modela el comportamiento temporal de las señales mediante sus rotaciones. Posteriormente, se calculan las características estadísticas de los cuaterniones rotados y los resultados son evaluados por medio de árboles de decisión (*decision trees*), máquinas de vectores de soporte (*support vector machine*, SVM) y técnicas de k -vecinos más cercanos (*k-nearest neighbor techniques*, kNN) que alcanzan una exactitud de 84.75 %, 77.35 % y 83.62 %, respectivamente.

En el presente trabajo, un nuevo algoritmo es propuesto para clasificar tres estados de un motor de inducción utilizando por lo menos dos muestras. En el método *QSA* el cuaternión es constituido por la señal de corriente y las tres señales producidas por los ejes de un acelerómetro, el cuaternión es rotado y analizado por medio de métodos estadísticos. Diversas medidas estadísticas fueron evaluadas, dando los mejores resultados la media, el cluster shape y el custer prominence, con los cuales se obtiene un vector de características. Este vector se aplica al árbol de decisión para obtener la condición del motor de inducción. Las condiciones

son etiquetadas en este trabajo como *HLT* (sano), *BAL* (falla mecánica de desbalance) y *BRN* (falla en balero). Se cambia la cantidad de muestras por ventana utilizada para calcular las estadísticas y los resultados son comparados para encontrar el mejor sistema. El método *QSA* es presentado por la facilidad de obtener valores estadísticos en el espacio temporal, los cuales permiten una alta exactitud haciendo uso de un clasificador con estructura simple como el árbol de decisión. Debido a estas características, la técnica propuesta es implementada en *FPGA*, lo que le agrega alta velocidad y portabilidad para su futura implementación en la líneas de producción.

2.2. Marco Teórico

2.2.1. Cuaternión

Hamilton se vio motivado a buscar un equivalente en $3D$ de los número complejos, analizando la representación de un número complejo representado por una pareja de valores y la cual es capaz de rotar puntos en el plano. Entonces un número complejo de tres valores podría girar puntos en el espacio. Finalmente, el número complejo de tres variables tuvo que ser sustituido por uno de cuatro variables, creando el cuaternión. El cuaternión se considera una extensión no conmutativa de números complejos con cuatro componentes (i , j y k), el cual permite un tratamiento unificado de procesos de cuatro dimensiones. La estructura del cuaternión se define en [6] como

$$q = q_0 + q_1i + q_2j + q_3k \quad (2.1)$$

q_0 , q_1 , q_2 y q_3 corresponden a la magnitud de cada una de las componentes de cuaternión, el cual tiene tres operadores imaginarios

$$i^2 = j^2 = k^2 = i \cdot j \cdot k = -1. \quad (2.2)$$

Las componentes del cuaternión siguen las reglas de productos

$$i \cdot j = -j \cdot i = k \quad (2.3)$$

$$j \cdot k = -k \cdot j = i \quad (2.4)$$

$$k \cdot i = -i \cdot k = j. \quad (2.5)$$

Dichas propiedades permiten operaciones como las multiplicaciones descritas por [6]

$$\begin{aligned} q \otimes p &= (q_0 p_0 - q_1 p_1 - q_2 p_2 - q_3 p_3) \\ &+ (q_0 p_1 + q_1 p_0 + q_2 p_3 - q_3 p_2) i \\ &+ (q_0 p_2 - q_1 p_3 + q_2 p_0 - q_3 p_1) j \\ &+ (q_0 p_3 + q_1 p_2 - q_2 p_1 + q_3 p_0) k. \end{aligned} \quad (2.6)$$

Por otro lado, los coeficientes de un cuaternión pueden ser representados de forma vectorial como

$$q = [q_0, q_1, q_2, q_3]^T. \quad (2.7)$$

Los cuaterniones rotados son elementos que representan las rotaciones en tres dimensiones y se relacionan a la representación angular de ejes de rotación. Estos siguen el teorema de rotación de Euler, por lo que cualquier rotación se puede determinar mediante un vector unitario y un ángulo θ . Los componentes de los ejes $(\theta, \hat{x}, \hat{y}, \hat{z})$ pueden ser convertido a un cuaternión rotado q de la forma

$$q_0 = \cos\left(\frac{\theta}{2}\right) \quad (2.8)$$

$$q_1 = \hat{x} \sin\left(\frac{\theta}{2}\right) \quad (2.9)$$

$$q_2 = \hat{y} \sin\left(\frac{\theta}{2}\right) \quad (2.10)$$

$$q_3 = \hat{z} \sin\left(\frac{\theta}{2}\right) \quad (2.11)$$

donde θ es el ángulo de rotación y \hat{x} , \hat{y} y \hat{z} son los vectores unitarios que representan los ejes de rotación como se puede observar en la Fig. 2.1 [6].

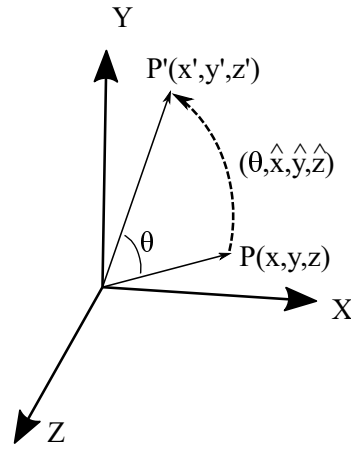


Fig. 2.1: Rotación de cuaterniones.

El vector v es un vector de $3D$ $v = v_1i + v_2j + v_3k$ y se puede expresar como

$$v = [v_1, v_2, v_3]^T. \quad (2.12)$$

Entonces la rotación de un vector en $3D$ v es

$$v' = q \otimes v \otimes q^{-1} \quad (2.13)$$

donde v' es el vector rotado, q es el cuaternión y q^{-1} es el inverso de este cuaternión, el cual es expresado mediante

$$q^{-1} = \frac{\bar{q}}{|q|^2} \quad (2.14)$$

del cual $\bar{q} = q_0 - q_1i - q_2j - q_3k$ [6].

La ecuación 2.13 puede adaptarse a la variable $q(t)$ que es el cuaternión de la muestra presente. Debe entenderse que t es la variable de tiempo discreta. El cuaternión desplazado será expresado $q(t + \Delta t)$, mientras que $q_{rot}(t)$ es el vector rotado y el cual es definido como

$$q(t + \Delta t) = q_{rot}(t) \otimes q(t) \otimes q_{rot}^{-1}(t). \quad (2.15)$$

De la misma forma, la rotación se puede expresar

$$q(t + \Delta t) = R(t) \times q(t). \quad (2.16)$$

En [38] se muestra la matriz de rotación R resultante de $q_{rot}(t)$. Si esta sustitución es representada como $R(t)$, la estructura estará dada por

$$R(t) = \begin{bmatrix} 1 - 2r_2^2 - 2r_3^2 & 2(r_1r_2 + r_0r_3) & 2(r_1r_3 - r_0r_2) \\ 2(r_1r_2 - r_0r_3) & 1 - 2r_1^2 - 2r_3^2 & 2(r_2r_3 + r_0r_1) \\ 2(r_1r_3 + r_0r_2) & 2(r_2r_3 - r_0r_1) & 1 - 2r_1^2 - 2r_2^2 \end{bmatrix} \quad (2.17)$$

en el cual $r = [r_0, r_1, r_2, r_3]^T$ y

$$\begin{aligned} r_0 &= q_{0_rot}(t) \\ r_1 &= q_{1_rot}(t) \\ r_2 &= q_{2_rot}(t) \\ r_3 &= q_{3_rot}(t). \end{aligned} \quad (2.18)$$

Un cuaternión puede ser usado para modelar el comportamiento de una máquina mediante una señal de vibración en $3D$. Tomando en cuenta que se tiene un conjunto de señales discretas $I(t)$, $x(t)$, $y(t)$ y $z(t)$ con m muestras, donde $I(t)$ es la señal corriente y $x(t)$, $y(t)$ y $z(t)$ son las señales de los ejes del acelerómetro, entonces estas cuatro señales pueden representarse como el cuaternión

$$q(t) = I(t) + x(t)i + y(t)j + z(t)k \quad (2.19)$$

o en forma vectorial

$$q(t) = [I(t), x(t), y(t), z(t)]^T. \quad (2.20)$$

Analizando la señal de la última ecuación en tiempo y tomando en consideración m muestras, con un tiempo de desfase Δt en cada muestra, la ecuación de rotación (2.16) es aplicada de forma recurrente para obtener la estimación del modelo $R(t)$, con el cual se describe el comportamiento en la señal temporal como

$$\begin{bmatrix} q(t + \Delta t) \\ q(t + 2\Delta t) \\ q(t + 3\Delta t) \\ \vdots \\ q(t + m\Delta t) \end{bmatrix} = \begin{bmatrix} R(t) \\ R(t + \Delta t) \\ R(t + 2\Delta t) \\ \vdots \\ R(t + (m-1)\Delta t) \end{bmatrix} \begin{bmatrix} q(t) \\ q(t + \Delta t) \\ q(t + 2\Delta t) \\ \vdots \\ q(t + (m-1)\Delta t) \end{bmatrix} \quad (2.21)$$

Este modelo es expresado como un cuaternión por las ecuaciones (2.17) y (2.18), con lo que se obtienen $m - \Delta t$ modelos, los que se expresan como

$$Q_R = \begin{bmatrix} q_{rot}(t) \\ q_{rot}(t + \Delta t) \\ q_{rot}(t + 2\Delta t) \\ \vdots \\ q_{rot}(t + (m - 1)\Delta t) \end{bmatrix} = \begin{bmatrix} R(t) \\ R(t + \Delta t) \\ R(t + 2\Delta t) \\ \vdots \\ R(t + (m - 1)\Delta t) \end{bmatrix}. \quad (2.22)$$

2.2.2. Estadísticas

Las características que describen el comportamiento del modelo se obtienen por el análisis estadístico de los N módulos de Q_R , los cuales son descritos como $|q_{rot}(t + k\Delta t)|$. Por ejemplo, la esperanza matemática de estos módulos resultantes se calcula como se describe en [39]

$$\mu(t + k\Delta t) = \frac{1}{N} \sum_l |q_{rot}(t + (k + l)\Delta t)|. \quad (2.23)$$

Mediante este valor se obtiene características de segundo orden como *cluster shades* y *cluster prominence*. El *cluster shades* es la medida de falta de simetría de una matriz. De acuerdo con [39], el *cluster shades* puede expresarse en forma vectorial

$$CS(t + k\Delta t) = \frac{1}{N} \sum_l (|q_{rot}(t + (k + l)\Delta t)| - \mu(t + k\Delta t))^3. \quad (2.24)$$

Igualmente, el *cluster prominence* indica la falta de simetría y puede describirse como

$$CP(t + k\Delta t) = \frac{1}{N} \sum_l (|q_{rot}(t + (k + l)\Delta t)| - \mu(t + k\Delta t))^4. \quad (2.25)$$

Mediante estos valores estadísticos, un vector de características puede expresarse como

$$w(t + k\Delta t) = [\mu(t + k\Delta t), CS(t + k\Delta t), CP(t + k\Delta t)]^T. \quad (2.26)$$

2.2.3. Árbol de Clasificación

Un árbol de clasificación es una máquina de aprendizaje que puede dividir el espacio de características en dos partes y cada parte resultante se divide en dos nuevamente y así sucesivamente para valores de $w(t + k\Delta t)$ como se muestra en la Fig. 2.2

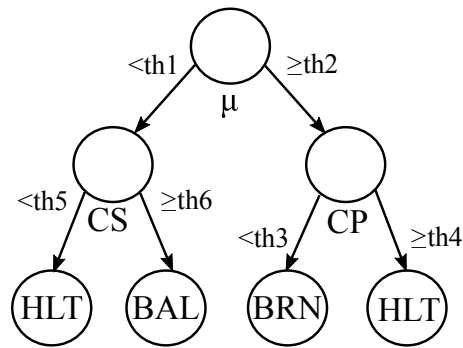


Fig. 2.2: Estructura del árbol de clasificación.

El nodo superior es conocido como “nodo raíz”, el cual es el nodo de clasificación más efectivo. Los nodos que se encuentran al final son llamados “hojas”, las cuales pueden ser eliminadas para garantizar mejores resultados de clasificación de acuerdo a la cantidad de muestras que presentan errores. Este tipo de árboles de clasificación son conocido como “árbol de decisión podado” [40].

El proceso del árbol de clasificación se divide en dos fases: entrenamiento y prueba. Como se muestra en la Fig. 2.3, el entrenamiento arroja los valores de la estructura del árbol de decisión por medio de los coeficientes de cada nodo. Estos coeficientes se obtienen relacionando el vector de características $w_e(t + k\Delta t)$ con la etiqueta de propiedades correspondiente $L_e(t + k\Delta t)$. Las etiquetas se componen de los estados del motor de la forma

$$L(t + k\Delta t) = \{HLT, BAL, BRN\}. \quad (2.27)$$

La fase de prueba consiste en estructurar el árbol de decisión con los coeficientes calculados en la primer etapa para clasificar el conjunto de vectores de características $w(t + k\Delta t)$. Cada vector obtiene la respuesta $c(t + k\Delta t)$ de la forma

$$W = \{(w(t + \Delta t), c(t + \Delta t)), (w(t + 2\Delta t), c(t + 2\Delta t)), \dots, (w(t + k\Delta t), c(t + k\Delta t))\}. \quad (2.28)$$

Son evaluados los vectores de características de las tres condiciones del motor de inducción para obtener un conjunto de etiquetas $c(t + k\Delta t)$, que serán comparadas con el estado real.

En el trabajo presentado, el programa *MATLAB* se utiliza para calcular los coeficientes que conforman los valores de los nodos mediante el entrenamiento de un archivo elegido al azar. Posteriormente, los coeficientes son utilizados para clasificar los archivos de la base de datos que se presenta en el capítulo 4.

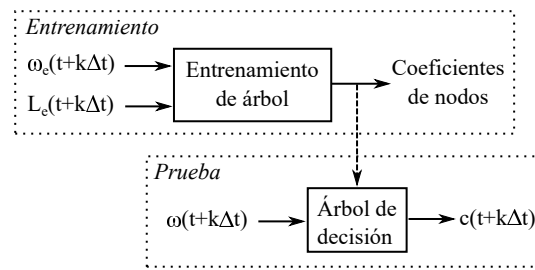


Fig. 2.3: Proceso de árbol de clasificación.

2.2.4. Lógica Programable

Los sistemas *FPGA* tienen densidades altas de compuertas, lo que permite una gran cantidad de entradas y salidas que el usuario puede definir. Así mismo, el *FPGA* cuenta con un alto rendimiento en el consumo de potencia y una alta optimización de recursos en los bloques digitales de los dispositivos. El *FPGA* tiene una arquitectura estilo Manhattan, la cual contiene estructuras lógicas simples que se encuentran conectadas entre sí. Estas celdas se dividen en celdas lógicas, celdas de interconexión y celdas de entrada/salida. La celda lógica (*CLB*) se compone de bloques lógicos de compuertas programables son los que ocupan una mayor cantidad de espacio. El *CLB* consta de cuatro capas en los que se desarrolla la lógica combinacional y registros de almacenamiento con elementos como *look-up tables (LUT)*, multiplexores, acarrees lógicos, compuertas *AND* y elementos secuenciales. Las salidas de los bloques *CLB* se encuentran conectadas a segmentos cableados por medio de puntos de interconexión programables (*PIP*).

Las celdas de entrada/salida (*E/S*) son los bloques encargados de realizar la conectividad de la tarjeta *FPGA* con el exterior. Estas celdas se pueden programar independientemente para su función de entrada, salida, con control tri-estado o un pin bidireccional. Las celdas tienen la interconectividad de la *FPGA* restringida a celdas adyacentes, lo que permite un menor consumo de potencia [41].

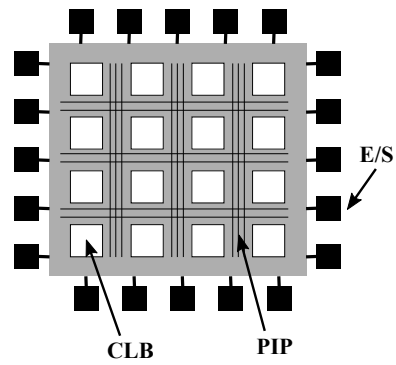


Fig. 2.4: Conformación a bloques de un FPGA

Los métodos que se han presentado serán utilizados para desarrollar los algoritmos que se muestran en el capítulo siguiente. Los cuales son explicados paso a paso, así como su estructura en la implementación en *FPGA*.

En este capítulo se presentan las ecuaciones y diagramas a bloques que constituyen el desarrollo del método QSA, con el cual se obtiene un modelado de trayectorias para el reconocimiento de patrones en el dominio de tiempo de tres o cuatro señales simultáneamente. Este método permite la clasificación de los principales estados estudiados en los motores de inducción como son el estado sano, falla mecánica de desbalance y falla en balero. A los motores de inducción se les extrae una señal de corriente y tres de los ejes de un acelerómetro, con lo que cumple la condición de ser utilizadas cuatro señales para la conformación del cuaternión que será aplicado al análisis. En este capítulo también se presentan los algoritmos implementados en el *FPGA* de cada uno de los bloques que conforman el sistema.

3.1. Algoritmo

El algoritmo presentado en este trabajo se muestra en la Fig. 3.1. El método *QSA* consta de cinco pasos: conformación del cuaternión, normalización de cuaternión, rotación, norma de modelo rotacional y extracción de características estadísticas, tal como se presenta en [42].

Inicialmente, el cuaternión $q(t)$ está conformado por los elementos q_0 , q_1 , q_2 y q_3 , los cuales corresponden con las señales $I(t)$, $x(t)$, $y(t)$ y $z(t)$. El cuaternión se estructura como se muestra en la ecuación 2.20. Posteriormente, el cuaternión es normalizado para permitir la correcta separación de las propiedades estadísticas y mejora la escalabilidad del sistema. El cuaternión $q_n(t)$ se calcula usando la norma Euclidiana

$$q_n = \frac{1}{\sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}} [q_0, q_1, q_2, q_3]^T. \quad (3.1)$$

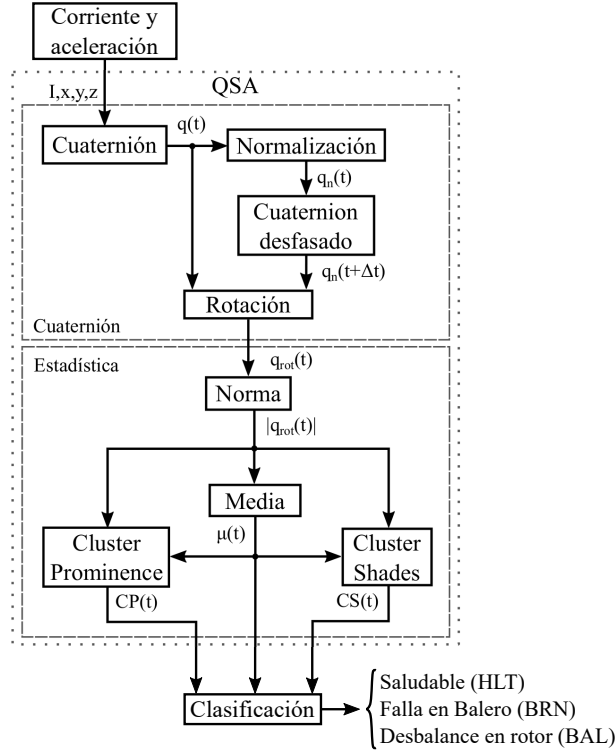


Fig. 3.1: Algoritmo del análisis de señal cuaternión.

Este cuaternión $q_n(t)$ es retrasado una cantidad de tiempo Δt con el fin de contar con un cuaternión normalizado desfasado. Posteriormente, se utiliza una muestra del cuaternión actual $q(t)$ y del desfasado $q_n(t + \Delta t)$ para obtener un modelo tridimensional de orientación y rotación $q_{rot}(t)$ por medio de la ecuación

$$q_{rot} = \begin{bmatrix} 1 - 2q_{n2}^2 - 2q_{n3}^2 & 2(q_{n1}q_{n2} + q_{n0}q_{n3}) & 2(q_{n1}q_{n3} - q_{n0}q_{n2}) \\ 2(q_{n1}q_{n2} - q_{n0}q_{n3}) & 1 - 2q_{n1}^2 - 2q_{n3}^2 & 2(q_{n2}q_{n3} + q_{n0}q_{n1}) \\ 2(q_{n1}q_{n3} + q_{n0}q_{n2}) & 2(q_{n2}q_{n3} - q_{n0}q_{n1}) & 1 - 2q_{n1}^2 - 2q_{n2}^2 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix}. \quad (3.2)$$

Esta rotación realizada mediante la ecuación 3.2, describe el comportamiento en la señal en el tiempo t para obtener la misma señal retrasada en $t - 1$. Enseguida, la norma es aplicada al modelo tridimensional rotado para obtener su magnitud $|q_{rot}(t)|$ como

$$|q_{rot}| = \sqrt{q_{rot1}^2 + q_{rot2}^2 + q_{rot3}^2}. \quad (3.3)$$

A cada muestra de $q(t)$ se le aplica este método para crear un vector con las magnitudes de la rotación de cuaterniones. Los valores estadísticos μ , CP y CS son calculados mediante una ventana de N muestras de este vector de normas por medio de las ecuaciones

$$\begin{aligned}\mu &= \frac{1}{N} \sum x \\ CS &= \frac{1}{N} (\sum x^3 - 3\mu \sum x^2 + 3\mu^2 \sum x - \mu^3 N) \\ CP &= \frac{1}{N} (\sum x^4 - 4\mu \sum x^3 + 6\mu^2 \sum x^2 - 4\mu^3 \sum x + \mu^4 N).\end{aligned}\tag{3.4}$$

Estas ecuaciones son una adecuación con el fin de obtener valores estadísticos en términos de sumatorias [7], donde x es la norma $|q_{rot}(t)|$. Dichas características estadísticas se calculan a partir de los cuaterniones rotados ya que representan la evolución de la señal en el espacio temporal.

Finalmente, un clasificador de árbol de decisión es aplicado a las características estadísticas para detectar si el motor de inducción está en buen estado, tiene falla en baleros o presenta un desbalance mecánico del eje [42].

3.2. Implementación

En esta sección se describe el diseño y la implementación del *QSA* y el clasificador de árbol de decisión en una tarjeta DE2 115 con *FPGA* Cyclone IV EP4CE115F19. En la Fig. 3.2 se muestra el diagrama general del sistema propuesto. Las señales q_0 , q_1 , q_2 y q_3 corresponden a las componentes del cuaternión que son manejadas en formato punto fijo de ocho bits con signo. El bloque *QSA* desarrolla el método para obtener los valores estadísticos μ , CS y CP de las ventanas del cuaternión q . Finalmente, el bloque clasificación arroja la clase *HLT*, *BRN* o *BAL*.

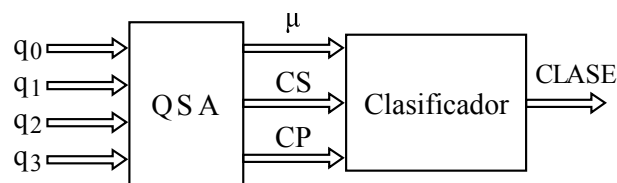


Fig. 3.2: Diagrama general del sistema propuesto.

El diagrama del sistema se muestra en la Fig. 3.3, donde se observa que el bloque *QSA* se conforma por el bloque de normalización del cuaternión (*NORZ*), los registros de almacenamiento (*HREG*), el registro de retardo (*REG*), el bloque que realiza la operación de la rotación de cuaternión (*ROT*), el bloque de norma de tres variables (*NORM3*) y el bloque

de cálculo de estadísticos (*STA*). Cabe señalar que todos estos bloques son sincronizados mediante una máquina de estados, la cual es la encargada de activar y desactivar cada uno de los bloques a lo largo del procedimiento.

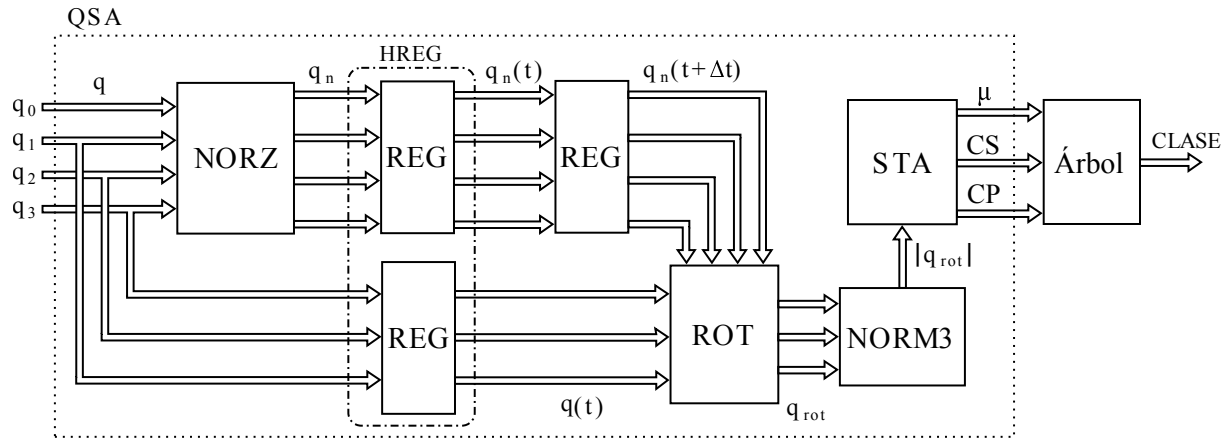


Fig. 3.3: Representación interna del sistema.

3.2.1. Normalización

Una vez que se ha estructurado, el cuaternión q se normaliza en el bloque *NORZ* usando la ecuación mostrada en (2.1). El bloque *NORZ* se conforma por dos bloques internos: el bloque *NORM* y el bloque de división A/B como se muestra en la Fig. 3.4(a). La normalización del cuaternión q se obtiene dividiendo cada componente del cuaternión entre su norma.

En la Fig. 3.4(b) se muestra la estructura interna del bloque que calcula la norma del cuaternión *NORM*. En este bloque, los componentes del cuaternión son multiplicados por sí mismos para obtener el cuadrado de q_i , posteriormente, cada una de las componentes elevadas al cuadrado se suman dando el valor de S . A este último se le aplica la raíz cuadrada obteniéndose la norma $|q|$. En la Fig. 3.4(c) se presenta la estructura de la raíz cuadrada que se ejecuta en el bloque *NORM* y se describe en el Algoritmo 1. Este algoritmo es una versión modificada del algoritmo de raíz cuadrada presentado en [43]. El tamaño del vector lógico S es n , las variables internas D_1 y D_2 son vectores lógicos, a los cuales se les inicializa con el valor decimales de cero y uno decimal respectivamente. El ciclo *for* se itera $n/2$ veces como muestra la línea 4. En cada iteración del ciclo, los bits menos significativos de D_1 son concatenados con los bits más significativos de S para reasignar el vector D_1 . El cuadrado del vector lógico D_2 es comparado con D_1 en la línea 6, si el cuadrado de D_2 es menor que

D_1 , entonces, el bit lógico B se establece en '1'; de lo contrario, es establecido en '0'. El bit B es desplazado a la izquierda con respecto de la salida D_o en la línea 11. Después de esto, los bits menos significativos de D_o y el bit '1' son concatenados y asignados a D_2 . Finalmente, en la línea 13, la variable S es reasignada con el valor del desplazamiento a la izquierda de S antes de que se repita el proceso dentro del ciclo *for*. Cuando finaliza en la línea 15, el último valor de D_o es la raíz cuadrada del valor inicial de S .

Alg. 1 Algoritmo de la raíz cuadrada

1: **Proceso** SQRT ▷ Entrada: S ▷ Salida: Do

Necesita $n \leftarrow \text{tamaño}(S)$

2: $D_1 \leftarrow 0$

3: $D_2 \leftarrow 1$

4: **for** $i \leftarrow 1, n/2$

5: $D_1 \leftarrow D_1[n - 2 : 1] \ \& \ S[n : n - 1]$

6: **if** $D_2^2 \leq D_1$

7: $B \leftarrow '1'$

8: **else**

9: $B \leftarrow '0'$

10: **fin if**

11: $D_o \leftarrow D_o[n - 1 : 1] \ \& \ B$

12: $D_2 \leftarrow D_o[n - 1 : 1] \ \& \ '1'$

13: $S \leftarrow \text{desplaza_izq}(S)$

14: **fin for**

15: **fin Proceso**

Una vez que se ha obtenido la norma de q en el módulo de normalización, la siguiente operación a realizar es la división de cada elemento entre la norma ya obtenida. La estructura interna del bloque de la división se muestra en la Fig. 3.4(d) y el pseudocódigo se presenta en el Algoritmo 2, donde A y B representan las entradas numéricas. A la variable n se le asigna el tamaño del vector lógico A y el vector lógico re se inicializa con el valor decimal 0 en la línea 2. El vector re representa el residuo de la división. En la línea 3, el ciclo *for* se repite n veces. Dentro del ciclo, en la línea 4, el vector lógico D es asignado con la concatenación de los bits menos significativos de re y el bit más significativo de A . Este vector lógico es

comparado con el vector lógico B en la línea 5. Si D es menor, el bit lógico auxiliar r es asignado con valor binario '0' y el vector lógico re con el valor de D . De lo contrario, a r se le asigna el valor lógico '1' y al vector lógico re la resta $D - B$. Después de esta evaluación, los bits más significativos de d_o se concatenan con el bit r y esto es reasignado al vector d_o , que será el vector auxiliar para formar el vector de salida final d en la línea 12. En la línea siguiente, el vector lógico A es desplazado a la izquierda, y este mismo procedimiento es realizado para cada elemento del vector A . Finalmente en la línea 15, se asigna d_o a d , el cual es el cociente de división.

Alg. 2 Algoritmo de la division

 1: **Proceso** DIV(A, B, d)

 ▷ Entradas: A, B

 ▷ Salida: d
Necesita $n \leftarrow \text{tamaño}(A)$

 2: $re \leftarrow 0$

 3: **for** $i \leftarrow 1, n$

 4: $D \leftarrow re[n - 1 : 1] \ \& \ A[n]$

 5: **if** $D < B$

 6: $r \leftarrow '0'$

 7: $re \leftarrow D$

 8: **else**

 9: $r \leftarrow '1'$

 10: $re \leftarrow D - B$

 11: **fin if**

 12: $d_o \leftarrow d_o[n : 2] \ \& \ r$

 13: $A \leftarrow \text{desplaza_izq}(A)$

 14: **fin for**

 15: $d \leftarrow d_o$

 16: **fin Proceso**

Regresando al diagrama a bloques de la Fig. 3.3, una vez completado el bloque de normalización de cuaternión, se ejecuta el bloque *HREG* utilizando los vectores q_n y q . Este bloque permite la sincronización de muestras de ambos cuaterniones al ser implementado el sistema en *FPGA*. Posteriormente, el registro de retardo *REG* es aplicado a la señal $q_n(t)$ para obtener el cuaternión retardado $q_n(t + \Delta t)$.

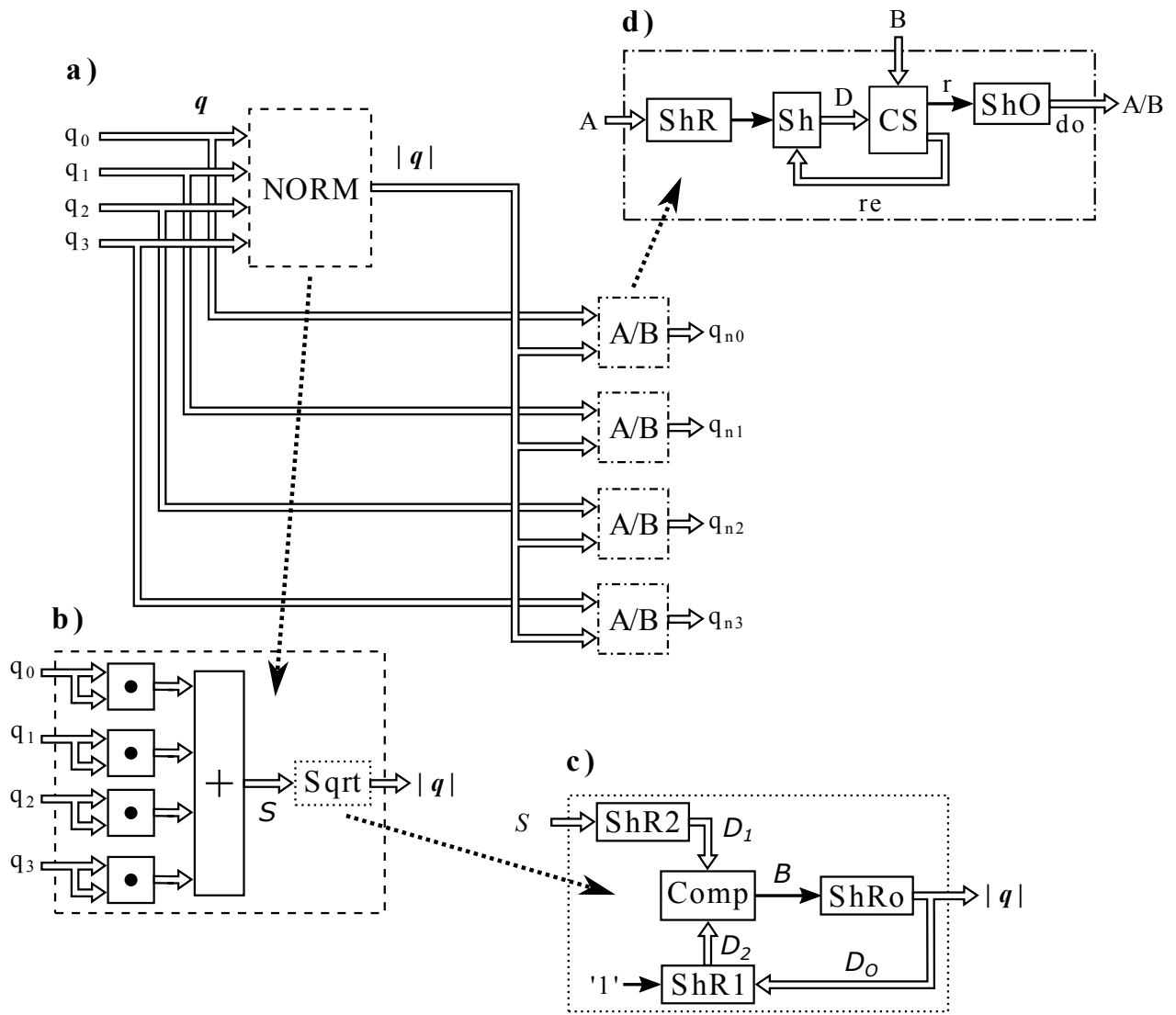


Fig. 3.4: Bloque normalización del cuaternión. a) Estructura general. b) Estructura interna del bloque de norma. c) Estructura interna del bloque raíz cuadrada. d) Estructura del bloque división.

3.2.2. Rotación de Cuaternión

Las señales previamente obtenidas $q_n(t)$ y $q_n(t + \Delta t)$ se emplean para el cálculo de la rotación de cuaternión a través de la ecuación 3.2, como se ilustra mediante el bloque *ROT* en la Fig. 3.3. Con los componentes resultantes de la rotación se calcula la norma en el bloque *NORM3*, el cual utiliza la ecuación 3.3. El diagrama a bloques interno de *NORM3* calcula el cuadrado de cada componente del cuaternion, para posteriormente ser sumados y aplicarle al resultado la raíz cuadrada, tal como se observa en el diagrama a bloques presentado en la Fig. 3.5. Este bloque es similar al bloque *NORM* representado en la Fig. 3.4(b), con la

variación de que este caso usa solamente los tres componentes complejos.

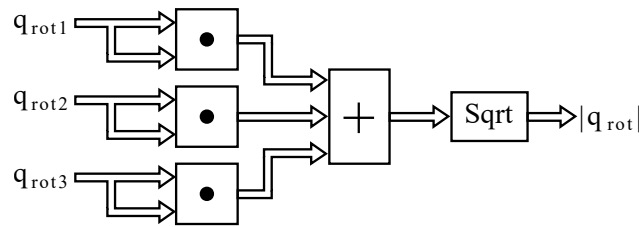


Fig. 3.5: Estructura de la norma de tres componentes.

3.2.3. Extracción de Características Estadísticas

El resultado de la norma del cuaternión de rotación de cada muestra $|q_{rot}|$ es guardado en una ubicación de la memoria de acceso aleatorio (*Random access memory, RAM*) implementada como una memoria interna en el *FPGA*, como se puede observar en la Fig. 3.6. Los datos que son almacenados en cada ubicación de memoria son leídos en la variable x para, posteriormente, ser elevados al cuadrado, cubo y a la cuarta potencia. Estos valores son utilizados para calcular las sumatorias de x , x^2 , x^3 y x^4 . Estas son utilizadas para calcular los valores estadísticos. El valor de la media μ es obtenida al dividir la suma de x entre el número N de muestras de la ventana seleccionada. De forma similar, este valor es elevado al cuadrado, al cubo y a la cuarta potencia. Por lo tanto, las características estadísticas CS y CP son calculadas con la multiplicación, la suma, la resta y la división entre los valores de las sumatorias de x , μ y N , tal como se observa en las ecuaciones 3.4

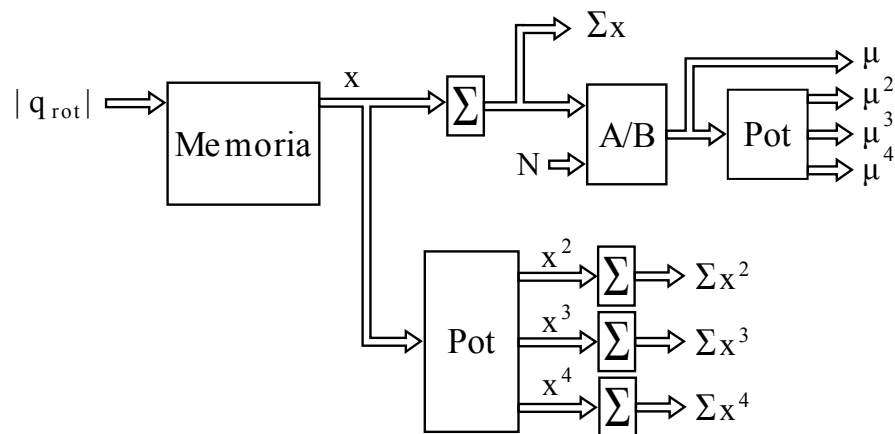


Fig. 3.6: Estructura interna del bloque estadístico.

3.2.4. Clasificación

La segunda etapa del proceso consiste en obtener una clasificación de las características estadísticas del *QSA*. Los valores estadísticos μ , *CS* y *CP* son utilizados como características de entrada de un clasificador de árbol de decisión como se muestra en la Fig. 3.7. El clasificador cuenta con una memoria de solo lectura (*Read only memory, ROM*) implementada en *FPGA*, la cual contiene las características de cada nodo del árbol calculado. Cada dirección de la ROM contiene los siguientes valores: *tipo*, *lim*, *dir1*, *dir2*, *clase* y *fin*. El valor de *dir* corresponde a la dirección del puntero en el cual se leerá la memoria, mientras que la variable *tipo* selecciona la característica estadística que evaluará el nodo actual del árbol y la variable *lim* es el valor límite con el que se realiza la comparación de la característica seleccionada. Si la comparación es verdadera ('1'), el valor *dir1* será asignada a *dir* como la siguiente dirección de memoria de la que se extraerá un valor de la ROM. En caso contrario, la siguiente dirección será el valor *dir2*. La clase final está contenida en un valor de *clase*, mientras que la bandera *fin* indica cuando el proceso de clasificación ha terminado. Los detalles de implementación de estos módulos de arquitectura se muestran enseguida.

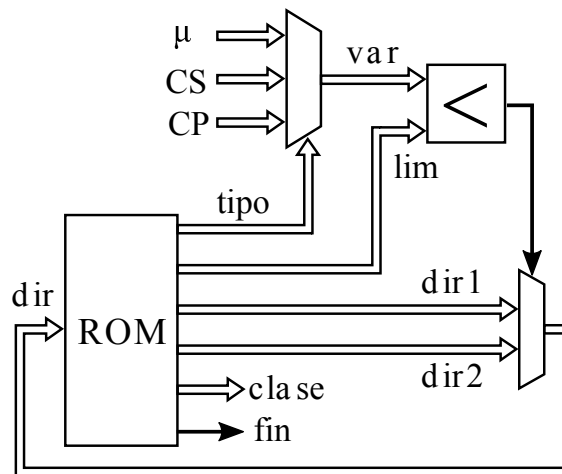


Fig. 3.7: Estructura interna del árbol de decisión.

El Algoritmo 3 presenta el clasificador de árbol de decisión. En la línea 2, la dirección inicial *dir* se establece en cero y en esta dirección de memoria se contienen los valores de configuración del nodo actual en el árbol. Posteriormente, las líneas 3 a 16 evalúan el árbol de decisión hasta que la bandera *fin* se establece en '1'. En las líneas 4 a 9, el valor de *tipo* selecciona la característica estadística correspondiente entre μ , *CS* y *CP* y se le asigna a *var*.

Después de eso, en la línea 11, se compara el valor de la característica con el valor límite lim . Si var es menor que el valor de lim , a dir se le asigna la siguiente dirección indicada por $dir1$. En caso contrario, la siguiente dirección de dir se le asignará el valor de $dir2$. Cuando el indicador fin es activado, el valor $clase$ del nodo correspondiente contiene el resultado de la clasificación. Para su implementación, un árbol es entrenado para cada longitud de ventana seleccionada y los valores se almacenan en el *FPGA*. Posteriormente, los bancos de pruebas de desbalance, falla de balero y motor en buen estado se envían al *FPGA* por la comunicación RS-232 para verificar su asertividad.

Alg. 3 Algoritmo de árbol de decisión

1: Proceso ÁRBOL DE DECISIÓN 2: $dir \leftarrow 0$ 3: while $end = FALSE$ 4: if $tipo = "00"$ 5: $var \leftarrow \mu$ 6: else if $tipo = "01"$ 7: $var \leftarrow cp$ 8: else 9: $var \leftarrow cs$ 10: fin if 11: if $var < lim$ 12: $dir \leftarrow dir1$ 13: else 14: $dir \leftarrow dir2$ 15: fin if 16: fin while 17: $clase(dir)$ 18: fin Proceso	▷ Entradas: μ, cp, cs ▷ Salida: clase
---	---

La metodología presentada se desarrolló en software y hardware. Su eficiencia es mostrada en el capítulo siguiente.

Pruebas y Resultados

Este capítulo es dividido en la configuración experimental y la presentación de resultados. En la primera parte se presentan las partes que componen el sistema de donde se adquirieron las señales de corriente y vibración para comprobar el correcto funcionamiento del *QSA*. Se presentan características de cada componente, principalmente del motor y la configuración de fallas. La segunda sección de este capítulo presentan los resultados de las pruebas de exactitud general, la comparación de exactitud entre clases y la comparación de la exactitud obtenida en cada implementación. En este capítulo también se presenta la comparación entre los resultados del método *QSA* y los trabajos descritos en el estado del arte para comprobar el correcto desarrollo de nuestro método. Finalmente, se discuten los resultados obtenidos entre la implementación el método en *FPGA* y otros trabajos similares.

4.1. Configuración Experimental

Como se mencionó en la sección de estado del arte, diversos algoritmos han sido aplicados para la clasificación de estados del motor de inducción. Los estados evaluados continuamente en estos trabajos son la polea desequilibrada o desbalanceada (*BAL*), falla del motor en el balero (*BRN*) y motor en buen estado (*HLT*). Estos son elegidos para evaluar debido a su aparición constante en maquinaria industrial y por la facilidad de simulación en laboratorio. La configuración experimental es mostrada en la Fig. 4.1. Como se puede observar, se cuenta con un motor motor de inducción al cual se le aplica una carga mecánica mediante un alternador ordinario. Los datos experimentales medidos del motor trifásico (WEG 00136APE48T) consisten de cuatro señales: la señal de corriente del motor de inducción medida con la pinza de corriente modelo i200 de Fluke y las tres señales

producidas por el acelerómetro basado en MEMS (LIS3L02AS4) montado en el chasis del motor. El motor cuenta con 1 hp, una potencia nominal de 0.74 kW, 2 polos, 28 barras, peso de 9 kg, longitud de 239 mm y una anchura de 150 mm. La adquisición de las cuatro señales se realiza mediante un sistema (DAS) cuya frecuencia de muestreo es de 1.5 kHz, el cual proporciona 4096 muestras por prueba en un estado estable del motor *QSA*.

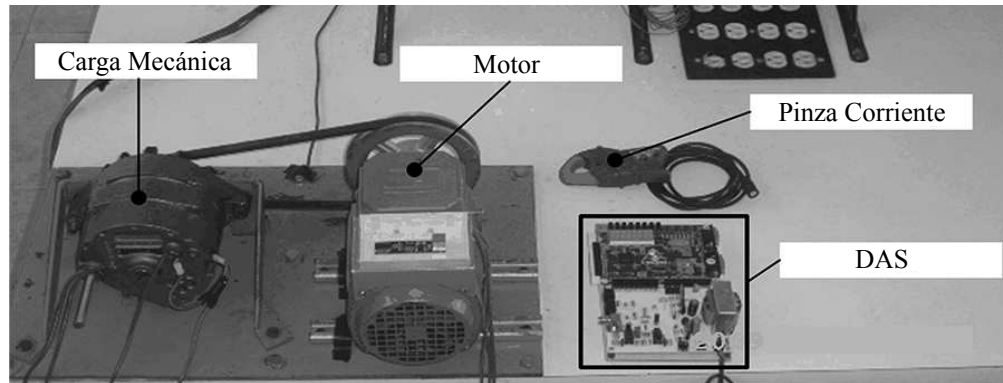


Fig. 4.1: Configuración experimental.

El banco de pruebas se adquiere con el motor trabajando a 3402 RPM y la falla en el balero es provocada mediante un orificio perforado de 7.938 mm de diámetro, mientras que la falla en el desbalance en el eje es causada adicionando masa a la polea. Estas fallas se muestran en la Fig. 4.2.

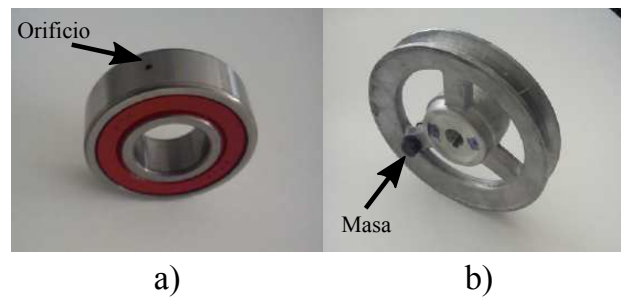


Fig. 4.2: Configuración de fallas. a) Falla de balero. b) Falla de desbalance.

El banco de pruebas contiene cuarenta y cinco archivos en total, de los cuales cinco mediciones son de falla en balero, veinte mediciones son de desbalance mecánico en la barra y veinte mediciones son de motor de inducción sano. Una prueba del banco total se muestra en la Fig. 4.3.

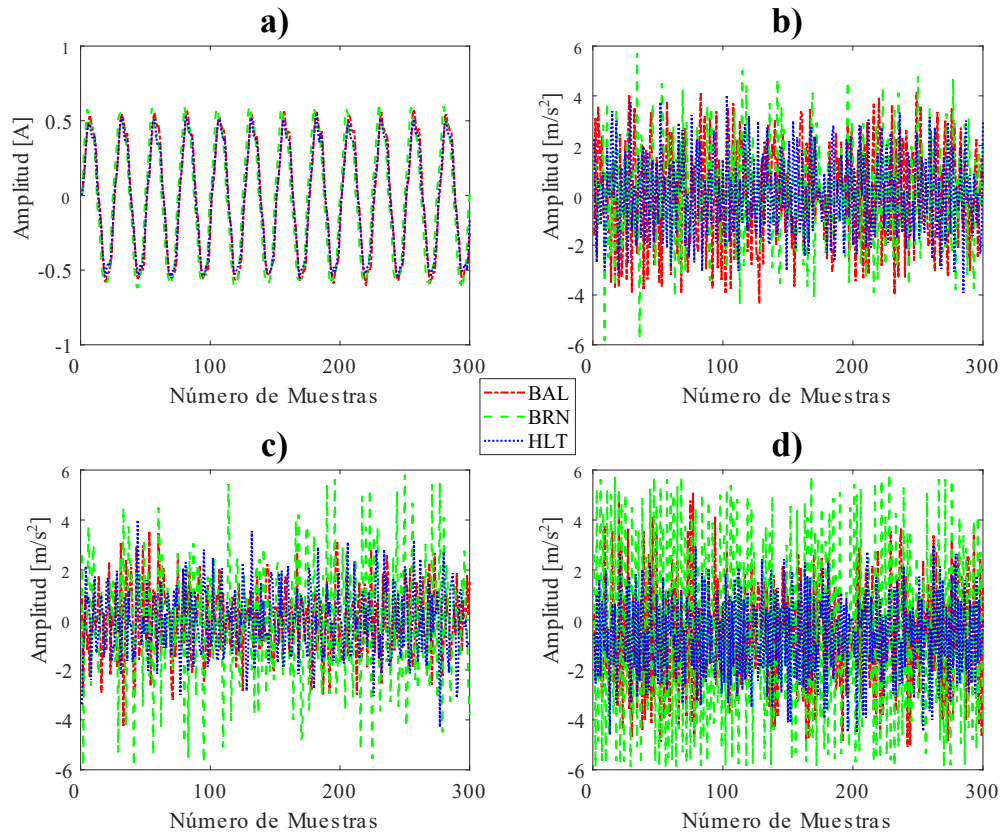


Fig. 4.3: Prueba de señales. a) Señales de corriente. b) Señales de vibración para eje X. c) Señales de vibración para eje Y. d) Señales de vibración para eje Z.

El entrenamiento del árbol de decisión se realiza eligiendo al azar un archivo de cada condición del motor. Las señales del archivo son adaptadas al cuaternión y se le aplica el método *QSA* para obtener μ , CP y CS . Los resultados estadísticos son utilizados para entrenar el clasificador. Los coeficientes resultantes son calculados para cada tamaño de ventana desde las dos a las doscientas muestras, eligiendo un archivo al azar en cada incremento.

La efectividad del método se comprueba aplicando el algoritmo *QSA* y el clasificador correspondiente a la totalidad de archivos que componen el banco de pruebas. Las etiquetas resultantes y las etiquetas reales son comparadas para revisar las clasificaciones correctas y determinar las métricas que validen el método. A pesar de que la cantidad de pruebas no es equitativa, la elección de una prueba para entrenar y las demás para comprobar no genera un problema, ya que los porcentajes de efectividad se toman con respecto al total de las pruebas

de cada condición.

Los resultados en la clasificación del *QSA* son cuantizados mediante una matriz de confusión por cada evaluación de ventana. En la Tabla 4.1 se presenta un ejemplo de una matriz de confusión obtenida utilizando ventanas de 15 muestras.

Tabla 4.1: Muestra de matriz de confusión utilizando ventana de 15 muestras.

Clase Verdadera	Clase Identificada		
	BAL	BRN	HLT
BAL	3510	245	326
BRN	408	3673	0
HLT	367	0	3714

El experimento desde la evaluación con la base de datos total es repetido veinte veces para asegurar su repetibilidad, por lo que se tienen veinte matrices de confusión para cada valor de ventana. Los árboles de decisión son entrenados previamente.

Se hace uso de tres métricas para evaluar el rendimiento del método, las cuales corresponden a exactitud, recall y especificidad, definidas como

$$Exactitud = \frac{TP + TN}{TP + FP + FN + TN} \quad (4.1)$$

$$Recall = \frac{TP}{TP + FN} \quad (4.2)$$

$$Especificidad = \frac{TN}{TN + FP} \quad (4.3)$$

En estas ecuaciones, TP es el número de verdaderos positivos declarados (estado positivo clasificado como positivo), TN es el número de verdaderos estados negativos (estado negativo clasificado como negativo), FP es el número de estados falsos positivos (estado negativo clasificado como positivo) y FN es el número de estados falsos negativos (estado positivo clasificado como negativo).

Las tres métricas tienen el rango de 0 a 1, siendo 1 el mejor valor posible. Las métricas son calculadas para las veinte matrices de confusión de cada ventana. Asimismo las exactitudes,

recall y especificidad generales son obtenidas para un análisis amplio. Para calcular la exactitud general del experimento se obtiene la exactitud de las veinte matrices por separado y se promedian. En el caso del recall y la especificidad, cada valor de las veinte matrices es sumado y posteriormente se calculan las métricas con los valores totales. Estas métricas son promediadas para obtener un valor general de cada una.

El método propuesto en este trabajo se aplicó para el banco de pruebas total con ventanas desde dos hasta doscientas muestras.

4.2. Resultados

Se creó una base de datos con las etiquetas de las clases resultantes de aplicar el *QSA* a todas las pruebas que se tienen en la configuración experimental, incluyendo las seleccionadas para entrenamiento. Cada prueba cuenta con la etiqueta de la falla que se presenta en el motor. Esta etiqueta es comparada con cada uno de los resultados de la base de datos para calcular las métricas de rendimiento. Esto se realiza para cada extensión de ventana analizado.

Es importante mostrar cómo es que va cambiando la distribución de atributos conforme se va incrementando el número de muestras por ventana. Estas son mostradas en las Figs. 4.4–4.8.

La distribución se reduce conforme el número de muestras incrementa como se muestra en las Figs. 4.4–4.6.

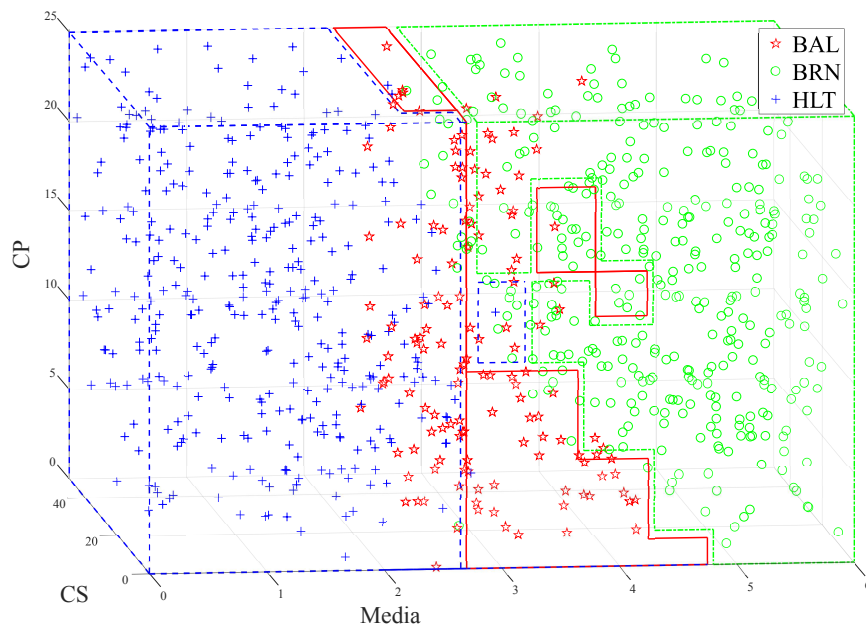


Fig. 4.4: Distribución de clasificación usando ventana de diez muestras.

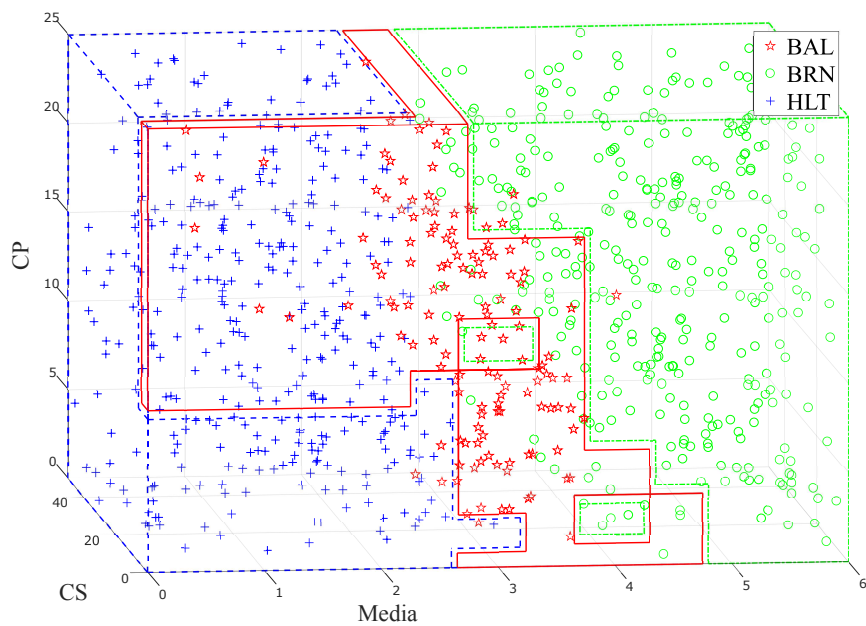


Fig. 4.5: Distribución de clasificación usando ventana de quince muestras.

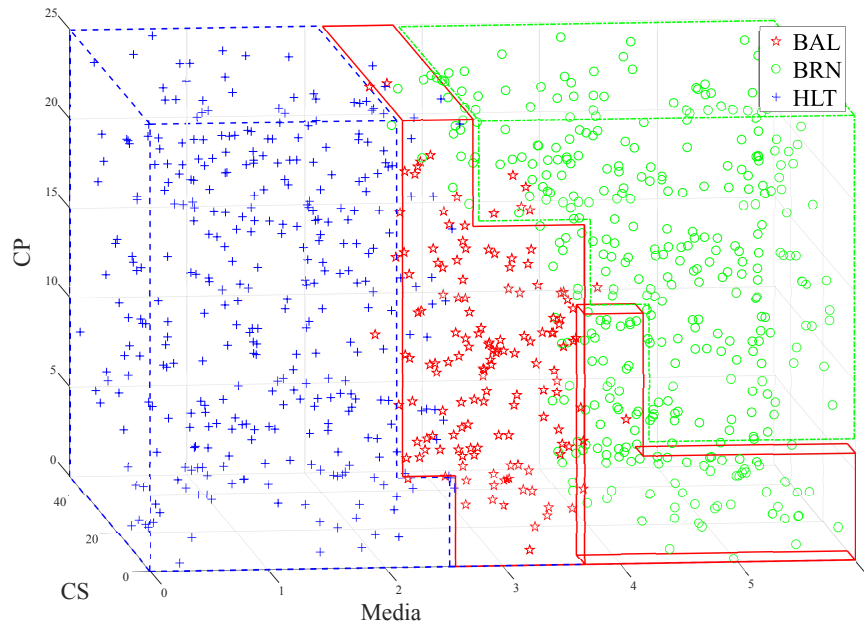


Fig. 4.6: Distribución de clasificación usando ventana de treinta muestras.

Cuando el número de muestras por ventana es mayor de 100, la distribución de clasificación presenta una mayor forma regular para una mejor clasificación, como se muestra en las Figs. 4.7 y 4.8. Es estos casos la clasificación es determinada solamente por la media y esto genera un plano como la cantidad límite de clases.

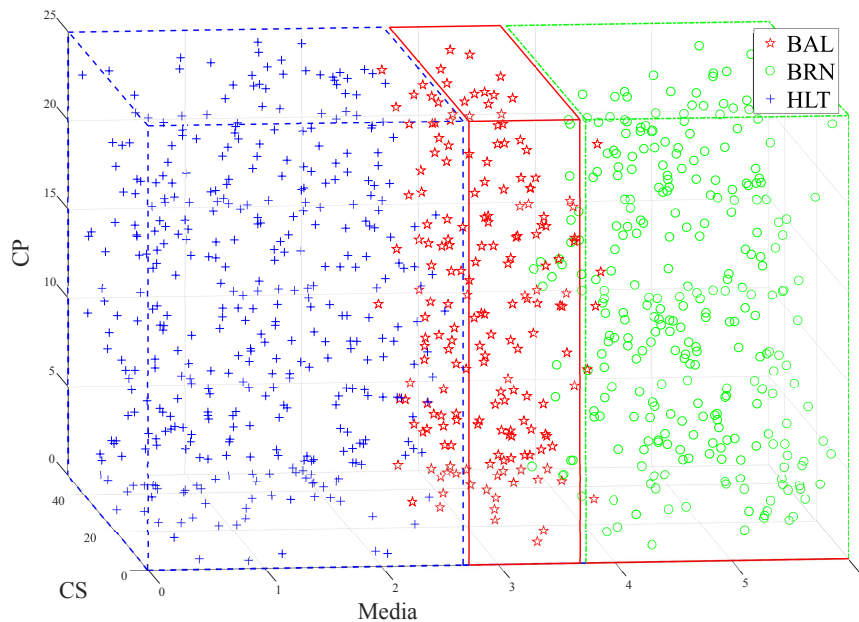


Fig. 4.7: Distribución de clasificación usando ventana de cien muestras.

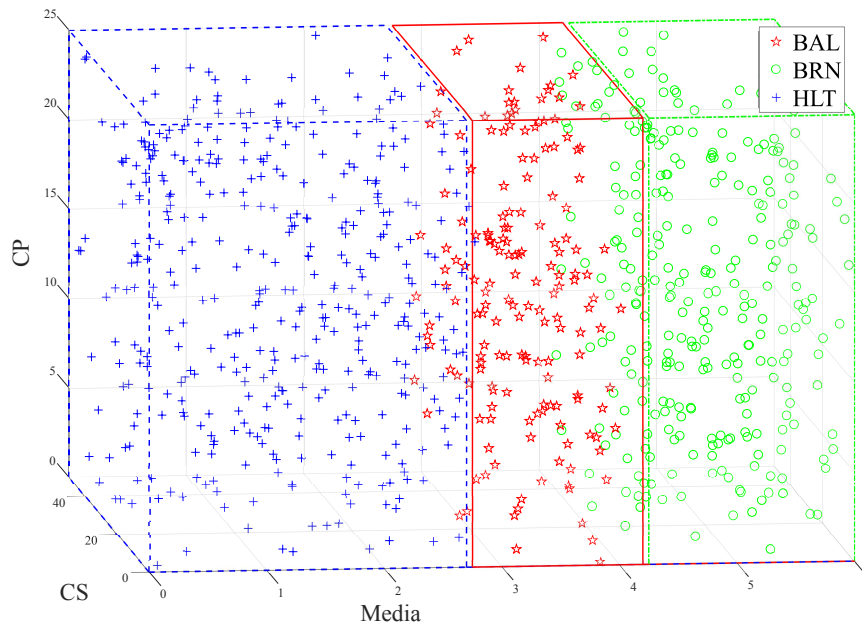


Fig. 4.8: Distribución de clasificación usando ventana de doscientas muestras

4.2.1. Árboles de Decisión

En el estado del arte de este trabajo se presentaron distintos métodos que son aplicados para la clasificación, tal es el caso de *SVM*, *kNN* y *ANN*. Sin embargo, el método basado en el árbol de decisión es elegido por presentar una alta asertividad en la clasificación de datos con una estructura simple para su implementación en hardware. Los árboles de clasificación generados utilizan los valores estadísticos en sus ramas, lo cual proporciona una mejor clasificación para el banco de pruebas con cada ventana. En las Figs. 4.9 y 4.10 se muestran las ramas y los resultados de evaluación con ventanas de cincuenta y doscientas muestras respectivamente. El árbol de clasificación necesita de los valores de la μ , el *CP* y *CS* para obtener una evaluación. La cantidad de nodos del árbol obtenida en el entrenamiento aumenta si la cantidad de muestras por ventana es baja y disminuye si la cantidad de muestras aumenta.

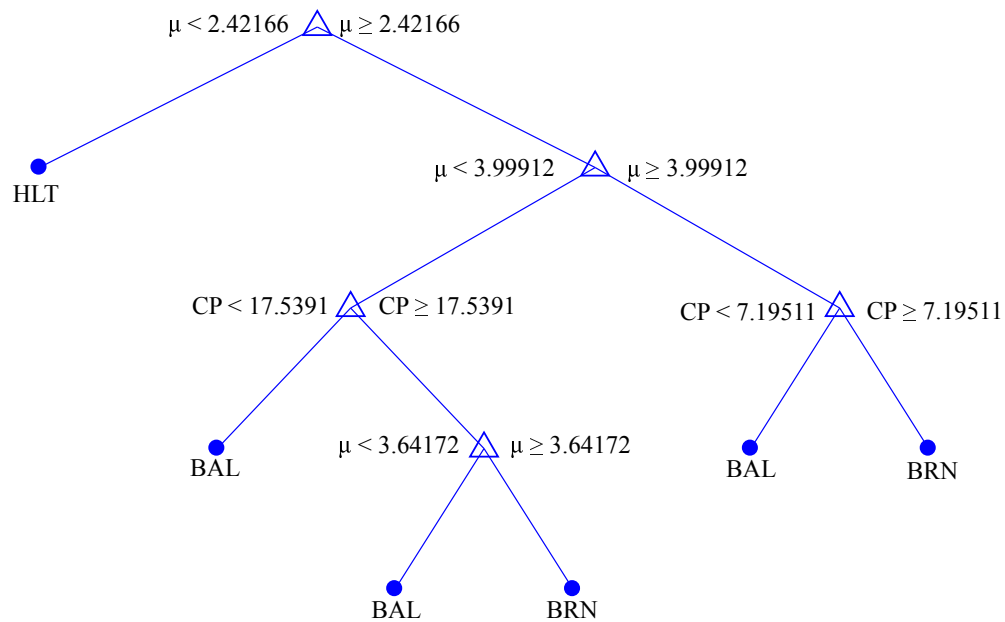


Fig. 4.9: Árbol de clasificación generado con ventana de cincuenta muestras.

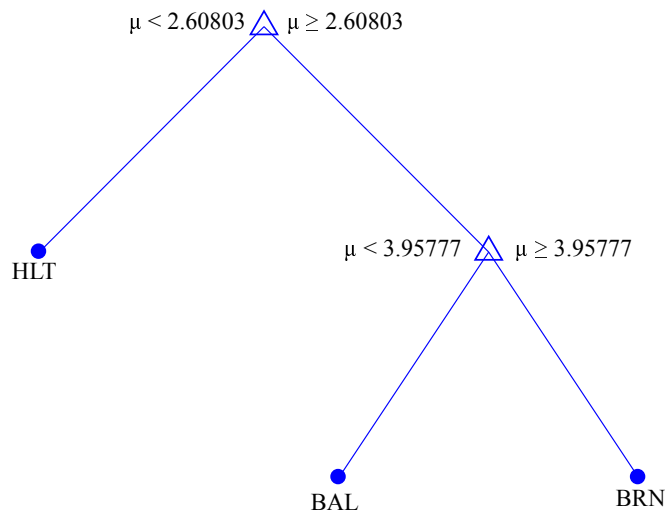


Fig. 4.10: Árbol de clasificación generado con ventana de doscientas muestras.

4.2.2. Implementación en Matlab

Un factor importante en la comprobación del sistema es la evaluación de la exactitud de la clasificación general. La evaluación compara la exactitud de las tres clases con diferentes números de muestras en la ventana. Para encontrar la eficacia del método, se implementa usando MATLAB R2017a con la activación del procesamiento paralelo. La simulación fue ejecutada en una computadora personal DELL optiplex 7040 con procesador Intel™ Core

i7, 4 núcleos, 3.4 GHZ, memoria cache de 8 MB y memoria RAM 8 GB.

La Fig. 4.11 presenta las exactitudes máximas, mínimas y promedio obtenidas en cada variación de muestras por ventana aplicando todos los archivos del banco de pruebas. Nótese que conforme el número de muestras incrementa, la exactitud lo hace rápidamente. Con alrededor de cien muestras de ventana, la exactitud de clasificación es aceptable. Además, el promedio de la exactitud se encuentra cargado hacia la exactitud máxima, lo que implica que el sistema tiende a entregar clasificaciones con exactitudes altas.

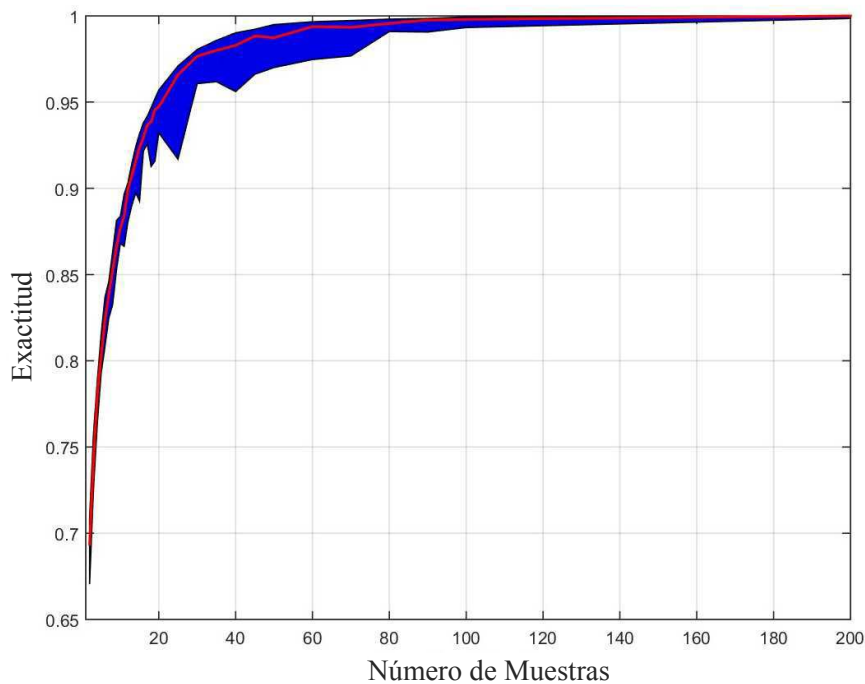


Fig. 4.11: Exactitud general de clasificación.

En la Fig. 4.12 se muestran las exactitudes de clasificación para cada una de los estados del motor con diferentes números de muestras de ventana.

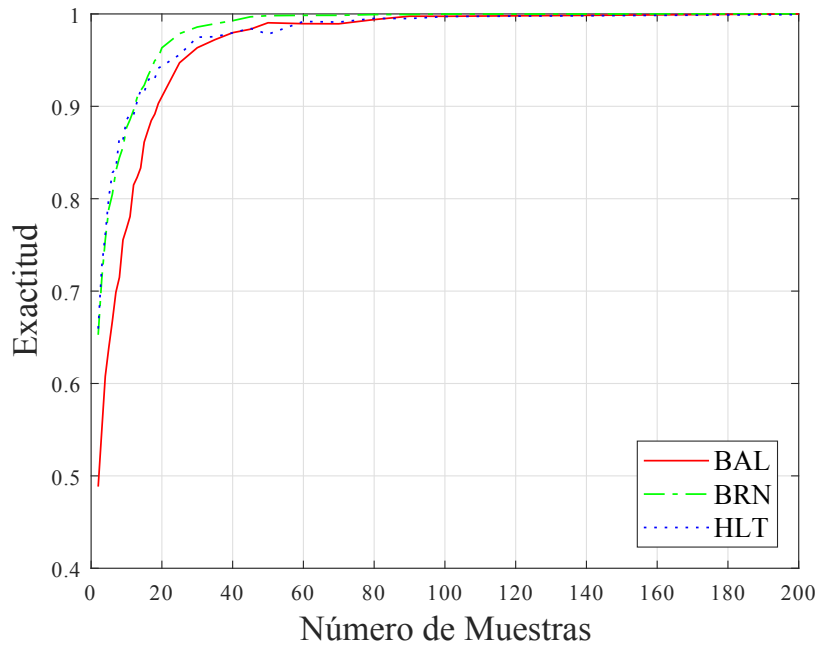


Fig. 4.12: Exactitud en la clasificación por estado.

En la Tabla 4.2 se muestran las exactitudes resultantes para el método propuesto. En esta tabla, se presentan los porcentajes de exactitudes promedio, mínimas y máximas junto con los porcentajes de error para cada clase. La identificación de la condición *HLT* presenta una mayor exactitud que la identificación de las condiciones *BRN* y *BAL* con una menor cantidad de muestras. Conforme las muestras aumentan, *BRN* es la que más fácil se identifica.

Tabla 4.2: Exactitud promedio, mínima y máxima de la clasificación.

Muestras	Clase Verdadera	Clase Identificada		
		BAL	BRN	HLT
		<i>Promedio(min,max)</i>	<i>Promedio(min,max)</i>	<i>Promedio(min,max)</i>
2	BAL	0.53 (0.43,0.70)	0.22 (0.07,0.32)	0.25 (0.15,0.39)
	BRN	0.27 (0.17,0.37)	0.65 (0.56,0.75)	0.08 (0.06,0.11)
	HLT	0.22 (0.07,0.42)	0.04 (0.00,0.13)	0.74 (0.50,0.93)
5	BAL	0.69 (0.56,0.81)	0.15 (0.05,0.27)	0.16 (0.08,0.28)
	BRN	0.19 (0.12,0.27)	0.79 (0.70,0.86)	0.02 (0.01,0.03)
	HLT	0.15 (0.02,0.38)	0.00 (0.00,0.06)	0.85 (0.58,0.98)
10	BAL	0.80 (0.66,0.91)	0.10 (0.03,0.19)	0.10 (0.03,0.25)
	BRN	0.12 (0.07,0.18)	0.88 (0.81,0.93)	0.00 (0.00,0.01)
	HLT	0.09 (0.00,0.40)	0.00 (0.00,0.02)	0.91 (0.60,1.00)
15	BAL	0.88 (0.76,0.97)	0.06 (0.01,0.14)	0.06 (0.01,0.16)
	BRN	0.08 (0.04,0.13)	0.92 (0.87,0.96)	0.00 (0.00,0.00)
	HLT	0.07 (0.00,0.51)	0.00 (0.00,0.01)	0.93 (0.49,1.00)
30	BAL	0.96 (0.86,1.00)	0.02 (0.00,0.08)	0.02 (0.00,0.11)
	BRN	0.01 (0.00,0.07)	0.99 (0.93,1.00)	0.00 (0.00,0.00)
	HLT	0.02 (0.00,0.43)	0.00 (0.00,0.01)	0.98 (0.57,1.00)
100	BAL	0.99 (0.95,1.00)	0.01 (0.00,0.05)	0.00 (0.00,0.00)
	BRN	0.00 (0.00,0.00)	1.00 (1.00,1.00)	0.00 (0.00,0.00)
	HLT	0.01 (0.00,0.11)	0.00 (0.00,0.00)	0.99 (0.89,1.00)
200	BAL	1.00 (1.00,1.00)	0.00 (0.00,0.00)	0.00 (0.00,0.00)
	BRN	0.00 (0.00,0.00)	1.00 (1.00,1.00)	0.00 (0.00,0.00)
	HLT	0.01 (0.00,0.07)	0.00 (0.00,0.00)	0.99 (0.93,1.00)

El método propuesto es aplicado a un banco de pruebas similar al que previamente se analizó, cambiando la frecuencia de muestreo de 1.5 KHz a 750 Hz. La variación en la cantidad de muestras de ventana es igual a las aplicadas en las pruebas anteriores. La comparación del promedio de exactitudes con los resultados obtenidos de la señal a una frecuencia de muestreo de 1.5KHz se muestra en la Fig. 4.13. La exactitud del método no se ve afectada

por la frecuencia de muestreo de la señal de entrada.

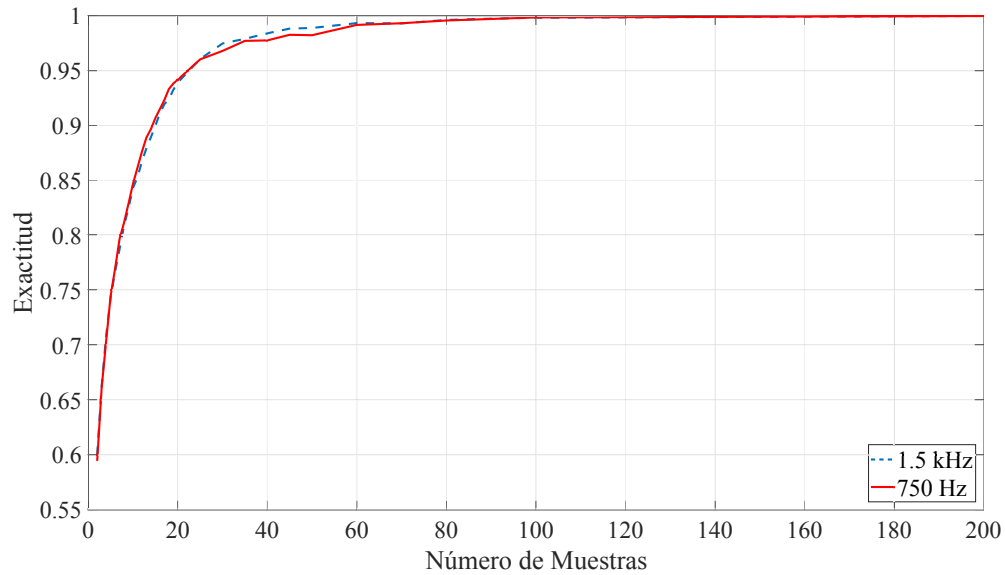


Fig. 4.13: Comparación de exactitud de clasificación general del QSA con baja frecuencia

4.2.3. Implementación en FPGA

El algoritmo *QSA* es implementado en el *FPGA* que fue presentado en la sección configuración experimental.

El uso de un sistema digital implica que los valores presenten un error en las señales de entrada, en los coeficientes asignados al árbol de decisión y en los cálculos por la conversión realizada en un formato de punto fijo. En las Figs. 4.14–4.16 se presentan los análisis de las exactitudes generales obtenidas por estado del motor al implementar el *QSA* en *FPGA* y en *MATLAB*. Así mismo, en software son evaluados los resultados de exactitud con valores de entrada convertidos a binario (*MATLAB BIN*). Estas evaluaciones son presentadas con el fin de comparar el comportamiento provocado por el error de conversión.

Como se puede observar, la implementación en *FPGA* de la metodología presenta baja exactitud al utilizar una cantidad de muestras de ventana baja para la clasificación de *BRN* con respecto a las clasificaciones *BAL* y *HLT*. Con cantidades de muestras de ventana mayores a cuarenta, el valor de la exactitud es igual y en algunos casos aumenta. Con esto se puede concluir que la implementación del *QSA* en *FPGA* presenta una exactitud similar a la del método implementado en software y el error en la conversión de valores a binario no influye drásticamente en los resultados.

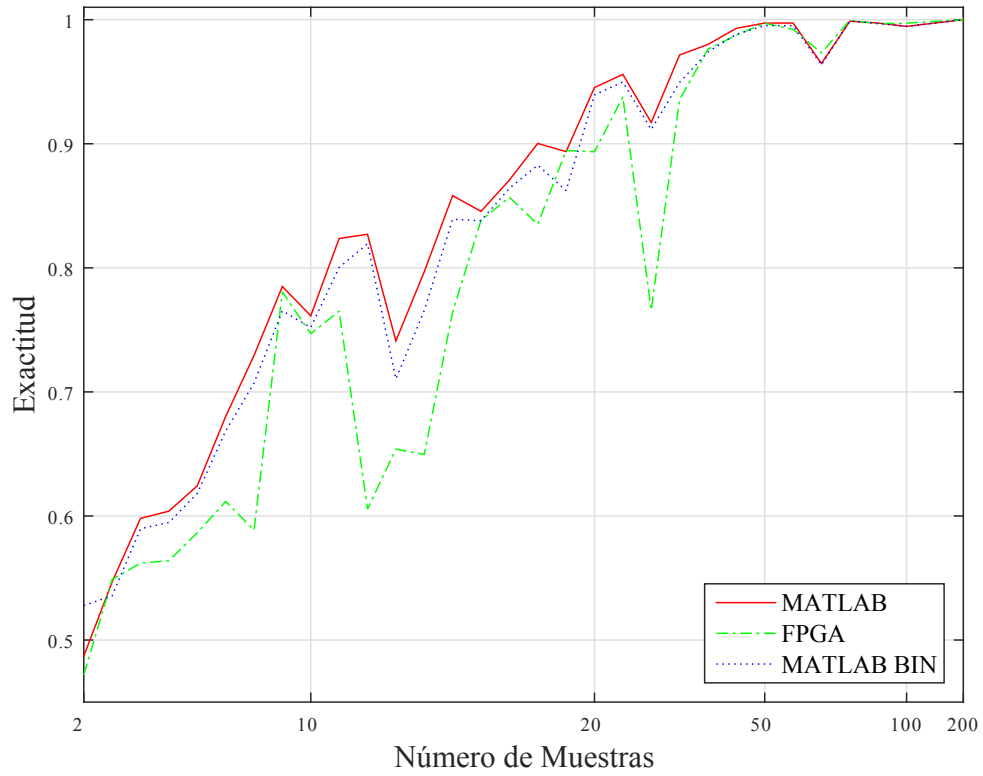


Fig. 4.14: Comparación de exactitudes de clasificación *BAL*.

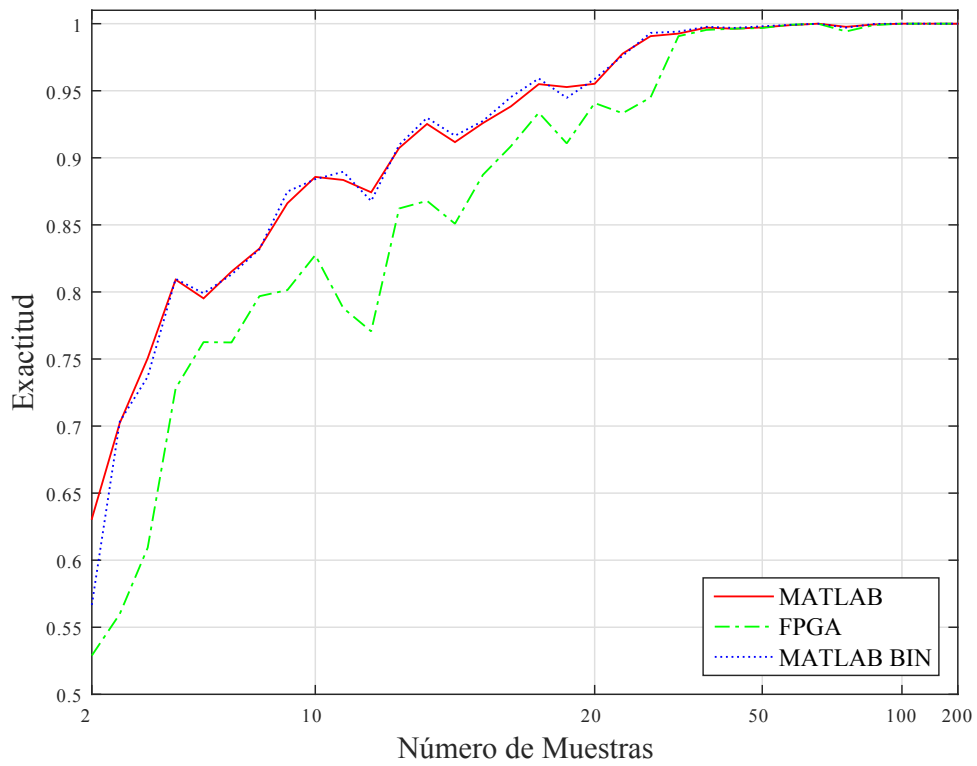


Fig. 4.15: Comparación de exactitudes de clasificación *BRN*.

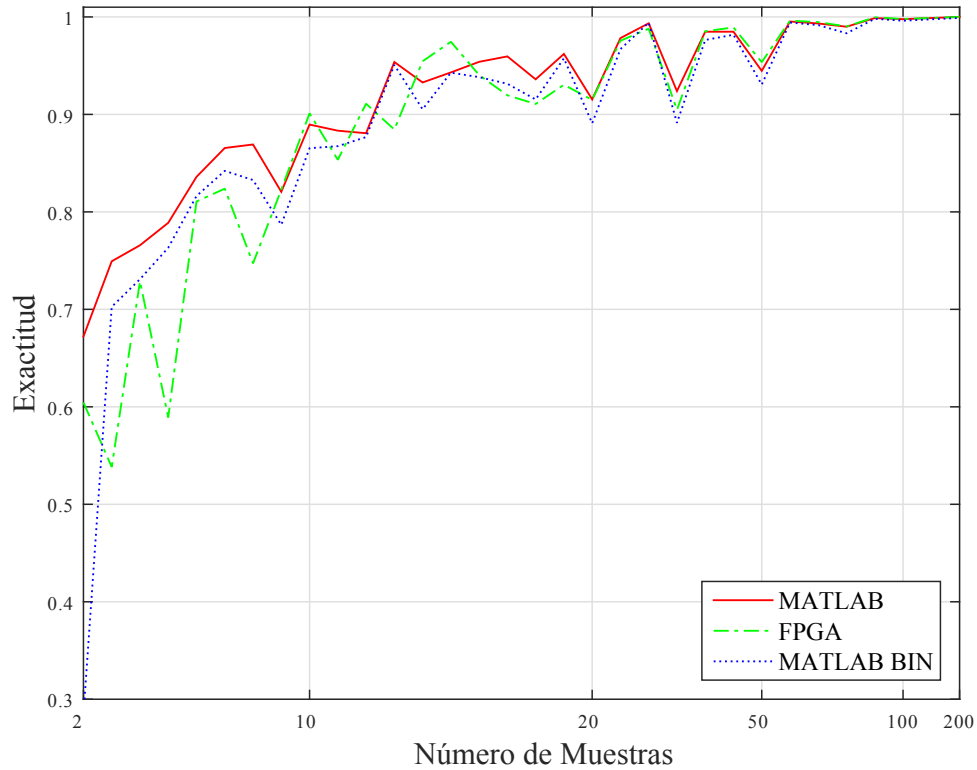


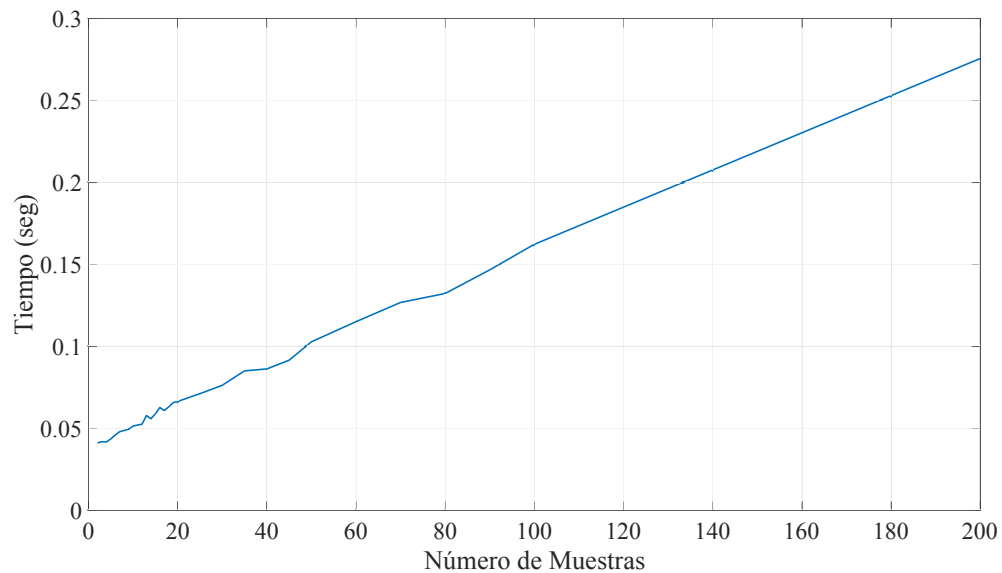
Fig. 4.16: Comparación de exactitudes de clasificación *HLL*.

En la Tabla 4.3, el método *QSA* se compara con otros trabajos donde utilizan diferentes métodos para la detección de fallas en motores. Se nota que se requieren pocas muestras para el método desarrollado en este trabajo comparadas con las de otros métodos para obtener una alta exactitud de clasificación, un alto recall promedio y una alta especificidad promedio sin necesidad de usar transformaciones espaciales, utilizando árboles de decisión y operadores de bajo costo. Esto implica la reducción del tiempo de muestreo.

Tabla 4.3: Comparación de métodos en trabajos relacionados con el QSA en Matlab.

Método	Clasificación	Clases	Muestras	Tiempo de Muestreo	Precisión	Recall	Especificidad
CCA [44]	Red Neuronal Jerárquica	6	10 000	1 s	0.8-1.0	0.97	0.99
AAC [45]	Inteligencia Artificial	3	100 000	5 s	0.71-1.0		
PCA & LDA [28]	Red Neuronal	5	270 000	1 s	0.90-0.92	0.92	0.98
SMOTE [46]	ADABOOST	5	800 000	10 s	0.8-1.0	0.93	
QSA	Árbol	3	2	1.3 ms	0.53-0.74	0.64	0.78
(Nuestro Desarrollo)	de Decisión		30	20 ms	0.96-0.99	0.97	0.98
			200	130 ms	0.99-1.0	0.99	0.99

El método propuesto presenta tiempos de procesamiento bajo para cada muestra, como se muestra en la Fig. 4.17. El tiempo promedio fue obtenido de la implementación del método usando MATLAB R2017a. Durante la simulación, el tiempo de procesamiento fue de 0.0412 segundos a 0.2756 segundos desde dos hasta doscientas muestras de ventana respectivamente.

**Fig. 4.17:** Tiempo computacional.

Los resultados de la comparación numérica del método *QSA* implementado en *FPGA* con los métodos desarrollados en otros trabajos se muestran en la Tabla 4.4. Como se puede

observar, el método implementado utiliza doscientas muestras para la obtención de 100% del banco de pruebas con clasificación correcta, mientras que el método *SMOTE* utiliza ochocientas mil muestras y el método *CCA* diez mil. La exactitud en la clasificación del *QSA* implementado en *FPGA* es mayor que otros métodos con baja latencia y presenta un tiempo de procesamiento bajo, lo que permite su ejecución en línea.

Tabla 4.4: Comparación de métodos en trabajos relacionados con el *QSA* implementado en *FPGA*.

Método	Clases	Muestras	Tiempo de Procesamiento	Latencia	Precisión
CCA [44]	6	10 000	1 s	-	0.80-1.00
AAC [45]	3	100 000	5 s	-	0.71-1.00
PCA & LDA [28]	5	270 000	1 s	90 s	0.90-0.92
SMOTE [46]	5	800 000	-	10 s	0.8-1.0
QSA (Nuestro Desarrollo)	3	2	4.23 μ s	1.33 ms	0.47-0.60
30		4.21 μ s	0.02 s	0.76-0.98	
200		4.09 μ s	0.133 s	1.00	

En la Tabla 4.5 se muestran los porcentajes recursos utilizados de registros lógicos (*LR*), elementos de multiplicación embebidos (*EME*), funciones combinacionales (*CF*), elementos lógicos (*LE*) y bits de memoria (*MB*). Como se observa en la tabla, el valor de *MB* es el único recurso que disminuye significativamente a medida que aumenta el tamaño de la cantidad de muestras de la ventana. Esto se debe a la reducción de nodos en los árboles de decisión, lo que implica el menor uso de *MB*, como se muestra en la Fig. 4.18.

Tabla 4.5: Comparación de recursos consumidos.

Muestras	LR	EME	CF	LE	MB
2	3.29	27.63	8.62	11.90	2.27
10	3.29	27.63	8.73	12.01	0.94
25	3.29	30.26	8.80	12.08	0.15
50	3.29	30.26	8.80	12.08	0.01
100	3.29	30.26	8.79	12.07	0.01
200	3.29	30.26	8.79	12.07	0.01

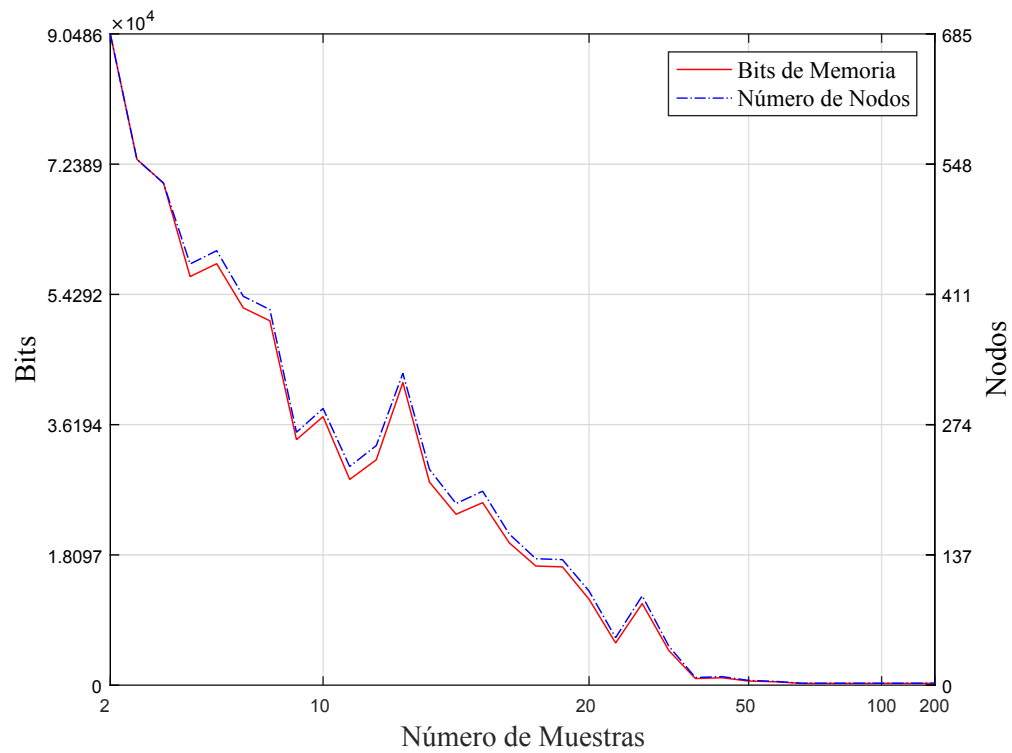


Fig. 4.18: Recursos de memoria y el número de nodos del árbol de decisión en la implementación en FPGA.

Conclusiones y Perspectivas

5.1. Conclusiones

En el trabajo se presenta el desarrollo del método *QSA* y su implementación en *FPGA* para demostrar la eficacia del método propuesto clasificando tres posibles estados de un motor de inducción. Este método realiza el análisis estadístico de cuaterniones para extraer características de la detección de fallas en motores mediante operaciones básicas, evitando los espacios de transformación. Los resultados experimentales demuestran que el método clasifica estados con buena efectividad incluso con pocas muestras. Conforme el número de muestras incrementa, la exactitud del método mejora y en consecuencia, los niveles de la separación de características incrementan. Adicionalmente, el método requiere menos datos que otros tradicionales usados para fallas en motores. Así mismo, el método propuesto permite el análisis de más de tres clases debido a las características discriminantes de las estadísticas, sin modificar el clasificador a uno más complejo.

La arquitectura implementada en *FPGA* muestra bajos y estables porcentajes de recursos consumidos, por lo que es factible su implementación en sistemas de tamaño pequeño para el manejo de una versión portátil. Así mismo, la baja cantidad de muestras por ventana permite un bajo tiempo de procesamiento, lo cual facilita el análisis de motores en línea con gran exactitud.

Los resultados obtenidos en el desarrollo del algoritmo *QSA* en software y hardware demuestran ser un excelente método para la clasificación de estados de motores en inducción, lo que permitió la publicación de dos artículos JCR en revistas de alto impacto.

- "Quaternion Signal Analysis Algorithm for Induction Motor Fault Detection", IEEE

Transactions on Industrial Electronics, Editorial IEEE, ISSN: 0278-0046 (IF 7.503), vol. 66, no. 11, pp. 8843-8850, Nov. 2019. doi: 10.1109/TIE.2019.2891468

- "Motor fault detection using Quaternion Signal Analysis on FPGA", Measurement, Editorial Elsevier, ISSN: 0263-2241, (IF 2.791), vol. 138, pp. 416-424, 2019. doi: 10.1016/j.measurement.2019.01.088.

5.2. Perspectivas

Las investigaciones futuras deberán centrarse en aumentar el número de clases agregando las combinaciones de fallas, con altos resultados de clasificación y manteniendo una baja cantidad de muestras de procesamiento para desarrollar un sistema portátil que permita la adquisición de señales, el entrenamiento de árboles, el procesamiento estadístico de cuaterniones y la clasificación del estado del motor directo de la línea de producción. Así mismo, como parte de trabajo a futuro se buscará analizar el motor en estado transitorio para comprobar las propiedades que pueden ser evaluadas con el método *QSA*.

Bibliografía

- [1] M. Eltabach, A. Charara, and I. Zein, “A comparison of external and internal methods of signal spectral analysis for broken rotor bars detection in induction motors,” *IEEE Transactions on Industrial Electronics*, vol. 51, pp. 107–121, feb 2004.
- [2] S. Smith, *The Scientist and Engineer’s Guide to Digital Signal Processing*. California Technical Pub, 1997.
- [3] R. R. Schoen, T. G. Habetler, F. Kamran, and R. Bartfield, “Motor bearing damage detection using stator current monitoring,” *IEEE transactions on industry applications*, vol. 31, no. 6, pp. 1274–1279, 1995.
- [4] T. Chow and G. Fei, “Three phase induction machines asymmetrical faults identification using bispectrum,” *IEEE Transactions on Energy Conversion*, vol. 10, no. 4, pp. 688–693, 1995.
- [5] J. R. Cameron, W. T. Thomson, and A. B. Dow, “Vibration and current monitoring for detecting airgap eccentricity in large induction motors,” *IEE Proceedings B - Electric Power Applications*, vol. 133, pp. 155–163, May 1986.
- [6] J. Vince, *Quaternions for Computer Graphics*, p. 140. London: Springer London, 2011.
- [7] P. Batres-Mendoza, C. Montoro-Sanjose, E. I. Guerra-Hernandez, D. L. Almanza-Ojeda, H. Rostro-Gonzalez, R. J. Romero-Troncoso, and M. A. Ibarra-Manzano, “Quaternion-based signal analysis for motor imagery classification from electroencephalographic signals,” *Sensors*, vol. 16, pp. 1–17, 2016.
- [8] R. Romero-Troncoso, *Electronica digital y logica programable*. Universidad de Guanajuato, 2016.

- [9] R. J. Romero-Troncoso, R. Saucedo-Gallaga, E. Cabal-Yepez, A. Garcia-Perez, R. A. Osornio-Rios, R. Alvarez-Salas, H. Miranda-Vidales, and N. Huber, “FPGA-based online detection of multiple combined faults in induction motors through information entropy and fuzzy inference,” *IEEE Transactions on Industrial Electronics*, vol. 58, no. 11, pp. 5263–5270, 2011.
- [10] K. Park, C.-L. Jeong, S.-T. Lee, and J. Hur, “Early detection technique for stator winding inter-turn fault in BLDC motor using input impedance,” *2013 IEEE Energy Conversion Congress and Exposition*, vol. 51, no. 1, pp. 4453–4459, 2013.
- [11] J. L. Barrera Alvarez and F. E. Hernandez Montero, “Classification of mpsk signals through eighth-order statistical signal processing,” *IEEE Latin America Transactions*, vol. 15, no. 9, pp. 1601–1607, 2017.
- [12] I. R. S. Gregori, I. Sanches, and C. E. Thomaz, “Clutch judder classification and prediction: A multivariate statistical analysis based on torque signals,” *IEEE Transactions on Industrial Electronics*, vol. 64, pp. 4287–4295, May 2017.
- [13] F. Elasha, D. Mba, and C. Ruiz-Carcel, “A comparative study of adaptive filters in detecting a naturally degraded bearing within a gearbox,” *Case Studies in Mechanical Systems and Signal Processing*, vol. 3, pp. 1–8, 2016.
- [14] V. C. M. N. Leite, J. G. Borges Da Silva, G. F. Cintra Veloso, L. E. Borges Da Silva, G. Lambert-Torres, E. L. Bonaldi, and L. E. De Lacerda De Oliveira, “Detection of localized bearing faults in induction machines by spectral kurtosis and envelope analysis of stator current,” *IEEE Transactions on Industrial Electronics*, vol. 62, no. 3, pp. 1855–1865, 2015.
- [15] J. Tian, C. Morillo, M. H. Azarian, and M. Pecht, “Motor bearing fault detection using spectral Kurtosis-based feature extraction coupled with K-nearest neighbor distance analysis,” *IEEE Transactions on Industrial Electronics*, vol. 63, no. 3, pp. 1793–1803, 2016.
- [16] Q. Wang, P. Li, and Y. Kim, “A parallel digital vlsi architecture for integrated support vector machine training and classification,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, pp. 1471–1484, Aug 2015.

- [17] C. Pezzani, P. Donolo, A. Guzman, G. Bossio, M. Donolo, and E. Z. Stanley, “Detecting broken rotor bars with zero-setting protection,” *IEEE Transactions on Industry Applications*, vol. 50, no. 2, pp. 1373–1384, 2014.
- [18] V. Climente-Alarcon, J. A. Antonino-Daviu, F. Vedreno-Santos, and R. Puche-Panadero, “Vibration transient detection of broken rotor bars by PSH sidebands,” *IEEE Transactions on Industry Applications*, vol. 49, no. 6, pp. 2576–2582, 2013.
- [19] M. Engin Cemal, “Novel quaternion-valued least-mean kurtosis adaptive filtering algorithm based on the GHR calculus,” *IET Signal Processing*, vol. 12, pp. 487–495, jun 2018.
- [20] D. Xu, Y. Xia, and D. P. Mandic, “Optimization in Quaternion Dynamic Systems: Gradient, Hessian, and Learning Algorithms,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 2, pp. 249–261, 2016.
- [21] M. Zajac and M. Sulowicz, “Wavelet detectors and for extraction and of characteristic and features of induction and motor rotor and faults,” *2016 International Conference on Signals and Electronic Systems (ICSES)*, pp. 5–7, 2016.
- [22] Y. Gritli, S. B. Lee, F. Filippetti, and L. Zarri, “Advanced diagnosis of outer cage damage in double-squirrel-cage induction motors under time-varying conditions based on wavelet analysis,” *IEEE Transactions on Industry Applications*, vol. 50, no. 3, pp. 1791–1800, 2014.
- [23] K. D. Kompella, V. G. R. Mannam, and S. R. Rayapudi, “Bearing fault detection in a 3 phase induction motor using stator current frequency spectral subtraction with various wavelet decomposition techniques,” *Elsevier Ain Shams Engineering Journal*, 2017.
- [24] A. Bellini, A. Yazidi, F. Filippetti, C. Rossi, and G. Capolino, “High frequency resolution techniques for rotor fault detection of induction machines,” *IEEE Transactions on Industrial Electronics*, vol. 55, no. 12, 2008.
- [25] Y. H. Kim, Y. W. Youn, D. H. Hwang, J. H. Sun, and D. S. Kang, “High-resolution parameter estimation method to identify broken rotor bar faults in induction motors,” *IEEE Transactions on Industrial Electronics*, vol. 60, no. 9, pp. 4103–4117, 2013.

- [26] D. Xu, C. Jahanchahi, C. C. Took, and D. P. Mandic, “Enabling quaternion derivatives: the generalized HR calculus: Table 1.,” *Royal Society Open Science*, vol. 2, p. 150255, aug 2015.
- [27] S. P. Talebi and D. P. Mandic, “A quaternion frequency estimator for three-phase power systems,” *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2015-August, no. 1, pp. 3956–3960, 2015.
- [28] J. Saucedo-Dorantes, M. Delgado-Prieto, R. Osornio-Rios, and R. Romero-Troncoso, “Multi-fault diagnosis method applied to an electric machine based on high-dimensional feature reduction,” *IEEE Transactions on Industry Applications*, vol. 9994, no. c, pp. 1–1, 2016.
- [29] M. Kang, J. Kim, and J. Kim, “An fpga-based multicore system for real-time bearing fault diagnosis using ultrasampling rate ae signals,” *IEEE Transactions on Industrial Electronics*, vol. 62, pp. 2319–2329, April 2015.
- [30] R. D. Stefano, S. Meo, and M. Scarano, “Induction motor faults diagnostic via artificial neural network (ann),” *IEEE Proceedings of 1994 IEEE International Symposium on Industrial Electronics*, pp. 155–162, 1994.
- [31] G. Wang, Y. Liu, and T. Zhao, “A quaternion-based switching filter for colour image denoising,” *Signal Processing*, vol. 102, pp. 216 – 225, 2014.
- [32] C. E. Moxey, S. J. Sangwine, and T. A. Ell, “Hypercomplex correlation techniques for vector images,” *IEEE Transactions on Signal Processing*, vol. 51, no. 7, pp. 1941–1953, 2003.
- [33] M. Dzwonkowski, R. Rykaczewski, and B. Czaplewski, “Digital fingerprinting based on quaternion encryption for image transmission,” *Telecommunication review + Telecommunication news*, vol. 8-9, pp. 792–798, 09 2013.
- [34] P. Ginzberg and A. T. Walden, “Matrix-valued and quaternion wavelets,” *IEEE Transactions on Signal Processing*, vol. 61, pp. 1357–1367, March 2013.






- [35] N. L. Bihan, S. J. Sangwine, and T. A. Ell, “Instantaneous frequency and amplitude of orthocomplex modulated signals based on quaternion fourier transform,” *Signal Processing*, vol. 94, pp. 308 – 318, 2014.
- [36] M. Xiang, B. S. Dees, and D. P. Mandic, “Multiple-model adaptive estimation for 3-d and 4-d signals: A widely linear quaternion approach,” *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–13, 2018.
- [37] B. C. Ujang, C. C. Took, and D. P. Mandic, “Quaternion-valued nonlinear adaptive filtering,” *IEEE Transactions on Neural Networks*, vol. 22, pp. 1193–1206, aug 2011.
- [38] R. G. Valenti, I. Dryanovski, and J. Xiao, “Keeping a good attitude: A quaternion-based orientation filter for IMUs and MARGs,” *Sensors*, vol. 15, no. 8, pp. 19302–19330, 2015.
- [39] M. A. Ibarra-Manzano, D. L. Almanza-Ojeda, and J. M. López-Hernández, “Design and optimization of real-time texture analysis using sum and difference histograms implemented on an FPGA,” *Proceedings - 2010 IEEE Electronics, Robotics and Automotive Mechanics Conference, CERMA 2010*, pp. 325–330, 2010.
- [40] L. Rokach and O. Maimon, *Decision trees*, pp. 165–192. Springer US, 2005.
- [41] R. Woods, J. McAllister, G. Lightbody, and Y. Yi, *FPGA-Based Implementation of Signal Processing Systems*. Wiley Publishing, 2nd ed., 2017.
- [42] J. L. Contreras Hernandez, D. L. Almanza-Ojeda, S. Ledesma, A. Garcia-Perez, R. d. J. Romero-Troncoso, and M. Ibarra-Manzano, “Quaternion signal analysis algorithm for induction motor fault detection,” *IEEE Transactions on Industrial Electronics*, 2019.
- [43] E. M. Hans W. Gschwind, *Design of Digital Computers*. Springer-Verlag Berlin Heidelberg, second ed., 1975.
- [44] M. Prieto Delgado, G. Cirrincione, A. Garcia Espinosa, J. A. Ortega, and H. Henao, “Bearing fault detection by a novel condition-monitoring scheme based on statistical-time features and neural networks,” *IEEE Transactions on Industrial Electronics*, vol. 60, no. 8, pp. 3398–3407, 2013.

- [45] A. Soualhi, G. Clerc, and H. Razik, “Detection and Diagnosis of Faults in Induction Motor Using an Improved Ant Clustering Technique,” *IEEE Transactions on Industrial Electronics*, vol. 60, no. 9, pp. 4053–4062, 2013.
- [46] I. Martin-Díaz, D. Morinigo-Sotelo, O. Duque-Perez, and R. J. Romero-Troncoso, “Early fault detection in induction motors using AdaBoost with imbalanced small data and optimized sampling,” *IEEE Transactions on Industry Applications*, vol. 9994, no. c, pp. 1–1, 2016.

Apéndice A

**Artículo IEEE Transactions on
Industrial Electronics**

Quaternion Signal Analysis Algorithm for Induction Motor Fault Detection

Jose L. Contreras-Hernandez, Dora Luz Almanza-Ojeda , *Member, IEEE*, Sergio Ledesma-Orozco , Arturo Garcia-Perez , *Member, IEEE*, Rene J. Romero-Troncoso , *Senior Member, IEEE*, and Mario A. Ibarra-Manzano , *Member, IEEE*

Abstract—Induction motor fault identification is essential to improve efficiency in industrial processes improving costs, production line, and maintenance time. This paper presents a novel motor fault detection methodology based on quaternion signal analysis. The proposed method establishes the quaternion coefficients as the value of motor current measurement, and the variables x , y , and z are the measurements from a triaxial accelerometer mounted on the induction motor chassis. The method obtains the rotation of quaternions and applies quaternion rotation statistics such as mean, cluster shades, and cluster prominence in order to get their features, and these are used to classify the motor state using the proposed tree classification algorithm. This methodology is validated experimentally and compared to other methods to determine the efficiency of this method for feature detection and motor fault identification and classification.

Index Terms—Fault detection, induction motors, quaternion rotation statistics, tree classification.

I. INTRODUCTION

INDUCTION motors are important in industrial processes. Motor fault detection improves costs, production line, and maintenance time. The induction motor is affected by electrodynamic forces, winding insulation, large voltage stresses, thermal aging, and mechanical vibrations from external or internal sources [1]. The typical mechanical faults that occur in induc-

tion motors are stator fault, bearing fault, broken bar fault, and rotor fault.

The importance of the evaluation of induction motors has motivated the development of algorithms for early fault detection [2]. Bearing and rotor faults are the most common type of analysis performed on induction motor failures. For instance, in [3], a stator interturn fault is detected using an algorithm that analyzes the impedance. In this paper, the impedance using the winding function theory is calculated, and then, the results are compared with those in a database to detect a fault classification.

A statistical method such as the least mean squares is used with these kinds of algorithms to obtain bearing fault detection by analyzing the vibration signals that are taken throughout the endurance test [4]. Similarly, spectral kurtosis-based algorithms are used to find faults in the bearing elements by using the stator current or the vibration signal to analyze the characteristic frequencies [5], [6].

Among the methods developed for the identification of motor faults are the space transformation methods and the estimation methods. The fast Fourier transform (FFT) and the wavelet transform are the most commonly used methods in space transformation analysis. For instance, the FFT is used to calculate the frequency spectrum of the stator currents, which can be used to obtain the number of broken bars based on the relative magnitudes of the signals [7], [8]. Similarly, a method based on the discrete wavelet transform is proposed as a viable solution for the detection of outer-cage faults of double-cage motors by analyzing the stator current spectrum [9]. On the other hand, the authors of [10] proposed an estimation method, which employs a frequency estimator, an amplitude estimator, and a fault decision module, to detect a broken rotor bar using the stator current.

Quaternions are gaining popularity in signal processing due to the reduction in the number of parameters and because they offer mathematically tractable solutions with few constraints [11]. This has motivated the development of algorithms with quaternions such as the vector correlation method that is based on the hypercomplex Fourier transform to color image processing [12]. In the same sense, the concepts of existing algorithms have been expanded to the quaternion domain, such as the distributed quaternion Kalman filtering used in estimation applications such as target tracking [13] or in multiple model adaptive estimation for three-dimensional (3-D) and four-dimensional (4-D) signals [14]. Likewise, quaternions are applied in learning systems algorithms based on the novel generalized Hamilton-real

Manuscript received June 24, 2018; revised October 6, 2018 and December 8, 2018; accepted December 9, 2018. Date of publication January 14, 2019; date of current version June 28, 2019. This work was supported in part by the Universidad de Guanajuato under Grant CIIC-3 and in part by the Consejo Nacional de Ciencia y Tecnología through Scholarship 487660. (Corresponding author: Dora Luz Almanza-Ojeda.)

J. L. Contreras-Hernandez, D. L. Almanza-Ojeda, and M. A. Ibarra-Manzano are with the Digital Signal Processing Laboratory, Division de Ingenierías Campus Irapuato Salamanca (DICIS), University of Guanajuato, Salamanca 36885, Mexico (e-mail: jose.contreras@ugto.mx; dora.almanza@ugto.mx; ibarram@ugto.mx).

S. Ledesma-Orozco is with the Computer System Department, DICIS, University of Guanajuato, Salamanca 36885, Mexico (e-mail: selo@ugto.mx).

A. Garcia-Perez is with the Electronic Engineering Department, DICIS, University of Guanajuato, Salamanca 36885, Mexico (e-mail: arturo@ugto.mx).

R. J. Romero-Troncoso is with HSPdigital-CA Mecatronica, Facultad de Ingeniería, Universidad Autónoma de Querétaro, San Juan del Rio 76807, Mexico (e-mail: troncoso@hspdigital.org).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TIE.2019.2891468

calculus [15], such as the quaternion least mean kurtosis [16], nonlinear adaptive filtering [17], and recurrent neural networks [18]. The quaternion least-mean-squares algorithm was proposed by Took and Mandic to design adaptive filtering of 3-D and 4-D processes [19]. In turn, artificial neural networks and artificial intelligence methods are limited by the significant time required for training the network with the data and their limited fault identification reliability when using untrained data [20]. Adaboost classification presents noise vulnerability [21].

The quaternion signal analysis (QSA) method allows obtaining a model of signal behavior through three- or four-signal analysis simultaneously in the time domain. This method considers each sample as a point, which shapes trajectories over time providing high levels of accuracy from a small number of samples. In this paper, a novel algorithm is proposed to classify three states of an induction motor using at least two samples by means of the QSA method. Current and accelerometer signals are used to create the quaternion, which is rotated and analyzed by means of second-order statistical methods. Finally, a classification tree is applied to the vector of characteristics in order to obtain the induction motor conditions; these are labeled in this paper as HLT (for healthy), BAL (for mechanical unbalance fault), and BRN (for bearing fault). In this paper, different algorithm parameters are changed, and the resulting performance is compared to find the best system.

Many methods have been used in the identification of faults in induction motors. Some of them are known as spectral methods and are based on the FFT, which has some drawbacks including poor resolution, spectral leakage, inefficiency to provide a good time–frequency relation, etc. [22]. It is never an easy task to find if a fault has occurred, and less so if the magnitude of the fault is low when compared to the noise produced in the induction motor. Thus, to overcome these drawbacks of the FFT, many advanced signal processing techniques have been implemented, such as zoom FFT, ESPRIT, and MUSIC algorithms as high-resolution techniques; however, spectral leakage is still a challenging task for researchers; the computation costs are high, and the techniques are very sensitive to the parameters used, which sometimes results in the misidentification of faults (spurious fault frequencies) [23]. Other so-called statistical methods have the characteristic that the physical sense is lost; they require a large amount of data and involve high computation costs.

To address these problems, QSA is used because it provides high speed and accuracy with a small number of samples. A simple classification tree is needed to obtain the induction motor conditions. Thus, the advantage of QSA is that it does not require a complex or advanced classifier to obtain the induction motor conditions. In this paper, QSA classifies three states of an induction motor using at least two samples with a processing time of 0.0412 s.

II. QUATERNION

The quaternion domain is considered a noncommutative extension of complex numbers with four components (1 , i , j , and k) that allows for a unified treatment of 4-D processes. The quaternion structure is defined by Vince [24] as

$$q = q_0 + q_1 i + q_2 j + q_3 k \quad (1)$$

where q_0 , q_1 , q_2 , and q_3 correspond to the magnitude of each component. Quaternions have three imaginary operators

$$i^2 = j^2 = k^2 = i \cdot j \cdot k = -1. \quad (2)$$

The quaternion components abide by the following product rules:

$$i \cdot j = -j \cdot i = k \quad (3)$$

$$j \cdot k = -k \cdot j = i \quad (4)$$

$$k \cdot i = -i \cdot k = j. \quad (5)$$

These properties allow operations such as multiplications as described in [24]

$$\begin{aligned} q \otimes p = & (q_0 p_0 - q_1 p_1 - q_2 p_2 - q_3 p_3) \\ & + (q_0 p_1 + q_1 p_0 + q_2 p_3 - q_3 p_2) i \\ & + (q_0 p_2 - q_1 p_3 + q_2 p_0 - q_3 p_1) j \\ & + (q_0 p_3 + q_1 p_2 - q_2 p_1 + q_3 p_0) k. \end{aligned} \quad (6)$$

The coefficients of a quaternion can be represented as a vector as follows:

$$q = [q_0, q_1, q_2, q_3]. \quad (7)$$

Rotation quaternions are elements used to represent rotations in 3-D, and they are related to the axis–angle representation of rotation. They follow Euler’s rotation theorem, whereby any rotation can be specified using a unit vector and an angle θ . The axis–angle components (θ , \hat{x} , \hat{y} , \hat{z}) can be converted to a rotation quaternion q as

$$q_0 = \cos\left(\frac{\theta}{2}\right) \quad (8)$$

$$q_1 = \hat{x} \sin\left(\frac{\theta}{2}\right) \quad (9)$$

$$q_2 = \hat{y} \sin\left(\frac{\theta}{2}\right) \quad (10)$$

$$q_3 = \hat{z} \sin\left(\frac{\theta}{2}\right) \quad (11)$$

where θ is the rotation angle and \hat{x} , \hat{y} , and \hat{z} are the unit vectors that represent the rotation axis [24]. The vector v is a 3-D vector, $v = v_1 i + v_2 j + v_3 k$, and can be expressed as

$$v = [v_1, v_2, v_3]^T. \quad (12)$$

Then, the rotation of a 3-D vector v is

$$v' = q \otimes v \otimes q^{-1} \quad (13)$$

where v' is the rotated vector, q is a quaternion, and q^{-1} is the inverse of this quaternion [24].

The last equation can be expressed as in [25], and it is adapted to the variable $q(t)$ that is present quaternion. Thus, $q(t + \Delta t)$ is a shifted quaternion, while $q_{\text{rot}}(t)$ is the rotation vector, and it is defined as

$$q(t + \Delta t) = q_{\text{rot}}(t) \otimes q(t) \otimes q_{\text{rot}}^{-1}(t). \quad (14)$$

In the same way, the rotation can be expressed as

$$q(t + \Delta t) = R(t) \cdot q(t). \quad (15)$$

Valenti *et al.* [26] show the rotation matrix R resulting from $q_{\text{rot}}(t)$. If that substitution is represented as $R(t)$, the structure is given by

$$R(t) = \begin{bmatrix} 1 - 2r_2^2 - 2r_3^2 & 2(r_1r_2 + r_0r_3) & 2(r_1r_3 - r_0r_2) \\ 2(r_1r_2 - r_0r_3) & 1 - 2r_1^2 - 2r_3^2 & 2(r_2r_3 + r_0r_1) \\ 2(r_1r_3 + r_0r_2) & 2(r_2r_3 - r_0r_1) & 1 - 2r_1^2 - 2r_2^2 \end{bmatrix} \quad (16)$$

in which $r = [r_0, r_1, r_2, r_3]$ and

$$\begin{aligned} r_0 &= q_{0,\text{rot}}(t) \\ r_1 &= q_{1,\text{rot}}(t) \\ r_2 &= q_{2,\text{rot}}(t) \\ r_3 &= q_{3,\text{rot}}(t). \end{aligned} \quad (17)$$

A quaternion can be used to model 3-D signal vibrations of the behavior of a machine. Assume that we have a set of discrete signals $I(t)$, $x(t)$, $y(t)$, and $z(t)$ with m samples, where $I(t)$ is the current, and $x(t)$, $y(t)$, and $z(t)$ are the accelerometer coordinates; then, these four signals can be represented as the quaternion

$$q(t) = I(t) + x(t)i + y(t)j + z(t)k \quad (18)$$

or in a vector form

$$q(t) = [I(t), x(t), y(t), z(t)]^T. \quad (19)$$

Analyzing the signal of the last equation in time and taking into consideration m samples with a Δt time shift in each sample, the rotation equation (15) is applied recurrently until obtaining the estimation of the $R(t)$ model, which describes the time signal behavior as

$$\begin{bmatrix} q(t + \Delta t) \\ q(t + 2\Delta t) \\ q(t + 3\Delta t) \\ \vdots \\ q(t + m\Delta t) \end{bmatrix} = \begin{bmatrix} R(t) \\ R(t + \Delta t) \\ R(t + 2\Delta t) \\ \vdots \\ R(t + (m-1)\Delta t) \end{bmatrix} \bullet \begin{bmatrix} q(t) \\ q(t + \Delta t) \\ q(t + 2\Delta t) \\ \vdots \\ q(t + (m-1)\Delta t) \end{bmatrix}. \quad (20)$$

This model is expressed as a quaternion through (16) and (17) obtaining $m - \Delta t$ models, which can be expressed as

$$Q_R = \begin{bmatrix} q_{\text{rot}}(t) \\ q_{\text{rot}}(t + \Delta t) \\ q_{\text{rot}}(t + 2\Delta t) \\ \vdots \\ q_{\text{rot}}(t + (m-1)\Delta t) \end{bmatrix} = \begin{bmatrix} R(t) \\ R(t + \Delta t) \\ R(t + 2\Delta t) \\ \vdots \\ R(t + (m-1)\Delta t) \end{bmatrix}. \quad (21)$$

III. STATISTICS

The features that describe the model behavior can be obtained by performing a statistical analysis of Q_R . For instance, the mathematical expectation can be calculated through the modulus

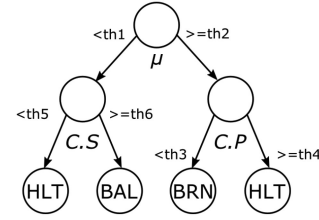


Fig. 1. Structure of the classification tree.

of the N models of Q_R as described in [27]

$$\mu(t + k\Delta t) = \frac{1}{N} \sum_l |q_{\text{rot}}(t + (k+l)\Delta t)|. \quad (22)$$

Using this value, we can obtain second-order characteristics, such as the cluster shades (CS) and the cluster prominence (CP). The CS are the measure of skewness of a matrix, i.e., the lack of symmetry. According to [27], the CS can be rewritten in vector form as

$$\text{CS}(t + k\Delta t) = \frac{1}{N} \sum_l (|q_{\text{rot}}(t + (k+l)\Delta t)| - \mu(t + k\Delta t))^3. \quad (23)$$

Similarly, the CP indicates the lack of symmetry, and it can be written as

$$\text{CP}(t + k\Delta t) = \frac{1}{N} \sum_l (|q_{\text{rot}}(t + (k+l)\Delta t)| - \mu(t + k\Delta t))^4. \quad (24)$$

Through these statistic values, a characteristic vector can be created as

$$w(t + k\Delta t) = [\mu(t + k\Delta t), \text{CS}(t + k\Delta t), \text{CP}(t + k\Delta t)]^T. \quad (25)$$

IV. CLASSIFICATION TREE

A classification tree is a tool in machine learning that divides the feature space into two parts, with each resulting part divided into two again, and so on, for $w(t + k\Delta t)$ values, as shown in Fig. 1.

The top node, known as “the root node,” is the most effective classification node. The last nodes are called “leaves” and can be eliminated in order to secure the best classification results according to the number of samples that present errors. This kind of classification tree is known as the “pruning decision tree” [28]. The classification tree was trained with 6.66% of the test bench data (as described later) to associate each motor property to a label $L(t + k\Delta t)$ as

$$L(t + k\Delta t) = \{\text{HLT}, \text{BAL}, \text{BRN}\}. \quad (26)$$

Once training is completed, the decision tree can classify a set of characteristic vectors with a response $c(t + k\Delta t)$ to each one of the induction motor conditions as

$$\mathbf{W} = \{(w(t + \Delta t), c(t + \Delta t)), (w(t + 2\Delta t), c(t + 2\Delta t)) \dots (w(t + k\Delta t), c(t + k\Delta t))\}. \quad (27)$$

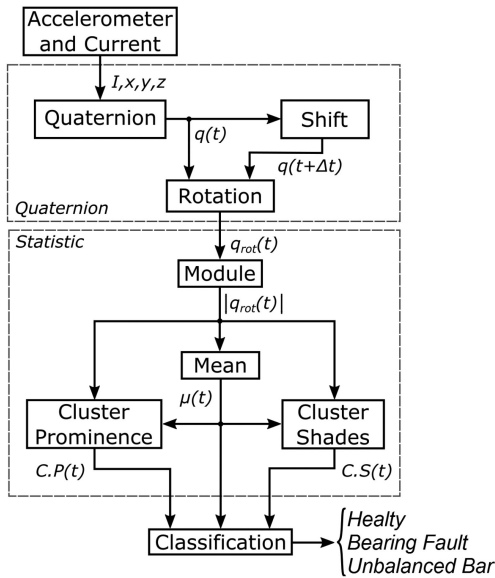


Fig. 2. Algorithm of QSA.

The response $c(t + k\Delta t)$ has a label that classifies the motor condition.

V. ALGORITHM

The algorithm presented in this paper is shown in Fig. 2. It is divided into the formulation of quaternion, statistical operations, and classification. The quaternion $q(t)$ is created using measurements from accelerometer axes (x , y , and z) and current measure (I), which comes from an induction motor. The quaternion is structured as shown in (19).

After that, quaternion $q(t)$ is shifted Δt samples in the original signal to obtain $q(t + \Delta t)$, which has the same length as $q(t)$. Both quaternions are used to get the rotational quaternion $q_{rot}(t)$ by $R(t)$. The module is applied to $q_{rot}(t)$ for statistical computation. The CP and CS are calculated using the mean and module of $q_{rot}(t)$. Next, CP and CS results are applied to the classification tree to assign a label that corresponds to one of the following conditions: healthy induction motor; bearing fault; or mechanical unbalance conditions.

VI. EXPERIMENTAL SETUP

The experimental setup for this paper consists of the analysis of four signals, three axis signals from a micro-electromechanical-system-based accelerometer (LIS3L02AS4) mounted on a three-phase induction motor chassis model (WEG 00136APE48T) with a power of 0.75 hp, two poles, and 28 bars having a weight of 9 kg, a length of 239 mm, and a width of 150 mm. One signal from the measure of induction motor current acquired using a current clamp model i200s from Fluke. The experimental setup is shown in Fig. 3.

The four signals have a sampling frequency of 1.5 kHz, which provides 4096 samples per test during the steady-state operation of the motor. The bearing fault bench is obtained when the motor is running at 3402 r/min with the fault being generated by drilling one hole of 7.938-mm diameter. Similarly, the me-

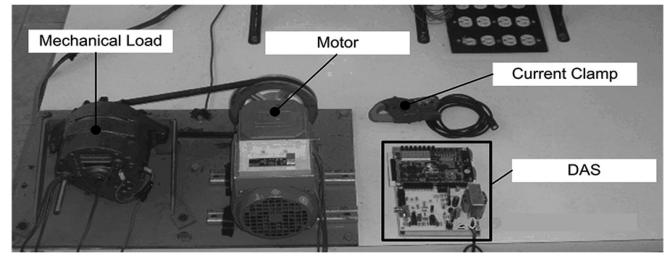


Fig. 3. Elements of the experimental setup.

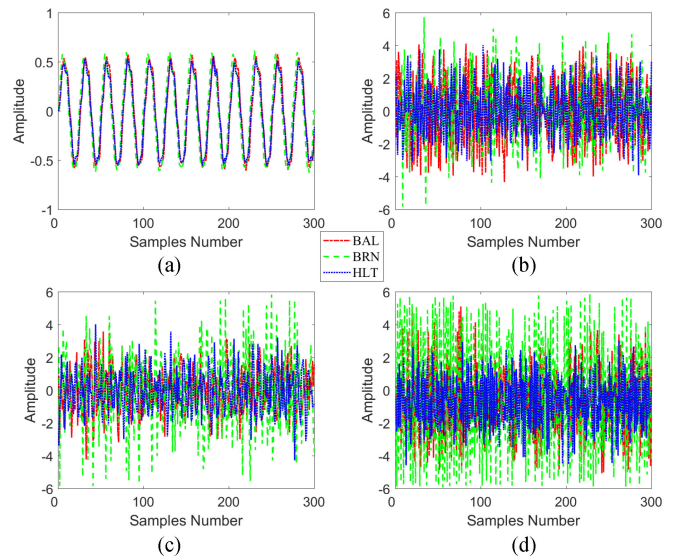


Fig. 4. Signals samples. (a) Current signals. (b) X -axis vibration signals. (c) Y -axis vibration signals. (d) Z -axis vibration signals.

chanical unbalance bar bench is acquired by adding some mass to the pulley, which produces a mass center out of the motor shaft. The test bench consists of five tests of bearing fault, 20 tests of mechanical unbalance bar, and 20 tests of a healthy induction motor. A sample of the total bench is illustrated in Fig. 4.

A test for each condition is randomly chosen to extract the CP and CS of the rotated quaternion module using a window length ranging from two samples to 200 samples. The statistical results of each condition are used to train the classifier. The algorithm is tested with the remaining bench data applying the same variable window lengths to check for assertiveness.

VII. RESULTS

The methodology proposed in this paper has been applied to the test bench from two to 200 samples. The distribution of attributes using the classification tree is shown in Figs. 5–9. The distribution is reduced as the number of samples increases, as shown in Figs. 5–7.

The classification distribution is more noticeable when the number of samples is above 100, as shown in Figs. 8 and 9. In these cases, the classification is determined only by the mean, and this generates a plane as the limiting among classes.

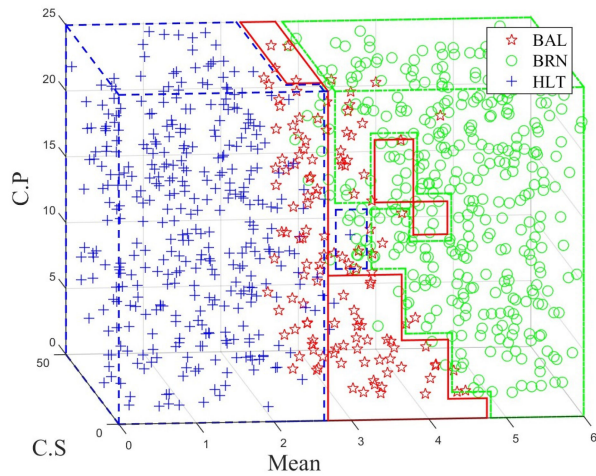


Fig. 5. Classification distribution using the ten-sample test.

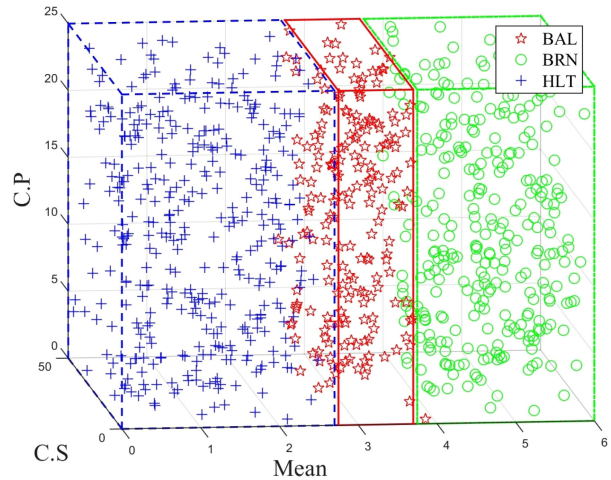


Fig. 8. Classification distribution using the 100-sample test.

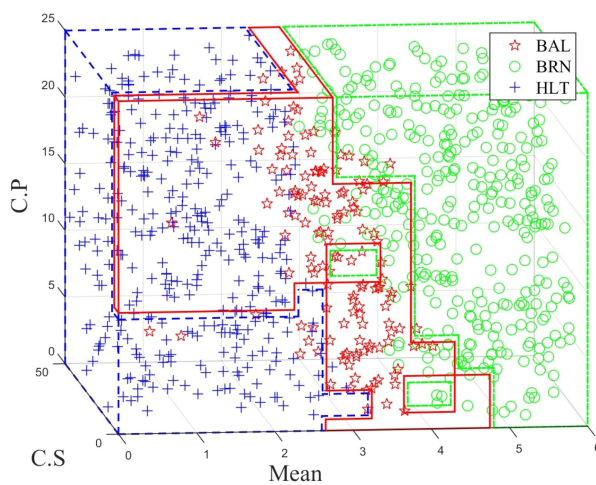


Fig. 6. Classification distribution using the 15-sample test.

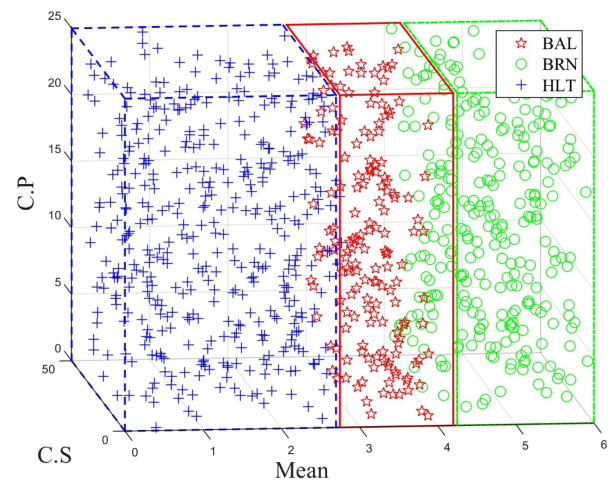


Fig. 9. Classification distribution using the 200-sample test.

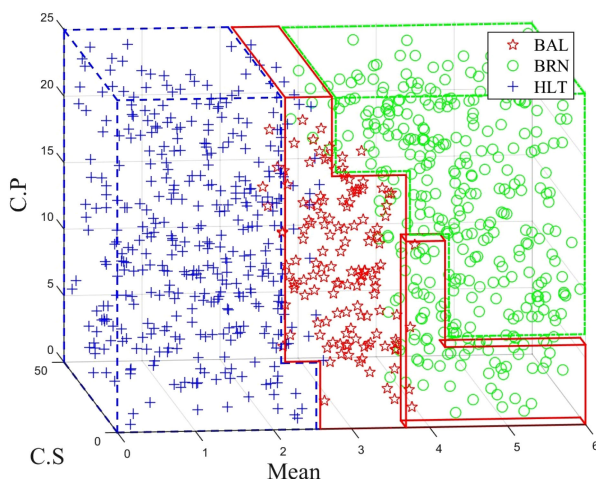


Fig. 7. Classification distribution using the 30-sample test.

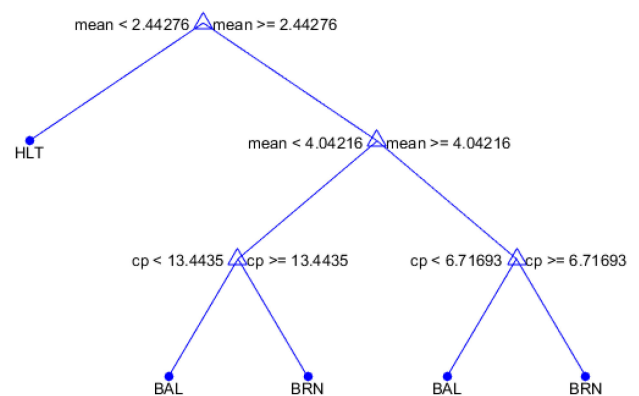


Fig. 10. Classification tree generated using 50 samples.

The generated classification trees use the statistical values in their branches, and these provide the best classification for the test bench and the number of samples applied. Figs. 10 and 11 show the branches and the evaluation results using 50 and 200 samples, respectively. The classification tree uses the mean, the

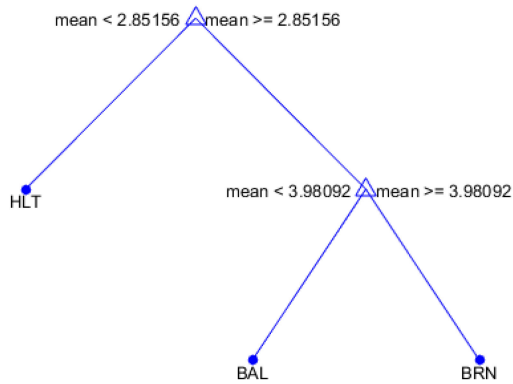


Fig. 11. Classification tree generated using 200 samples.

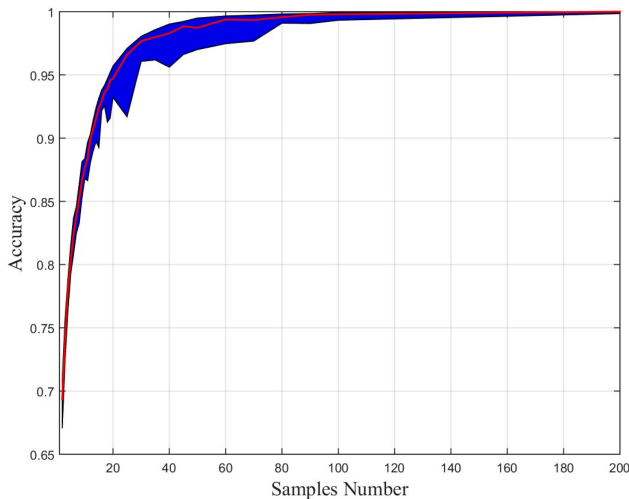


Fig. 12. General classification accuracy.

CP, or the CS to obtain an evaluation and produce a large number of nodes if the signal samples are low and vice versa.

It is important to evaluate the general classification accuracy to demonstrate the method's efficiency. The evaluation compares the accuracy of the three classes presented for different numbers of samples. Fig. 12 illustrates the maximum and minimum accuracy obtained for the test bench. Note that the accuracy increases rapidly as the number of samples increases. About 100 sample tests are enough to obtain an acceptable classification accuracy.

Fig. 13 shows classification accuracies for different numbers of samples.

In addition, the resulting accuracies for the proposed method are shown in Table I, where the mean, minimum, and maximum percents are presented including error percentages for each class. The HLT condition classification has more accuracy than the BRN and BAL condition classification with fewer samples. As the number of samples increases, the BRN condition reaches the best detection.

The proposed method is applied to a set of test signals with a sampling frequency of 750 Hz and 2–200 window samples. The accuracy averages are shown in Fig. 14, where they are compared against the results obtained from the signals obtained using a 1.5-kHz sampling frequency. The QSA method accuracy

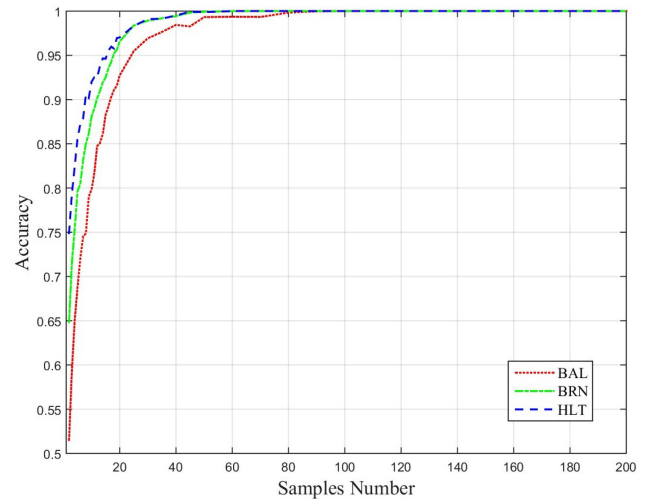


Fig. 13. Classification accuracy.

TABLE I
MEAN, MAXIMUM, AND MINIMUM CLASS CLASSIFICATION ACCURACY

Samples	True Class	Assigned Class		
		BAL	BRN	HLT
		<i>Mean(min,max)</i>	<i>Mean(min,max)</i>	<i>Mean(min,max)</i>
2	BAL	0.53 (0.43,0.70)	0.22 (0.07,0.32)	0.25 (0.15,0.39)
	BRN	0.27 (0.17,0.37)	0.65 (0.56,0.75)	0.08 (0.06,0.11)
	HLT	0.22 (0.07,0.42)	0.04 (0.00,0.13)	0.74 (0.50,0.93)
5	BAL	0.69 (0.56,0.81)	0.15 (0.05,0.27)	0.16 (0.08,0.28)
	BRN	0.19 (0.12,0.27)	0.79 (0.70,0.86)	0.02 (0.01,0.03)
	HLT	0.15 (0.02,0.38)	0.00 (0.00,0.06)	0.85 (0.58,0.98)
10	BAL	0.80 (0.66,0.91)	0.10 (0.03,0.19)	0.10 (0.03,0.25)
	BRN	0.12 (0.07,0.18)	0.88 (0.81,0.93)	0.00 (0.00,0.01)
	HLT	0.09 (0.00,0.40)	0.00 (0.00,0.02)	0.91 (0.60,1.00)
15	BAL	0.88 (0.76,0.97)	0.06 (0.01,0.14)	0.06 (0.01,0.16)
	BRN	0.08 (0.04,0.13)	0.92 (0.87,0.96)	0.00 (0.00,0.00)
	HLT	0.07 (0.00,0.51)	0.00 (0.00,0.01)	0.93 (0.49,1.00)
30	BAL	0.96 (0.86,1.00)	0.02 (0.00,0.08)	0.02 (0.00,0.11)
	BRN	0.01 (0.00,0.07)	0.99 (0.93,1.00)	0.00 (0.00,0.00)
	HLT	0.02 (0.00,0.43)	0.00 (0.00,0.01)	0.98 (0.57,1.00)
100	BAL	0.99 (0.95,1.00)	0.01 (0.00,0.05)	0.00 (0.00,0.00)
	BRN	0.00 (0.00,0.00)	1.00 (1.00,1.00)	0.00 (0.00,0.00)
	HLT	0.01 (0.00,0.11)	0.00 (0.00,0.00)	0.99 (0.89,1.00)
200	BAL	1.00 (1.00,1.00)	0.00 (0.00,0.00)	0.00 (0.00,0.00)
	BRN	0.00 (0.00,0.00)	1.00 (1.00,1.00)	0.00 (0.00,0.00)
	HLT	0.01 (0.00,0.07)	0.00 (0.00,0.00)	0.99 (0.93,1.00)

is not affected when the sampling frequency in the input signals is changed, and 1.5 kHz is an adequate sampling frequency because most of the fault spectral harmonics in the signal are kept.

In Table II, the proposed method is compared to other methods used to detect induction motor failures. It is noticeable that very few samples are required compared to the proposed methods to obtain a high accuracy classification, recall, and specificity without space transformations; this effectively reduces the sampling time.

The method proposed in this paper involves low processing times for each sample, as shown in Fig. 15. The average

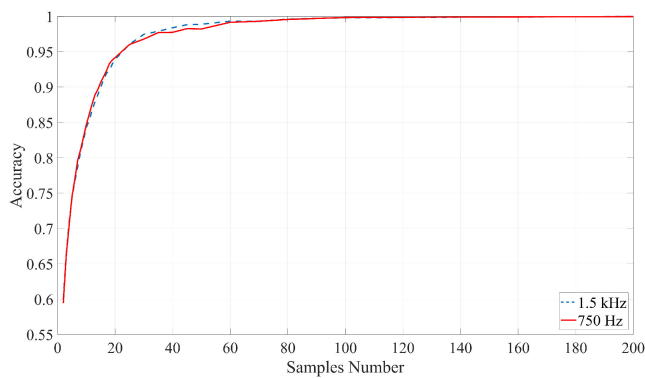


Fig. 14. General classification accuracy comparison of the QSA with low frequency.

TABLE II
COMPARISON OF METHODS

Method	Classification	Classes	Samples	Sampling time	Accuracy	Recall	Specificity	
CCA [29]	Hierarchical	6	10 000	1 s	0.8-1.0	0.97	0.99	
	Neural network							
AAC [30]	Artificial intelligence	3	100 000	5 s	0.71-1.0			
PCA & LDA [31]	Neural network	5	270 000	1 s	0.90-0.92	0.92	0.98	
SMOTE [21]	ADABOOST	5	800 000	10 s	0.8-1.0	0.93		
(Our approach)	Classification tree		2	1.3 ms	0.53-0.74	0.64	0.78	
			3	30	20 ms	0.96-0.99	0.97	0.98
			200	130 ms	0.99-1.0	0.99	0.99	

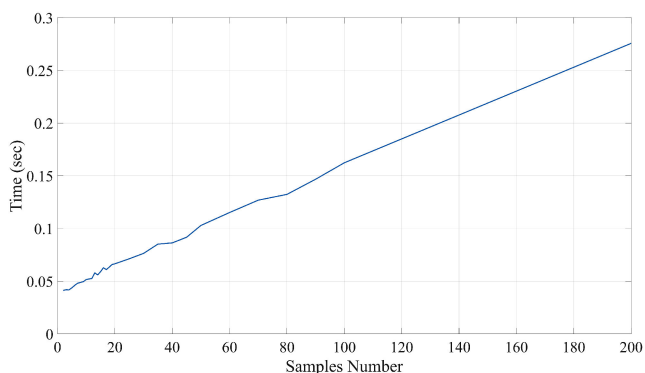


Fig. 15. Computation time.

time was captured from the implementation of the method using MATLAB R2017a to activate the parallel pool. MATLAB simulations were executed on a personal computer DELL optiplex 7040 with a 3.4-GHz IntelTM Core i7 processor, four cores, 8-MB cache, and 8-GB RAM. During the computer simulations, the processing time went from 0.0412 (when using two samples) to 0.2756 s (when using 200 samples in each test).

VIII. CONCLUSION

In this paper, the number of conditions analyzed was set to three in order to show the efficacy of the proposed method. However, the proposed methodology could be used to analyze more than three conditions. This method extract features for the detection of faults in induction motors through statistical quaternion analysis. This method used basic operations without

requiring space transformations. Experimental results showed that the proposed method obtained clear features, which could be used to detect three different states of induction motors even when only a few samples were available. As the number of samples increased, the accuracy of the method improved, and consequently, the feature-separation performance levels increased. Additionally, the proposed method required less data than other traditional methods used for fault identification in induction motors. This resulted in low data processing times, and it could be used for online monitoring in other industrial machinery to identify other faults.

REFERENCES

- [1] M. Eltabach, A. Charara, and I. Zein, "A comparison of external and internal methods of signal spectral analysis for broken rotor bars detection in induction motors," *IEEE Trans. Ind. Electron.*, vol. 51, no. 1, pp. 107–121, Feb. 2004.
- [2] R. J. Romero-Troncoso *et al.*, "FPGA-based online detection of multiple combined faults in induction motors through information entropy and fuzzy inference," *IEEE Trans. Ind. Electron.*, vol. 58, no. 11, pp. 5263–5270, Nov. 2011.
- [3] K. Park, C.-L. Jeong, S.-T. Lee, and J. Hur, "Early detection technique for stator winding inter-turn fault in BLDC motor using input impedance," in *Proc. IEEE Energy Convers. Congr. Expo.*, 2013, pp. 4453–4459.
- [4] F. Elasha, D. Mba, and C. Ruiz-Carcel, "A comparative study of adaptive filters in detecting a naturally degraded bearing within a gearbox," *Case Stud. Mech. Syst. Signal Process.*, vol. 3, pp. 1–8, 2016.
- [5] V. C. M. N. Leite *et al.*, "Detection of localized bearing faults in induction machines by spectral kurtosis and envelope analysis of stator current," *IEEE Trans. Ind. Electron.*, vol. 62, no. 3, pp. 1855–1865, Mar. 2015.
- [6] J. Tian, C. Morillo, M. H. Azarian, and M. Pecht, "Motor bearing fault detection using spectral kurtosis-based feature extraction coupled with K-nearest neighbor distance analysis," *IEEE Trans. Ind. Electron.*, vol. 63, no. 3, pp. 1793–1803, Mar. 2016.
- [7] C. Pezzani, P. Donolo, A. Guzman, G. Bossio, M. Donolo, and E. Z. Stanley, "Detecting broken rotor bars with zero-setting protection," *IEEE Trans. Ind. Appl.*, vol. 50, no. 2, pp. 1373–1384, Mar./Apr. 2014.
- [8] V. Climente-Alarcon, J. A. Antonino-Daviu, F. Vedreno-Santos, and R. Puche-Panadero, "Vibration transient detection of broken rotor bars by PSH sidebands," *IEEE Trans. Ind. Appl.*, vol. 49, no. 6, pp. 2576–2582, Nov./Dec. 2013.
- [9] Y. Gritli, S. B. Lee, F. Filippetti, and L. Zarri, "Advanced diagnosis of outer cage damage in double-squirrel-cage induction motors under time-varying conditions based on wavelet analysis," *IEEE Trans. Ind. Appl.*, vol. 50, no. 3, pp. 1791–1800, May/Jun. 2014.
- [10] Y. H. Kim, Y. W. Youn, D. H. Hwang, J. H. Sun, and D. S. Kang, "High-resolution parameter estimation method to identify broken rotor bar faults in induction motors," *IEEE Trans. Ind. Electron.*, vol. 60, no. 9, pp. 4103–4117, Sep. 2013.
- [11] S. P. Talebi and D. P. Mandic, "A quaternion frequency estimator for three-phase power systems," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2015, pp. 3956–3960.
- [12] C. E. Moxey, S. J. Sangwine, and T. A. Ell, "Hypercomplex correlation techniques for vector images," *IEEE Trans. Signal Process.*, vol. 51, no. 7, pp. 1941–1953, Jul. 2003.
- [13] S. P. Talebi, S. Kanna, and D. P. Mandic, "A distributed quaternion Kalman filter with applications to smart grid and target tracking," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 2, no. 4, pp. 477–488, Dec. 2016.
- [14] M. Xiang, B. S. Dees, and D. P. Mandic, "Multiple-model adaptive estimation for 3-D and 4-D signals: A widely linear quaternion approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 1, pp. 72–84, Jan. 2019.
- [15] D. Xu, C. Jahanchahi, C. C. Took, and D. P. Mandic, "Enabling quaternion derivatives: The generalized HR calculus: Table 1," *Roy. Soc. Open Sci.*, vol. 2, no. 8, Aug. 2015, Art. no. 150255.
- [16] E. C. Mengüç, "Novel quaternion-valued least-mean kurtosis adaptive filtering algorithm based on the GHR calculus," *IET Signal Process.*, vol. 12, no. 4, pp. 487–495, Jun. 2018.
- [17] B. C. Ujang, C. C. Took, and D. P. Mandic, "Quaternion-valued nonlinear adaptive filtering," *IEEE Trans. Neural Netw.*, vol. 22, no. 8, pp. 1193–1206, Aug. 2011.

- [18] D. Xu, Y. Xia, and D. P. Mandic, "Optimization in quaternion dynamic systems: Gradient, Hessian, and learning algorithms," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 2, pp. 249–261, Feb. 2016.
- [19] C. C. Took and D. P. Mandic, "The quaternion LMS algorithm for adaptive filtering of hypercomplex processes," *IEEE Trans. Signal Process.*, vol. 57, no. 4, pp. 1316–1327, Apr. 2009.
- [20] R. D. Stefano, S. Meo, and M. Scarano, "Induction motor faults diagnostic via artificial neural network (ANN)," in *Proc. IEEE Int. Symp. Ind. Electron.*, 1994, pp. 155–162.
- [21] I. Martin-Diaz, D. Morinigo-Sotelo, O. Duque-Perez, and R. J. Romero-Troncoso, "Early fault detection in induction motors using AdaBoost with imbalanced small data and optimized sampling," *IEEE Trans. Ind. Appl.*, vol. 53, no. 3, pp. 3066–3075, May/Jun. 2017.
- [22] K. D. Kompella, V. G. R. Mannam, and S. R. Rayapudi, "Bearing fault detection in a 3 phase induction motor using stator current frequency spectral subtraction with various wavelet decomposition techniques," *Ain Shams Eng. J.*, vol. 9, pp. 2427–2439, 2018.
- [23] A. Bellini, A. Yazidi, F. Filippetti, C. Rossi, and G. Capolino, "High frequency resolution techniques for rotor fault detection of induction machines," *IEEE Trans. Ind. Electron.*, vol. 55, no. 12, pp. 4200–4209, Dec. 2008.
- [24] J. Vince, *Quaternions for Computer Graphics*. London, U.K.: Springer, 2011, p. 140.
- [25] P. Batres-Mendoza *et al.*, "Quaternion-based signal analysis for motor imagery classification from electroencephalographic signals," *Sensors*, vol. 16, pp. 1–17, 2016.
- [26] R. G. Valenti, I. Dryanovski, and J. Xiao, "Keeping a good attitude: A quaternion-based orientation filter for IMUs and MARGs," *Sensors*, vol. 15, no. 8, pp. 19302–19330, 2015.
- [27] M. A. Ibarra-Manzano, D. L. Almanza-Ojeda, and J. M. López-Hernández, "Design and optimization of real-time texture analysis using sum and difference histograms implemented on an FPGA," in *Proc. IEEE Electron., Robot. Autom. Mech. Conf.*, 2010, pp. 325–330.
- [28] L. Rokach and O. Maimon, *Decision Trees*. New York, NY, USA: Springer, 2005, pp. 165–192.
- [29] M. Prieto Delgado, G. Cirrincione, A. Garcia Espinosa, J. A. Ortega, and H. Henao, "Bearing fault detection by a novel condition-monitoring scheme based on statistical-time features and neural networks," *IEEE Trans. Ind. Electron.*, vol. 60, no. 8, pp. 3398–3407, Aug. 2013.
- [30] A. Soualhi, G. Clerc, and H. Razik, "Detection and diagnosis of faults in induction motor using an improved ant clustering technique," *IEEE Trans. Ind. Electron.*, vol. 60, no. 9, pp. 4053–4062, Sep. 2013.
- [31] J. Saucedo-Dorantes, M. Delgado-Prieto, R. Osornio-Rios, and R. Romero-Troncoso, "Multifault diagnosis method applied to an electric machine based on high-dimensional feature reduction," *IEEE Trans. Ind. Appl.*, vol. 53, no. 3, pp. 3086–3097, May/Jun. 2017.



Jose L. Contreras-Hernandez received the B.Sc. degree in electronic engineering from the University of Guanajuato, Salamanca, Mexico, in 2012, and the M.Sc. degree in mechatronics from the Autonomous University of Queretaro, Santiago de Querétaro, Mexico, in 2014. He is currently working toward the Ph.D. degree in electric engineering with the University of Guanajuato.

He is involved in the development of a project with light visible communication. His research interests include digital design on field-programmable gate array for signal processing.



Dora Luz Almanza-Ojeda (M'04) received the B.Sc. degree in electronic engineering and the M.Eng. degree in electrical engineering from the University of Guanajuato, Salamanca, Mexico, in 2003 and 2005, respectively, and the Ph.D. (Hons.) degree in informatics and embedded systems from the University of Toulouse III: Paul Sabatier, Toulouse, France, in 2011.

She is currently an Assistant Professor with the Electronics Engineering Department, Universidad de Guanajuato. Her research interests include embedded vision for controlling autonomous robots and real-time systems.



Sergio Ledesma-Orozco received the M.S. degree from the University of Guanajuato, Salamanca, Mexico, in 1993, and the Ph.D. degree from the Stevens Institute of Technology, Hoboken, NJ, USA, in 2001.

He then worked for Barclays Bank as part of the IT-HR Group. He has worked as a Software Engineer for several years, and he is the creator of the software Neural Lab and Wintempla. He is currently a Research Professor with the University of Guanajuato. His research interests include artificial intelligence and software engineering.



Arturo Garcia-Perez (M'10) received the B.E. and M.E. degrees in electronics from the University of Guanajuato, Salamanca, Mexico, in 1992 and 1994 respectively, and the Ph.D. degree in electrical engineering from the University of Texas, Dallas, TX, USA, in 2005.

He is currently an Associate Professor with the Department of Electronic Engineering, University of Guanajuato. He is a National Researcher with the Consejo Nacional de Ciencias y Tecnologia. His research interests include digital signal processing.



Rene J. Romero-Troncoso (M'07–SM'12) received the Ph.D. degree in mechatronics from the Autonomous University of Queretaro, Santiago de Querétaro, Mexico, in 2004.

He is currently a Full Professor with the Autonomous University of Queretaro. He has been an Invited Researcher with the University of Valladolid, Valladolid, Spain. He has authored two books on digital systems (in Spanish) and coauthored more than 170 technical papers published in international journals and conferences. His research interests include hardware signal processing with a field-programmable gate array and monitoring and diagnosis on dynamic systems.

Dr. Romero-Troncoso was a recipient of the 2004 Asociacion Mexicana de Directivos de la Investigacion Aplicada y el Desarrollo Tecnológico Nacional Award on Innovation for his work in applied mechatronics and the 2005 IEEE ReConFig Award for his work in digital systems. He is a National Researcher level 3 with the National Council of Science and Technology, Mexico, and a Fellow of the Mexican Academy of Engineering.



Mario A. Ibarra-Manzano (M'05) received the B.Sc. degree in communication and electronic engineering and the M.Eng. degree in electrical engineering from the University of Guanajuato, Salamanca, Mexico, in 2003 and 2006, respectively, and the Ph.D. (Hons.) degree in microelectronics and microsystems from the Institut National des Sciences Appliquées, Toulouse, France, in 2011.

He is currently an Associate Professor with the Electronics Engineering Department, Universidad de Guanajuato. His research interests include digital design on field-programmable gate array for image processing applied on autonomous robots and real-time systems.

Apéndice B

Artículo Measurement



Motor fault detection using Quaternion Signal Analysis on FPGA [☆]

Jose L. Contreras-Hernandez, Dora L. Almanza-Ojeda, Sergio Ledesma,
Mario A. Ibarra-Manzano ^{*}

Departamento de Ingeniería Electrónica, DICIS, Universidad de Guanajuato, Carr. Salamanca-Valle de Santiago KM. 3.5 + 1.8 Km., Salamanca 36885, Mexico



ARTICLE INFO

Article history:

Received 14 August 2018
Received in revised form 27 January 2019
Accepted 29 January 2019
Available online 1 February 2019

2017 MSC:

00-01
99-00

Keywords:

QSA
Feature extraction
Fault detection
FPGA
Real time processing

ABSTRACT

In industry, the maintenance of motors strategy requires accurate monitoring techniques that indicate the motor working conditions and the type of fault detected. In this work, the Quaternion Signal Analysis (QSA) method is used to analyze statistical features of motor signals in order to detect faulty and healthy conditions in motor induction. The use of quaternion arithmetic to rotate and characterize signal behavior in a time domain provides accurate results using a reduced number of samples during classification. The accelerometer and current signals measured from the motor are modeled as quaternions. The decision trees method is used to classify healthy (*HLT*), unbalanced pulley (*BAL*) and bearing fault (*BRN*) motor conditions. The proposed methodology was implemented on an FPGA device through Very-High-Speed Hardware Description Language (VHDL) to validate the effectiveness of our method in real motor working conditions. This architecture was tested to validate the effectiveness in real motor working conditions.

© 2019 Elsevier Ltd. All rights reserved.

1. Introduction

Induction motors are an important part of the production process in industry. According to this, fault motor detection is essential to ensure the production time and to plan line maintenance without implying high processing time. Different methods are proposed to detect induction motor faults in frequency domain, as wavelets methods, which use vibration signals to detect the healthy motor state and one or two bars faults [21]. However, these methods involve high-cost operators, and generally, produce a delayed diagnostic time.

Statistical analysis provides the numeric characteristics of signal properties that can then be used to check the systems conditions. Through this, the authors in [10] developed a modulation frequencies classifier using signal-to-noise ratio (SNR) averages achieving high effectiveness. Some works focusing on mechanical classification are developed in [11], where covariance and mean of torque signals are calculated to classify and predict clutch judder through a training data matrix with 100% of accuracy. The forest

fault detection method is developed on a PC and shows a correct rate of diagnosis of 98.8%. The Fast Fourier Transform and Power Spectral Density are applied to vibration and current signals to detect faults in bearing and gears as is shown in [25]. Another work that classifies motor faults such as bearing defect, $\frac{1}{2}$ broken rotor bar, 1 broken rotor bar, unbalance and misalignment is presented in [12], where vibration signals are decomposed to calculate fifteen statistical features. After this, optimization, selection and extraction of the meaningful features are developed. Finally, classification is performed through a neural network implemented on PC. This method shows a 92.59% effectiveness. Similarly, the bearing failure diagnosis procedure is developed in [13].

Acoustic signal analysis is a popular method for motor fault detection shown in [22], in which an acoustic based diagnosis is developed to detect four motor states using the nearest neighbor classifier. Similarly, the authors in [23] use acoustic and current signals through Fast Fourier application to estimate eccentricity faults. Likewise, the Fast Fourier Transform is used to apply the shortened method of frequencies selection multiexpanded in acoustic signals to diagnose bearing, stator and rotor faults [24]. In addition, some time–frequency transformations of acoustic emission are used to compute statistical features. In this way, four bearing states are classified through multiclass support vector machines [14]. This algorithm is implemented in Xilinx Virtex-7 FPGA with 99.36% accuracy using 90.1% of RAM, 0.8% of LUT

[☆] This document is a collaborative effort.

^{*} Corresponding author.

E-mail addresses: jose.contreras@ugto.mx (J.L. Contreras-Hernandez), dora.almanza@ugto.mx (D.L. Almanza-Ojeda), selo@ugto.mx (S. Ledesma), ibarra@ugto.mx (M.A. Ibarra-Manzano).

and 0.07% of configurable logic block. The frequency sample of this method is 100 kHz allowing online processing.

On the other hand, the use of quaternions has increased in recent years in digital signal processes such as image filtering [1], encrypted [2] or space transformation [3,4]. For FPGA implementations, quaternions requires complex mathematical operations such as multiplication, which is usually performed by means of Coordinate Rotation Digital Computer (CORDIC) [5,6]. Likewise, in [7] quaternions are represented by a matrix, where a split-matrix approach is used to perform quaternion multiplication. In [8], matrix multiplication is used to obtain quaternion matrix rotation, which is developed by a parallel floating point multiplication in FPGA. Moreover, quaternions can be added, subtracted and divided to perform statistical operations as shown in [9].

Quaternion-based signal analysis (QSA) models and analyzes signal behavior in a time domain. Basically, the method considers the trajectory of samples as points, to estimate samples behavior achieving high accuracy for a reduced number of samples. In addition, the application of quaternions provides good trade off among computational complexity and free from Gimbal lock compared with other rotational techniques. In [15], QSA is used as a pattern recognition method in time domain of electroencephalographic signals (EEG). These signals represent brain activity related to motor imagery. Four of the EEG signals are used to form a quaternion. Moreover, the rotation of quaternions model the temporal behavior of the signals. Statistical features are calculated from the rotated quaternions and assessed using decision trees, support vector machine and k-nearest neighbor techniques achieving an accuracy rate of 84.75%, 77.35% and 83.62%, respectively.

This work implements the QSA technique to classify motor induction faults using vibration signals acquired from three axes accelerometers, and current signal acquired from a current clamp. Statistical features such as mean, cluster shade and cluster prominence are computed from measures modeled by a rotated quaternion. The use of QSA allows processing in the time domain with a reduced number of samples and high percentages of accuracy during classification. QSA method is implemented in FPGA and the low number of samples used for processing results in high speed implementation and performance of the proposed technique.

2. Methodology

Our approach identifies three motor states: unbalanced pulley (BAL), bearing fault (BRN) and healthy motor (HLT) following a reduced samples constraints strategy. The experimental data consists of four signals measured from a three-phase induction motor (WEG 00136APE48T) with 0.75 hp, 2 poles, 28 bars with a weight of 9 kg, a length of 239 mm and width of 150 mm. One of the signals is the induction motor current measured by a current clamp model i200s from Fluke. Additionally, three axes signals are acquired using a MEMS-based accelerometer (LIS3L02AS4) mounted on an induction motor chassis. The sampling frequency of the four signals is 1.5 kHz, which provides 4096 samples per test

during the steady state operation of the motor. The experimental setup is shown in Fig. 1.

The test bench is acquired with the motor running at 3402 RPM. The fault in the bearing is caused by a drilled hole with a diameter of 7.938 mm. The unbalanced pulley fault is caused by adding mass to the pulley. Both are shown in Fig. 2.

The test bench is shaped by twenty experimental tests of unbalanced pulley, five experimental tests of bearing fault and twenty experimental tests of healthy induction motor. A sample of the total bench is presented in Fig. 3.

The algorithm of the method proposed is shown in Fig. 4. The motor database measurements are modeled and analyzed in two stages: (1) QSA strategy for feature extraction and (2) Classification. Initially, the quaternion q is formed by the q_0, q_1, q_2 and q_3 elements that will be structured by current, X-, Y- and Z-axes accelerometer signals, respectively. The quaternion is delayed by an amount of time dt to obtain a current and displaced quaternion, separately. Then, a pair of displaced signal samples are used to obtain a three-dimensional model of orientations and rotations. The norm is applied to the three-dimensional model to obtain the magnitude of such quaternion. Some statistical values are calculated using N samples window of the norm value. Finally, a decision tree classifier is used to detect the motor state [16].

The QSA method consists of four steps: normalization, rotation, norm and statistical features extraction as is presented in [16]. Initially, the quaternion normalization stage allows the correct separation of statistical properties and improves the scalability of the system. The quaternion normalization of \mathbf{q} is calculated as:

$$\mathbf{q}_n = \frac{1}{\sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}} [q_0, q_1, q_2, q_3] \quad (1)$$

After the normalization process, the quaternion rotation q_{rot} is calculated using the equation presented in (2), in which only the complex components of the quaternion are used. The resulting values are the components of the rotation between the actual quaternion signal and the Δt delayed quaternion signal. Thus, this rotation describes the behavior in the present signal to obtain the delayed signal.

$$q_{rot} = \begin{bmatrix} 1 - 2q_{n2}^2 - 2q_{n3}^2 & 2(q_{n1}q_{n2} + q_{n0}q_{n3}) & 2(q_{n1}q_{n3} - q_{n0}q_{n2}) \\ 2(q_{n1}q_{n2} - q_{n0}q_{n3}) & 1 - 2q_{n1}^2 - 2q_{n3}^2 & 2(q_{n2}q_{n3} + q_{n0}q_{n1}) \\ 2(q_{n1}q_{n3} + q_{n0}q_{n2}) & 2(q_{n2}q_{n3} - q_{n0}q_{n1}) & 1 - 2q_{n1}^2 - 2q_{n2}^2 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} \quad (2)$$

The norm of rotated quaternion $|\mathbf{q}_{rot}|$ is calculated to obtain the magnitude, which represents a scalar statistical feature as:

$$|\mathbf{q}_{rot}| = \sqrt{q_{rot1}^2 + q_{rot2}^2 + q_{rot3}^2} \quad (3)$$

Finally, statistical features as mean (μ), cluster shades (CS), cluster prominence (CP) are calculated using the following equations:

$$\begin{aligned} \mu &= \frac{1}{N} \sum x \\ CS &= \frac{1}{N} (\sum x^3 - 3\mu \sum x^2 + 3\mu^2 \sum x - \mu^3 N) \\ CP &= \frac{1}{N} (\sum x^4 - 4\mu \sum x^3 + 6\mu^2 \sum x^2 - 4\mu^3 \sum x + \mu^4 N) \end{aligned} \quad (4)$$

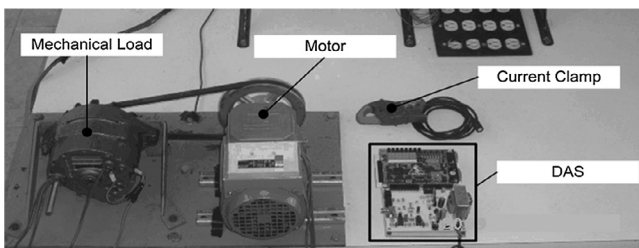


Fig. 1. Experimental setup.

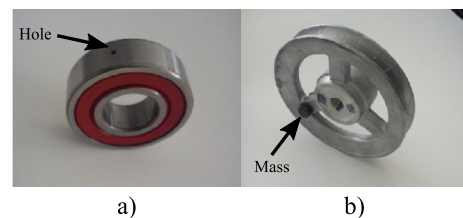


Fig. 2. Faults setup. (a) Bearing Fault. (b) Unbalanced Pulley Fault.

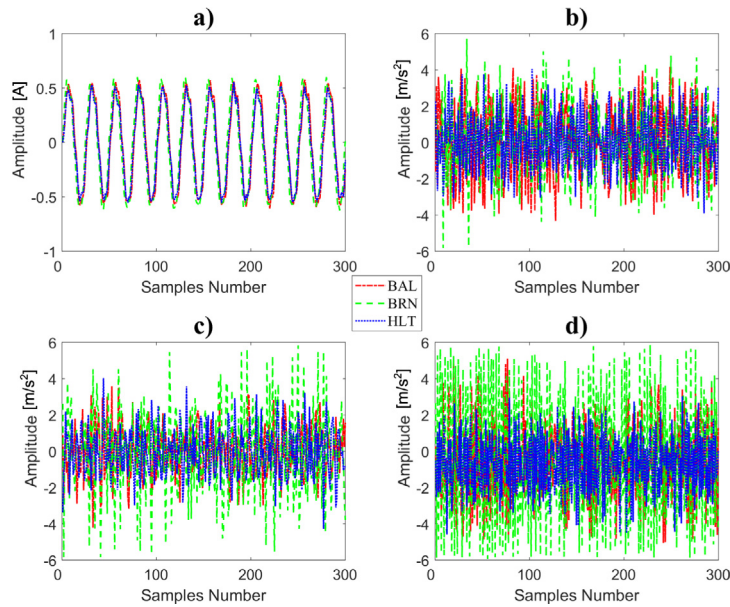


Fig. 3. Signals samples. (a) Current signals. (b) X axis vibration signals. (c) Y axis vibration signals. (d) Z axis vibration signals.

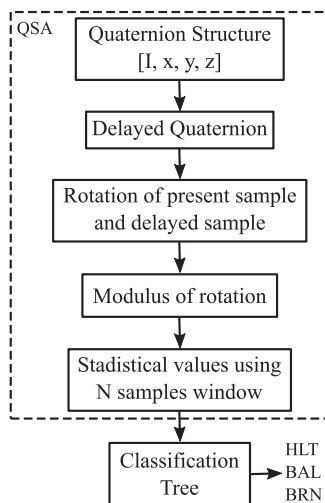


Fig. 4. Experimental setup.

These equations are an adequation to obtain statistical values in summation terms [15], where x is the calculated norm in each quaternion sample of a selected window length. Such statistical features are computed from rotated quaternions because they represent signal evolution in time.

After the QSA, a decision tree classifier calculates the statistical features to classify motor state. The next sections describe the design and implementation of the QSA and the decision three classifier using a DE2-115 board with Cyclone IV EP4CE115F19 FPGA.

3. Architecture design

The global strategy of the proposed system is presented in Fig. 5, where quaternion components are the inputs of the QSA block, a signal analysis is developed to obtain the mean, the cluster shape and the cluster prominence; finally, the classification block throws the resulted class.

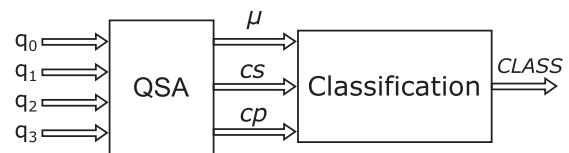


Fig. 5. General block diagram of the proposed system.

3.1. Normalization

The diagram of the system is shown in Fig. 6, where the QSA block is extended to perform the steps described above. This block is shaped by normalization block *NORZ*, hold registers *HREG*, delay register *REG*, quaternion rotation block *ROT*, three variables norm *NORM3* and statistical block *STA*.

At first, the quaternion \mathbf{q} is normalized in the *NORZ* block using the equation shown in (1). This block is made up of two internal blocks: the *NORM* block and the division *A/B* block as shown in Fig. 7(a). Quaternion normalization of \mathbf{q} is obtained through the division of each quaternion component by the quaternion norm.

The internal structure of the quaternion norm block is presented in Fig. 7(b). First, the quaternion components are multiplied by themselves to obtain the squared value q_i . Afterwards, the squared components are added yielding S to which square root is applied. The square root structure is shown in Fig. 7(c) and, it is described in Algorithm 1. This algorithm is a modified version of the square root algorithm presented in [17]. Here, the size of the logic vector S is n , internal variables D_1 and D_2 are logic vectors which are initialized with the decimal values 0 and 1 respectively. In lines 4 and 5, the *for* loop is iterated $n/2$ times, where the less significant bits of D_1 are concatenated with the more significant bits of S to reassign the D_1 vector. The logic vector D_2 is compared with D_1 in line 6, if the square of D_2 is lesser than D_1 , then, the logic bit B is set to '1' otherwise it is set to '0'. In line 11, this bit is left shifted through output variable D_o . Later, D_2 is assigned with the concatenation of the less significant bits of D_o and bit '1'. Finally in line 13, the variable S is reassigned with left shifted of S before the process is repeated. When the *for* loop is finished in line 15, the last D_o value represents the square root of S .

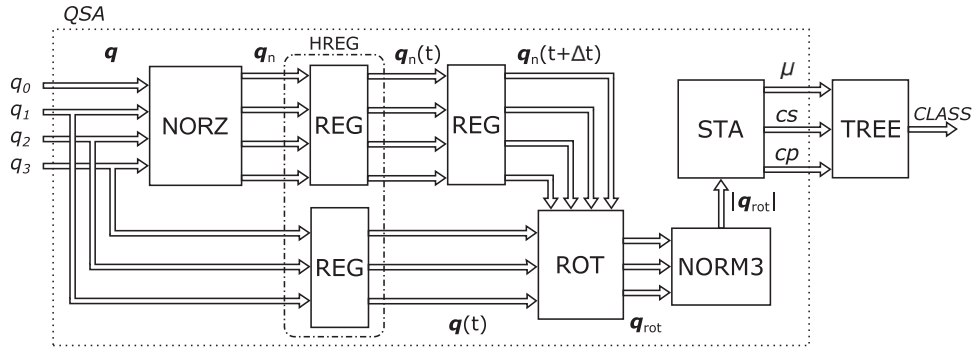


Fig. 6. Internal representation of system.

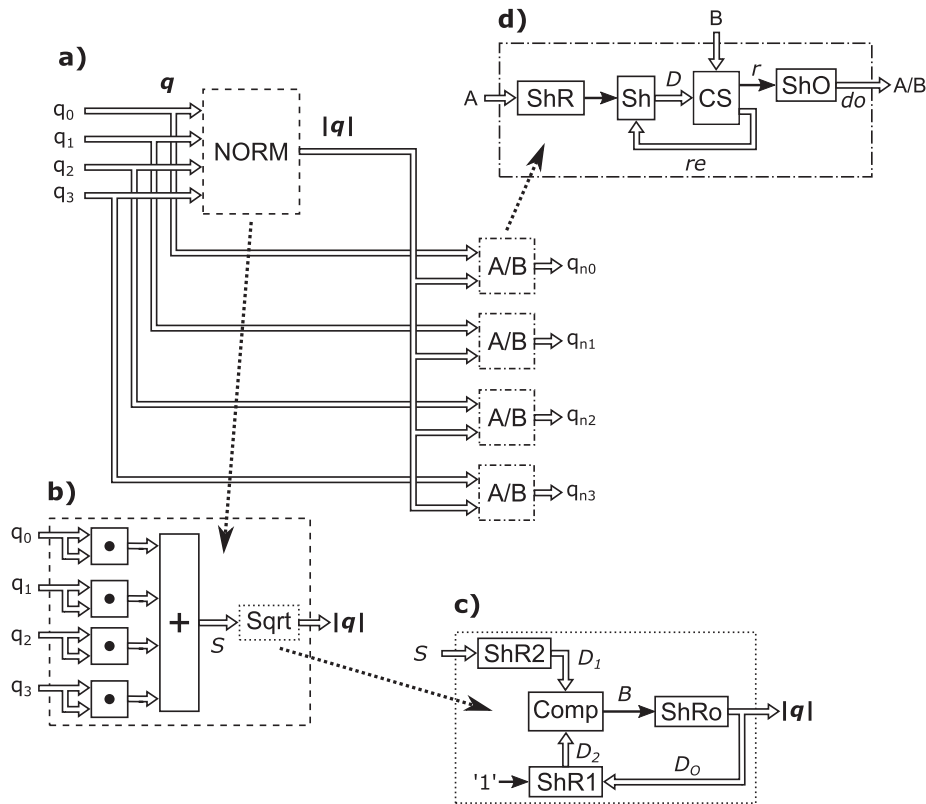


Fig. 7. Quaternion normalization block. (a) General structure. (b) Norm block structure. (c) Square root block structure. (d) Division block structure.

Algorithm 1. Square Root Algorithm

```

1: Procedure SQRT ▷ Input: S ▷ Output: Do
Require:  $n \leftarrow \text{size}(S)$ 
2:  $D_1 \leftarrow 0$ 
3:  $D_2 \leftarrow 1$ 
4: for  $i \leftarrow 1, n/2$  do
5:    $D_1 \leftarrow D_1[n - 2 : 1] \ \& \ S[n : n - 1]$ 
6:   if  $D_2^2 \leq D_1$  then
7:      $B \leftarrow '1'$ 
8:   else
9:      $B \leftarrow '0'$ 
10:  end if
11:   $D_0 \leftarrow D_0[n - 1 : 1] \ \& \ B$ 
12:   $D_2 \leftarrow D_0[n - 1 : 1] \ \& \ '1'$ 
13:   $S \leftarrow \text{left\_shift}(S)$ 
14: end for
15: end procedure
    
```

The last operation of the normalization module is a division. The internal structure of the division block is shown in Fig. 7(d) and , the pseudocode is presented in Algorithm2, where A and B represent the numerical inputs. In the algorithm, the size of the logic vector A is assigned to n and the logic vector re , which represents the remainder, is initialized to a 0 decimal value in line 2. In line 3 and 4, the for loop is repeated n times to assign logic vector D as the concatenation of the less significant bits of re with the most significant bit of A . This logic vector is compared in line 5 and if D is smaller than B the logic bit r is assigned to '0' value and logic vector re is assigned to D value. Otherwise, r is set to a value of '1' and logic vector re is $D - B$. After this evaluation, in line 12 the most significant bits of DO are concatenated with bit r and this is reassigned to the DO vector. In line 13, logic vector A is left shifted and the process is repeated using shifted A value. Finally in line 15, DO is assigned to d which is the division quotient.

Algorithm 2. Division Algorithm

```

1: Procedure DIV( $A, B, d$ ) ▷ Input:  $A, B$  ▷ Output:  $d$ 
Require:  $n \leftarrow \text{size}(A)$ 
2:  $re \leftarrow 0$ 
3: for  $i \leftarrow 1, n$  do
4:    $D \leftarrow re[n - 1 : 1] \& A[n]$ 
5:   if  $D < B$  then
6:      $r \leftarrow '0'$ 
7:      $re \leftarrow D$ 
8:   else
9:      $r \leftarrow '1'$ 
10:     $re \leftarrow D - B$ 
11:   end if
12:    $DO \leftarrow DO[n : 2] \& r$ 
13:    $A \leftarrow \text{left\_shift}(A)$ 
14: end for
15:  $d \leftarrow DO$ 
16: end procedure

```

After completing the quaternion normalization block, the HREG block is performed using q_n and q vectors as shown in Fig. 6. This block is important for the FPGA implementation because it allows the synchronization of quaternion samples. Then, $q_n(t)$ goes into delay register REG where $q_n(t + \Delta t)$ is obtained.

3.2. Quaternion rotation

Until now, $q_n(t)$ and $q_n(t + \Delta t)$ are obtained. These quaternion signals are employed to compute quaternion rotation through Eq. 2. This process is illustrated by the ROT block in Fig. 6. The resulting rotation components are normalized using the NORM3 block, which uses the Eq. 3. The internal block diagram of NORM3 is shown in Fig. 8, which is similar to the NORM block depicted in Fig. 7(c) but in this case only the three complex components are used. This block calculates the square of each component, after which, they are added to apply the Root block.

3.3. Statistical feature extraction and classification

The result of $|q_{rot}|$ is saved in a location of the random access memory implemented as an internal memory (see Fig. 9). The data stored in each memory location are read in the x variable, and then, this data is raised to the power of two, three and four. These values will be used to calculate the summation of x, x^2, x^3 and x^4 , which are used to obtain statistical values. The mean value μ is obtained by dividing the sum of x and the number of window samples selected N . Similarly, this value is raised to the power of two, three and four. Thus, the CS and CP features are calculated by multiplication, addition, subtraction and division of the x and μ terms, and the value N .

The second stage of the process consists in obtaining a classification of the statistical features from QSA. To this end, the features, μ , CS and CP are used as inputs of the classification block based on a

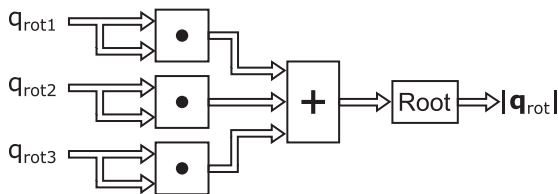


Fig. 8. Structure of three components norm.

decision tree classifier, illustrated in Fig. 10. Note that, each ROM location contains the following values: $type, lim, dir1, dir2, class$ and end . The value of dir corresponds to the actual direction of memory location whereas $type$ selects the statistical feature and lim is a threshold. A comparison is performed between the selected feature and the lim value, and if this comparison is true, then $dir1$ is the next memory direction to be extracted from the ROM, otherwise the next direction will be $dir2$. The final class is contained in a $class$ value, meanwhile the end flag indicates the end of the classification process. The implementation details of these architecture modules are provided in the next section.

The decision tree classifier is presented in Algorithm 3. In line 2, the initial direction dir is set to zero, and this memory direction contains the set up values of the current node in the tree. Then, lines 3 to 16 evaluate the decision tree until the end flag is set to 1. In lines 4 to 9, the $type$ value selects the corresponding statistical feature (among μ, CS , and CP) and it is assigned to var . After that in line 11, the comparison between the feature and the lim value is performed, thus, if the var value is smaller than the value of lim , then dir allocates the next direction indicated by $dir1$, otherwise the next direction of dir will allocate $dir2$. Once the end flag is TRUE, the $class$ node contains the classification result. A tree classification is trained for each window length selected, values obtained are stored on the FPGA. After this, the test benches of unbalanced pulley, bearing fault and healthy motor are sent to the FPGA through RS-232 communication to check their assertiveness.

4. Results

The implemented method was evaluated using three performance metrics: accuracy, recall, and specificity. The accuracy is defined as:

Algorithm 3. Decision Tree Algorithm

```

1: Procedure TREE ▷ Input:  $\mu, cp, cs$  ▷ Output: class
2:  $dir \leftarrow 0$ 
3: while  $end = FALSE$  do
4:   if  $type = "00"$  then
5:      $var \leftarrow \mu$ 
6:   else if  $type = "01"$  then
7:      $var \leftarrow cp$ 
8:   else
9:      $var \leftarrow cs$ 
10:  end if
11:  if  $var < lim$  then
12:     $dir \leftarrow dir1$ 
13:  else
14:     $dir \leftarrow dir2$ 
15:  end if
16: end while
17:  $class(Dir)$ 
18: end procedure

```

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \quad (5)$$

The recall is defined as:

$$Recall = \frac{TP}{TP + FP} \quad (6)$$

And the specificity is defined as:

$$Specificity = \frac{TN}{TN + FN} \quad (7)$$

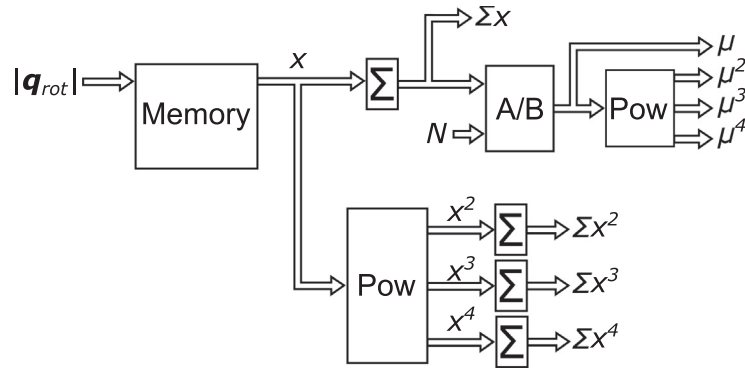


Fig. 9. Internal structure of statistical block.

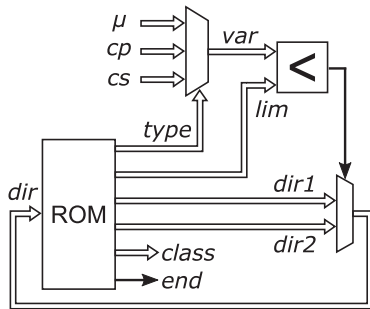


Fig. 10. Internal structure of classification block.

In Eqs. (5)–(7) are used to compute each motor state. In these equation, TP is the number of true positive stated (a positive state classified as positive), TN is the number of true negative states (a negative state classified as negative), FP is the number of false positive states (a negative state classified as positive) and FN is the number of false negative states (a positive state classified as negative). The three metrics are in the range of 0 to 1, with 1 being the

best possible value. The average for the three performance metrics was calculated using the three motor states, and the experiment was repeated twenty times to ensure repeatability.

The implemented method was tested with two kinds of motor failures using two to two-hundred window samples. The average accuracy of our method implemented in FPGA is compared with the accuracy of the software version using both floating and fixed point in Matlab. Accuracies obtained for the *BAL*, *BRN* and *HLT* motors are shown in Figs. 11–13. The FPGA implementation of the proposed methodology shows low accuracy using low window samples for the *BRN* classification with respect to the *BAL* and the *HLT* classifications. The accuracy value is similar and, in some cases, increases for windows with more than forty samples. According to this, the implementation of the QSA method in FPGA shows similar accuracy than the method implemented using software.

The QSA method is compared with other failure detection methods in induction motors as is shown in Table 1. QSA uses two hundred samples to obtain a high accuracy range with low samples, high recall and specificity. Thus, this method presents high classification using decision-tree and low-cost operators.

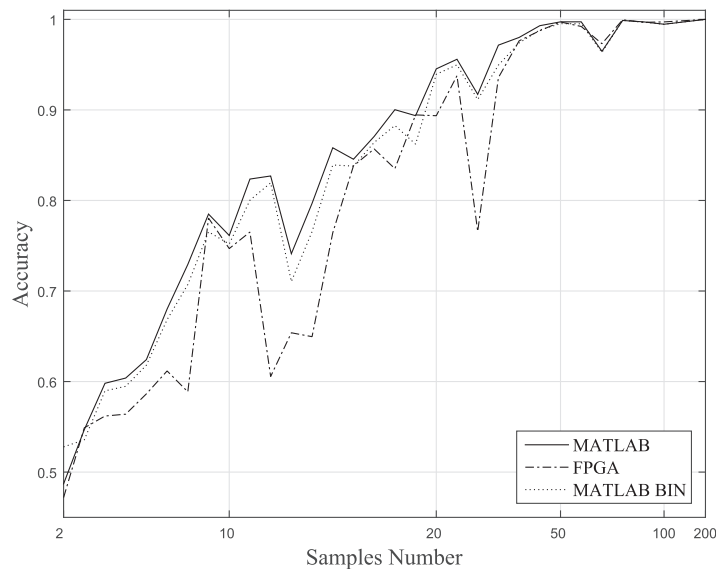


Fig. 11. Comparison accuracy results: BAL classification.

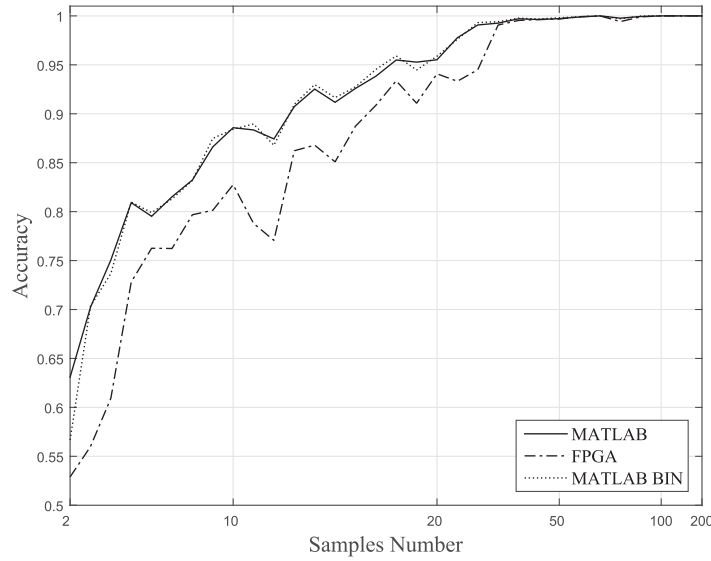


Fig. 12. Comparison accuracy results: BRN classification.

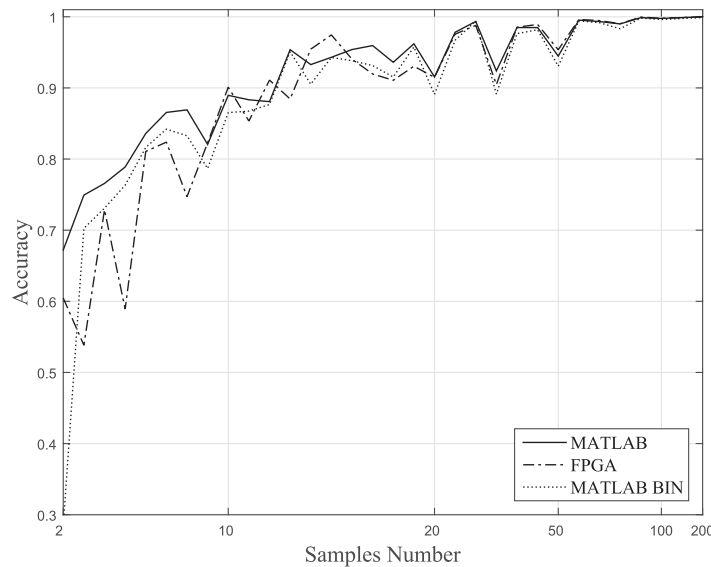


Fig. 13. Comparison accuracy results: HLT classification.

Table 1
Comparison with related works based on similar methodology.

Method	Classification	Classes	Samples	Accuracy	Recall	Specificity
CCA [18]	Hierarchical Neural Network	6	10 000	0.80–1.00	0.97	0.99
AAC [19]	Artificial Intelligence	3	100 000	0.71–1.00		
PCA & LDA [12]	Neural Network	5	270 000	0.90–0.92	0.92	0.98
SMOTE [20]	ADABOOST	5	800 000	0.80–1.00	0.93	
QSA	Decision Tree	3	2	0.53–0.74	0.58–0.74	0.73–0.85
			30	0.96–0.99	0.85–1.00	0.94–1.00
			200	0.99–1.00	0.98–1.00	0.97–1.00

The numerical comparison results of the QSA method implemented in FPGA are shown in Table 2. We want to point out that, our approach requires fewer samples than other methods to perform the classification. Our method uses two hundred samples to obtain 100% of test bench with a correct classification, while the SMOTE method uses eight hundred thousand samples and the CCA method uses ten thousand. The accuracy of QSA implemented

in FPGA improves more than other methods with low latency and processing time allowing on line processing.

The resource usage percentages of logic registers (*LR*), embedded multipliers elements (*EME*), combinational functions (*CF*), logic elements (*LE*) and memory bits (*MB*) are shown in Table 3. In particular, note that *MB* is the only resource that decreases significantly as the size of window sample increases. This is due to the

Table 2
Comparative results among related works and our strategy based on FPGA implementation.

Method	Classes	Samples	Processing time	Latency	Accuracy
CCA [18]	6	10 000	1 s	–	0.80–1.00
AAC [19]	3	100 000	5 s	–	0.71–1.00
PCA & LDA [12]	5	270 000	1 s	90 s	0.90–0.92
SMOTE [20]	5	800 000	–	10 s	0.8–1.0
QSA	2	2	4.23 μs	1.33 ms	0.47–0.60
(Our approach)	3	30	4.21 μs	0.02 s	0.76–0.98
		200	4.09 μs	0.133 s	1.00

Table 3
Consumed resource comparison table.

Samples	LR	EME	CF	LE	MB
2	3.29	27.63	8.62	11.90	2.27
10	3.29	27.63	8.73	12.01	0.94
25	3.29	30.26	8.80	12.08	0.15
50	3.29	30.26	8.80	12.08	0.01
100	3.29	30.26	8.79	12.07	0.01
200	3.29	30.26	8.79	12.07	0.01

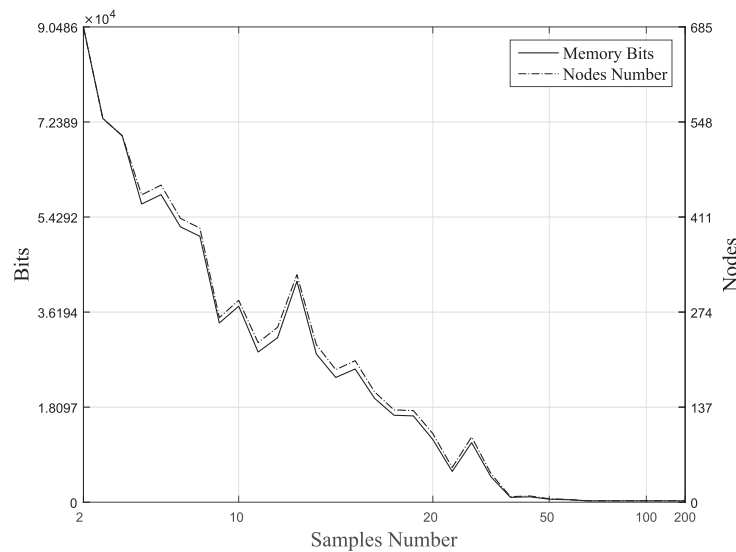


Fig. 14. Memory resources and number of decision tree nodes.

fact that the number of nodes of the decision trees also decreases as shown in Fig. 14 thus using more MB.

5. Conclusion

In this work we presented the implementation of the QSA method in FPGA to classify three possible states of an induction motor. Our architecture shows stable and low percentages of FPGA consumed resources. Therefore, the implementation in small-size systems to obtain a portable version is feasible. Similarly, low window samples provide low time of signals processing allowing online motor analysis with full accuracy. The proposed method can be adjusted to analyze more classes due to discriminant statistical features; in addition, the decision tree classifier shows a multi-class property. Future research should focus on increasing the number of the classes with high classification results, reduce the required samples to the algorithm and developing a portable system in FPGA which performs signal acquisition, tree training, statistical quaternion processing and motor state classification.

Acknowledgments

This work was partially supported by the project CIIC-3 funded by the Universidad de Guanajuato. Moreover, Jose-Luis Contreras-Hernandez is funded by the scholarship 487660 granted by the Consejo Nacional de Ciencia y Tecnología.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at <https://doi.org/10.1016/j.measurement.2019.01.088>.

References

[1] G. Wang, Y. Liu, T. Zhao, A quaternion-based switching filter for colour image denoising, *Signal Processing* 102 (2014) 216–225, <https://doi.org/10.1016/j.sigpro.2014.03.027>.
 [2] B. Czaplowski, M. Dzwonkowski, R. Rykaczewski, Digital fingerprinting based on quaternion encryption scheme for gray-tone images, *J. Telecommun. Inf. Technol.* 2 (07) (2014) 3–11.

- [3] P. Ginzberg, A.T. Walden, Matrix-valued and quaternion wavelets, *IEEE Trans. Signal Process.* 61 (6) (2013) 1357–1367, <https://doi.org/10.1109/TSP.2012.2235434>.
- [4] N. Le, S. Sangwine, T. Ell, Instantaneous frequency and amplitude of orthocomplex modulated signals based on quaternion fourier transform, *Signal Processing* 94 (2014) 308–318, <https://doi.org/10.1016/j.sigpro.2013.06.028>.
- [5] M. Pareniuk, S. Park, A versatile quaternion multiplier based on sparse-iteration 4d cordic, *New Circuits and Systems Conference (NEW-CAS)* 14. doi:10.1109/NEWCAS.2016.7604788.
- [6] D. Biswas, K. Maharatna, G. Panic, E.B. Mazomenos, J. Achner, J. Klemke, M. Jobges, S. Ortmann, Low-complexity framework for movement classification using body-worn sensors, *IEEE Trans. Very Large Scale Integration (VLSI) Syst.* 25 (4) (2017) 1537–1548, <https://doi.org/10.1109/tvlsi.2016.2641046>.
- [7] N. Petrovsky, A. Stankevich, A. Petrovsky, Low read-only memory distributed arithmetic implementation of quaternion multiplier using split matrix approach, *Electron. Lett.* 50 (24) (2014) 2–3, <https://doi.org/10.1049/el.2014.1775>.
- [8] B. Shivaprasad, K. Shinde, V. Muddi, Design and implementation of parallel floating point matrix multiplier for quaternion computation, *Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, 2015, pp. 289–293, <https://doi.org/10.1109/ICCICCT.2015.7475292>.
- [9] T. Ogunfunmi, T. Paul, An alternative kernel adaptive filtering algorithm for quaternion-valued data, *Signal & Information Processing Association Annual Summit and Conference (APSIPA ASC)*, 2012, pp. 1–5.
- [10] J.L. Barrera, F.E. Hernandez, Classification of mp-sk signals through eighth-order statistical signal processing, *IEEE Latin Am. Trans.* 15 (2017) 1601–1607, <https://doi.org/10.1109/TLA.2017.8015041>.
- [11] I.R.S. Gregori, I. Sanches, C.E. Thomaz, Clutch judder classification and prediction: a multivariate statistical analysis based on torque signals, *IEEE Trans. Industr. Electron.* 64 (2017) 4287–4295, <https://doi.org/10.1109/TIE.2016.2630666>.
- [12] J. Saucedo-Dorantes, M. Delgado-, R. Osornio-Rios, R.J. Romero Troncoso, Multi-fault diagnosis method applied to an electric machine based on high-dimensional feature reduction, *IEEE Trans. Ind. Appl.* 53 (2016) 3086–3097, <https://doi.org/10.1109/TIA.2016.2637307>.
- [13] M. Kang, J. Kim, J.M. Kim, An fpga-based multicore system for real-time bearing fault diagnosis using ultrasampling rate ae signals, *IEEE Trans. Industr. Electron.* 62 (4) (2015) 2319–2329, <https://doi.org/10.1109/tie.2014.2361317>.
- [14] Q. Wang, P. Li, Y. Kim, A parallel digital VLSI architecture for integrated support vector machine training and classification, *IEEE Trans. Very Large Scale Integr. VLSI Syst.* 23 (8) (2015) 1471–1484, <https://doi.org/10.1109/tvlsi.2014.2343231>.
- [15] P. Batres-Mendoza, C. Montoro-Sanjose, E.I. Guerra-Hernandez, D.L. Almanza-Ojeda, H. Rostro-Gonzalez, R.J. Romero-Troncoso, M.A. Ibarra-Manzano, Quaternion-based signal analysis for motor imagery classification from electroencephalographic signals, *Sensors* 16 (2016) 1–17, <https://doi.org/10.3390/s16030336>.
- [16] J.L. Contreras-Hernandez, D.L. Almanza-Ojeda, S.E. Ledesma-Orozco, A. Garcia-Perez, R.J. Romero-Troncoso, M.A. Ibarra-Manzano, Quaternion signal analysis algorithm for induction motor fault detection, *IEEE Trans. Industr. Electron.* (2019), <https://doi.org/10.1109/TIE.2019.2891468>, 1–1.
- [17] H.W. Gschwind, E.J. McCluskey, *Design of digital computers: an introduction, 2nd Edition., Monographs in Computer Science, Springer, Berlin Heidelberg*, 2012.
- [18] M.P. Delgado, G. Cirrincione, A. Garcia Espinosa, J.A. Ortega, H. Henao, Bearing fault detection by a novel condition-monitoring scheme based on statistical-time features and neural networks, *IEEE Trans. Ind. Electron.* 60 (8) (2013) 3398–3407, <https://doi.org/10.1109/TIE.2012.2219838>.
- [19] A. Soualhi, G. Clerc, H. Razik, Detection and diagnosis of faults in induction motor using an improved ant clustering technique, *IEEE Trans. Industr. Electron.* 60 (9) (2013) 4053–4062, <https://doi.org/10.1109/TIE.2012.2230598>.
- [20] I. Martin-Diaz, D. Morinigo-Sotelo, O. Duque-Perez, R.J. Romero-Troncoso, Early fault detection in induction motors using adaboost with imbalanced small data and optimized sampling, *IEEE Trans. Ind. Appl.* 9994 (c) (2016) 1, <https://doi.org/10.1109/TIA.2016.2618756>.
- [21] M. Zajac, M. Sulowicz, *Wavelet Detectors and for Extraction and of Characteristic and Features of Induction and Motor Rotor and Faults*, 2016 *International Conference on Signal and Electronic Systems (ICSSES)*, 2016, pp. 5–7.
- [22] A. Glowacz, Acoustic based fault diagnosis of three-phase induction motor, *Appl. Acoust.* 137 (2018) 82–89, <https://doi.org/10.1016/j.apacoust.2018.03.010>.
- [23] S. Prainetr, S. Wangnippanto, S. Tunyasirut, Detection mechanical and fault of induction and motor and using harmonic and current and sound and acoustic, *5th International Electrical Engineering Congress*, 2017, pp. 8–10, <https://doi.org/10.1109/IEECON.2017.8075725>.
- [24] A. Glowacz, Fault diagnosis of single-phase induction motor based on acoustic signals, *Mech. Syst. Signal Process.* 117 (2019) 65–80, <https://doi.org/10.1016/j.jymssp.2018.07.044>.
- [25] J.J. Saucedo-Dorantes, M. Delgado-Prieto, J. A Ortega-Redondo, R.A Osornio-Rios, R.J. Romero-Troncoso, Multiple-Fault Detection Methodology Based on Vibration and Current Analysis Applied to Bearings in Induction Motors and Gearboxes on the Kinematic Chain, *Shock and Vibration* 2016 (2016) 1–13. doi: 10.1155/2016/5467643.

Apéndice C

Artículo CONIMI

Análisis de señales para la detección de armónicos basado en el espacio complejo

Signal analysis to harmonics detection based in complex space

CONTRERAS-HERNÁNDEZ, Jose L. †*, ALMANZA-OJEDA, Dora L. y IBARRA-MANZANO Mario A.

Universidad de Guanajuato

ID 1^{er} Autor: *José, Contreras-Hernández* / ORC ID: 0000-0003-0405-5554 CVU: 487660

ID 1^{er} Coautor: *Dora, Almanza-Ojeda* / ORC ID: 0000-0002-3373-0929 CVU: 50006

ID 2^{do} Coautor: *Mario, Ibarra-Manzano* / ORC ID: 0000-0003-4317-0248 CVU: 105633

DOI: 10.35429/JCS.2019.7.3.1.4

Recibido: 12 de Enero, 2019; Aceptado 02 de Marzo, 2019

Resumen

Este trabajo presenta la detección de armónicos mediante el análisis de señales en espacio complejo. Se inicia obteniendo la división punto a punto de la muestra de interés con la muestra anterior de las figuras de Lissajous generadas por dos señales en el espacio complejo. Después, con la señal obtenida de la división, se calcula el ángulo de cada elemento para analizar su comportamiento. Las características conseguidas proporcionan información numérica de las frecuencias que componen una señal. Esto simplifica el análisis sin necesidad de transformaciones u operaciones de alto costo computacional.

Lissajous, Espacio complejo, Detección armónicos

Abstract

This work presents harmonic detection through signal analysis in complex space. This method obtains the point-to-point division of the present sample by the last sample of the Lissajous curve which is generated by two signals in complex space. Then, the angle is calculated to each element to obtain the behavior. The obtained features provide numerical information of the frequencies that form a signal. This is obtained without space transformation or operation with high computational cost.

Lissajous, Complex space, Harmonics detection

Citation: CONTRERAS-HERNANDEZ, Jose L. †*, ALMANZA-OJEDA, Dora L. y IBARRA-MANZANO Mario A. Análisis de señales para la detección de armónicos basado en el espacio complejo. Revista de Simulación Computacional. 2019 3-7: 1-4

* Correspondencia al autor: jose.contreras@ugto.mx

† Investigador contribuyendo como primer autor.

Introducción

Las señales armónicas se presentan en maquinaria de combustión interna, líneas eléctricas o instrumentos musicales y las cuales nos proporcionan características de los distintos sistemas que las producen [1]. Las frecuencias que componen este tipo de señales obtienen por medio del procesamiento y análisis profundo de atributos. Entre los métodos más utilizados para la detección de armónicos en señales se encuentra la Transformada Rápida de Fourier (FTT) [2], métodos de clasificación como las Máquinas de Vectores de Soporte (SVM) [3] o Redes neuronales (ANN) [4] y los métodos basados en Wavelets [5].

En trabajos como el mostrado en [1], los métodos basados en FFT realizan la aproximación de la reconstrucción de la señal a analizar por medio de frecuencias discretas, lo cual conlleva un cambio de espacio y cálculos con una mayor carga computacional. El método desarrollado por SVM utiliza funciones de ajuste para detectar los armónicos de las señales en sistemas eléctricos, sin embargo, el modelo y el pequeño muestreo llegan a afectar la precisión de sus componentes como se describe en [3]. De igual manera, las (ANN) son usadas para la detección de armónicos en redes eléctricas como lo describe [6], la cual presenta una mayor precisión con la dificultad de obtener una muestra ideal de entrenamiento, lo cual afecta la velocidad del método [7]. Como los autores describen en [5], los métodos basados en Wavelets para la detección de armónicos en señales de corriente han obtenido buenos resultados, en este trabajo, el esquema de elevación de daubechies 9/7 es aplicado para la detección de armónicos.

El trabajo aquí desarrollado propone el método de análisis geométrico. Éste se basa en la teoría de procesamiento algebraico de señales [8], en la cual la teoría algebraica hace la derivación de algoritmos de forma concisa y transparente, da una visión de su estructura, permite clasificar los existentes y descubrir nuevos para transformadas existentes y nuevas [9]. Se presenta el método para la obtención numérica de la cantidad de frecuencias que componen a una señal por medio de operaciones de bajo costo computacional en números complejos.

Este trabajo se divide en 3 secciones. En la Metodología se presenta el desarrollo matemático utilizado para la obtención de las componentes en frecuencia de una señal. En Resultados se presentan las señales que se probarán y el resultado del método. Finalmente, en la última sección se presentan las conclusiones de este trabajo y trabajo futuro.

Metodología

La figura de Lissajous es una herramienta que permite analizar e interpretar la interacción correspondiente a la superposición de dos señales con movimientos armónicos simples en direcciones perpendiculares. En este trabajo se presentan las señales seno y coseno descritas por las ecuaciones (1) y (2) de la forma

$$x(k) = \sin(kt) \quad (1)$$

$$y(k) = \cos(kt). \quad (2)$$

El comportamiento de las señales $x(k)$ y $y(k)$ es mostrado en la Figura 1a) con un periodo de 0 a 2π y amplitud unitaria. Estas señales pueden ser expresadas en espacio de los números complejos de la forma que se muestra en la ecuación siguiente

$$z(k) = x(k) + y(k)i. \quad (3)$$

El resultado de representar ambas señales en el espacio complejo corresponde a las figuras de Lissajous como se muestra en la Figura 1b). Partiendo de este tipo de representación, la apariencia de la figura creada por estas dos señales es sensible a los valores de frecuencia, amplitud, fase y offset, como se puede observar en la Figura 2, en la que se muestran las figuras de Lissajous resultantes al variar la amplitud de la señal en y , por lo que su análisis matemático permite determinar si alguna de sus características ha sido modificada [10].

Una opción en este tipo de análisis es la obtención de la componente z_c la cual, al ser multiplicada por $z(k)$, permite la obtención de $z(k+1)$. El valor $z_c(k)$ se consigue por medio de la división punto a punto de la parte real e imaginaria de $z(k+1)$ entre $z(k)$ señalada en la ecuación (4)

$$z_c(k) = \frac{\text{real}(z(k+1))}{\text{real}(z(k))} + \frac{\text{imag}(z(k+1))}{\text{imag}(z(k))} i \quad (4)$$

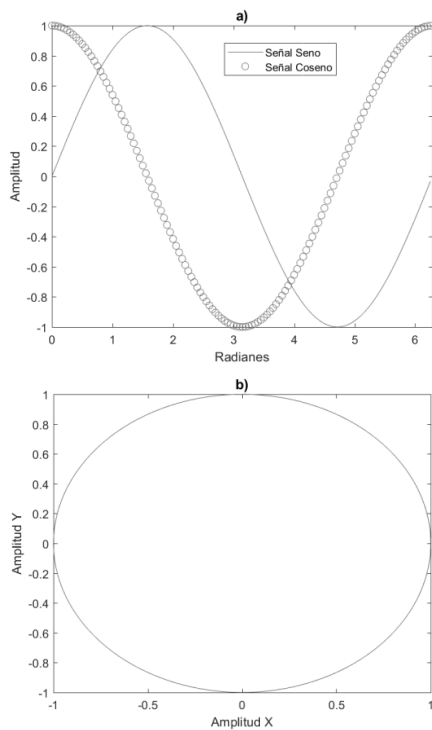


Figura 1 a) Señales x(k) y y(k). b) Gráfica de z(k)

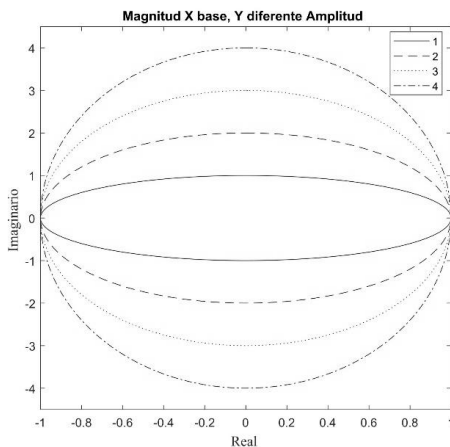


Figura 2 Gráficas de las figuras de Lissajous resultantes de la variación en la amplitud de la señal Y

Con la componente $z_c(k)$ calculada, se obtiene el ángulo $\theta_c(k)$ mediante

$$\theta_c(k) = \frac{\text{real}(z_c(k))}{\text{imag}(z_c(k))} \quad (5)$$

Analizando el comportamiento de este ángulo, se consiguen características de frecuencia en las señales, tal es el caso de los armónicos.

Con el fin de comprobar el método aquí presentado, a las señales originales se les aplica la suma de armónicos de acuerdo con las ecuaciones

$$x_a(k) = \sin(kt) + \sin(2kt) + \sin(3kt) \dots \sin(nkt) \quad (6)$$

$$y_a(k) = \cos(kt) + \cos(2kt) + \cos(3kt) \dots \cos(nkt) \quad (7)$$

Las pruebas realizadas constan de la suma de armónicos desde $n=1$ hasta $n=5$ para las señales X y Y. Las señales armónicas desarrolladas se muestran en la Figura 3. Así mismo, las figuras de Lissajous trazadas a partir de estas señales se muestran en la Figura 4.

Resultados

Después de obtener las señales con la suma de armónicos y las cuales generan una figura de Lissajous expresada en el espacio de números complejos, se calculan los ángulos de las componentes $z_c(k)$ de cada una, dando como resultado las gráficas mostradas en la Figura 5. En esta figura se muestra que existe una relación de la suma de armónicos en las señales con respecto a los picos que se presentan. Si se tiene una señal base sin armónico, el resultado es una recta. Si a esa señal se le suma su armónico, tanto al seno como en coseno, se presenta un pico en el ángulo formado en $z_c(k)$. Si a las señales a analizar se les suma su segundo armónico correspondiente, la gráfica resultante muestra dos picos y así sucesivamente. De esta manera se consiguen características de la suma de los armónicos correspondientes de dos señales calculando el ángulo de $z_c(k)$ de la figura de Lissajous representada en el espacio complejo.

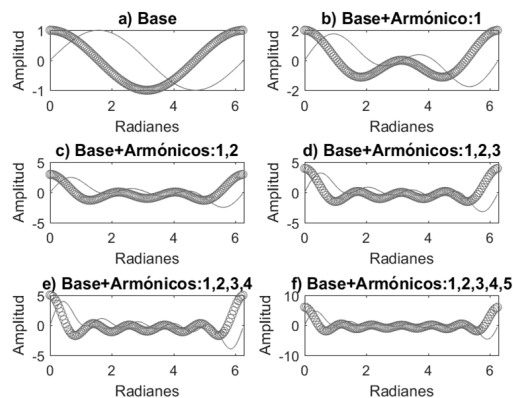


Figura 3 Gráficas de las señales resultantes de la suma de los armónicos de las señales seno (-) y coseno (o)

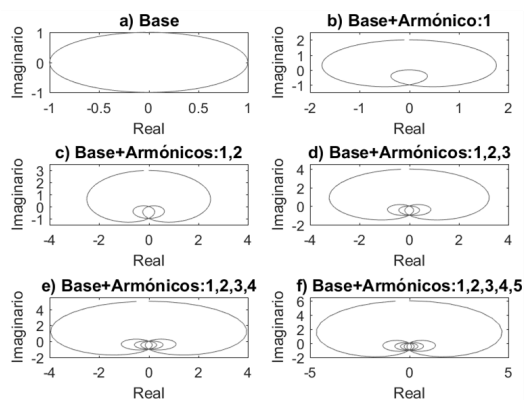


Figura 4 Gráficas de las figuras de Lissajous resultantes de la suma de los armónicos de las señales seno y coseno

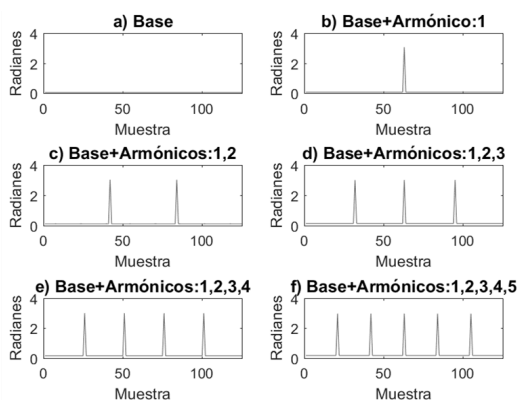


Figura 5 Gráficas de los ángulos calculados de las componentes de la suma de armónicos

Conclusiones

El análisis de las figuras de Lissajous en el espacio complejo es una herramienta muy útil. Mediante el cálculo del ángulo se extraen características de los armónicos de manera numérica representados por picos en las gráficas resultantes. El trabajo se desarrolló aplicando la suma de armónicos a las señales seno y coseno, por lo que es importante realizar pruebas tomando en cuenta las distintas variaciones y combinaciones que se tienen en las señales para extraer las características correspondientes de frecuencia, amplitud, fase y offset.

Referencias

[1] Piper, J. E. Reese, S. S. and Reese, W. G. (2012). Detection of harmonic signals. 2012 Oceans. Hampton Roads. 1-4.

[2] Zeytinoglu, M. and Wong, K. M. (Nov, 1995). Detection of harmonic sets. IEEE Trans on Signal Processing. (43). 2618-2630.

[3] Cao, J. Ling, T. Zhang, M. and Liu, L. (2008). Monitoring Method of Power System Interharmonics. High Voltage Engineering. (8). 1745-1750.

[4] Luo, H. and Luo, S. (Jan, 2011). The Research Summary of Harmonic Study In Electric Power System. China, Industrial Instrumentation and Automation. (5). 64-67.

[5] Wang, Y. Li, Z. and Guo, G. (2011). Research on methods of harmonic detection based on second generation wavelet algorithm. 2011 International Conference on Electric Information and Control Engineering. 6090-6093.

[6] Ma, X. Yin, Z. Zhou, L. and Zhu, Y. (2008). The research summary of harmonic detection in electric power system. China, Sichuan Electric Power Technology. (4). 53-56.

[7] Wei, W. Zengli, L. Lin, C. and Weiwei, S. (2013). Harmonic Detection of Power System Based on SVD and EMD. 2013 Ninth International Conference on Computational Intelligence and Security. 185-189.

[8] Fuhrmann, P. (2012) A polynomial approach to linear algebra. New York: Springer Press.

[9] Ran, T. Bing-zhao, L. and Hua-fei, S. (2013). Research Progress of The Algebraic and Geometric Signal Processing. (9). 40-47.

[10] Churchill, R.V. and Brown, J.W. (1992). Variable compleja y aplicaciones. McGraw Hill. Quinta Ed.

[11] French, A.P. (2003). Vibrations and Waves. CRC press. 198.

[12] Fuster R. and Giménez I. (2006). Variable compleja y ecuaciones diferenciales. Reverté.