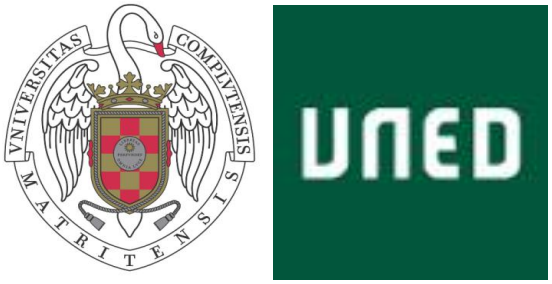
MÁSTER EN INGENIERÍA DE SISTEMAS Y CONTROL.

DETERMINACIÓN AUTOMÁTICA DE MODOS MULTIARMÓNICOS EN ESPECTROGRAMAS.



Departamento de Informática y Automática.

E.T.S.I.I Universidad Nacional de Educación a Distancia.

Alumno: Francisco José Martínez García.

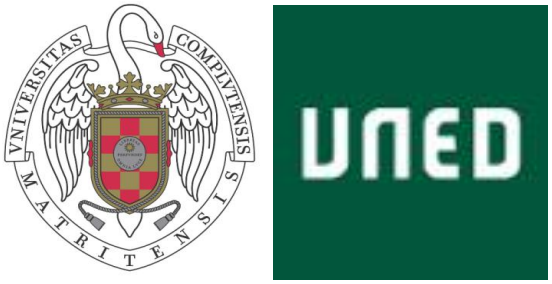
Dirigido por: Dr. D. Jesús Antonio Vega Sánchez.

Dr. D. Sebastián Dormido Canto.

Curso 2011-12. Convocatoria Septiembre.

MÁSTER EN INGENIERÍA DE SISTEMAS Y CONTROL.

DETERMINACIÓN AUTOMÁTICA DE MODOS MULTIARMÓNICOS EN ESPECTROGRAMAS.



Departamento de Informática y Automática.

E.T.S.I.I Universidad Nacional de Educación a Distancia.

Clase de Proyecto tipo A.

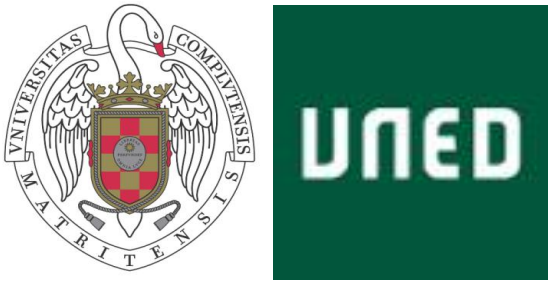
Alumno: Francisco José Martínez García.

Dirigido por: Dr. D. Jesús Antonio Vega Sánchez.

Dr. D. Sebastián Dormido Canto.

MÁSTER EN INGENIERÍA DE SISTEMAS Y CONTROL.

DETERMINACIÓN AUTOMÁTICA DE MODOS MULTIARMÓNICOS EN ESPECTROGRAMAS.



Departamento de Informática y Automática.

E.T.S.I.I Universidad Nacional de Educación a Distancia.

Tribunal evaluador:
Presidente: D./Da.……
(firma)

Vocal:D./Da.……
(firma)

Vocal: D./Da. ……
(firma)

Fecha de lectura y defensa: ……

Calificación: ……

Autorización

Autorizamos a la Universidad Complutense y a la UNED a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la memoria de este Trabajo Fin de Máster, como el código, la documentación y/o el prototipo desarrollado.

Firmado:

Francisco José Martínez García.

Resumen

Mediante el cálculo del espectrograma de una señal se puede determinar la presencia de ciertos fenómenos. A través de la segmentación de la imagen derivada del espectrograma es posible analizar estos modos mediante técnicas como son la extracción de bordes, la clasificación de texturas, y todo ello combinado con el análisis de regiones.

El objetivo de este trabajo se centra en la búsqueda de un algoritmo que sea capaz, a partir de datos extraídos de descargas de plasma nuclear, de automatizar la detección de estos modos. En este contexto, un modo viene determinado por una serie de armónicos acoplados a la señal asociada a una descarga. Una vez implementadas y analizadas diferentes alternativas, se diseñará un algoritmo para posteriormente codificarlo en una herramienta ad hoc que sea capaz de detectar automáticamente estos modos.

Palabras clave: Espectrograma, modo, regiones de imagen, plasma, operaciones morfológicas.

Agradecimientos

Quiero expresar mi agradecimiento a mis dos directores de Proyecto Dr. Jesús Vega Sánchez y Dr. Sebastián Dormido Canto por haberme dado esta oportunidad y sobre todo por su paciencia e interés. Sus consejos y orientación han sido decisivos para el desarrollo de este trabajo. Quiero agradecer también a Emilia Rodriguez-Solano el acceso a la base de datos de los modos externos. Asi mismo a Sergio Gonzalez por su ayuda en el desarrollo de algoritmos en el tratamiento de imágenes.

A Elisa, Gonzalo y Jaime quiero agradecerles todo su apoyo y comprensión por el tiempo que no he podido dedicarles. Espero que algún día entiendan el motivo por el que no les atendí como se merecen. Y en general, a todos aquellos que de una u otra forma han conseguido que este proyecto sea una realidad.

ÍNDICE

INDICE DE ILUSTRACIONES

1. INTRODUCCIÓN.

1.1. Identificación del problema.

La fusión nuclear por confinamiento magnético se perfila, junto con las energías renovables, como la fuente de energía del futuro.

A las temperaturas requeridas para que se produzcan reacciones de fusión, los electrones se encuentran completamente desligados de los núcleos, encontrándose la materia en estado de plasma. En el estudio del plasma por confinamiento magnético es crucial conocer con detalle las condiciones físicas del plasma confinado. En este sentido uno de los problemas de la fusión por confinamiento magnético es mantener el plasma confinado dentro del dispositivo de fusión sin que toque las paredes, lo que detendría las reacciones nucleares y dañaría el reactor. Las descargas de plasma nuclear generan fenómenos que deben ser estudiados con detalle. Por tanto, se hace necesario disponer de herramientas que ayuden a localizar dichos fenómenos y sobre todo que detecten dónde se encuentran.

Este trabajo se centra en uno de tantos fenómenos que generan las descargas de plasma nuclear. En particular, se trata de encontrar la localización de modos multiarmónicos acoplados a las señales que generan las descargas. Para conseguir el objetivo propuesto se calcula el espectrograma de la señal y se buscan en él los modos armónicos. Debido a la gran cantidad de señales disponibles se hace necesario crear un método automático que permita agilizar dicha tarea. Es aquí donde la visión por computador es de gran ayuda. El sistema de visión humana puede describir automáticamente un detalle, un borde, un color, diagnosticar enfermedades a partir una radiografía, etc. Pero cuando se trata de analizar de forma automática miles de fotografías y contrastar resultados, es cierto que el sistema de visión humana, no es el mejor método.

Esta memoria se organiza en cinco capítulos. En lo que resta del actual, se exponen los objetivos planteados, considerando unas hipótesis de partida y buscar dar solución al problema existente. En el capítulo dos, se hace un repaso a todos los métodos que se han utilizado hasta ahora para encontrar la solución. Los métodos descritos han sido utilizados y probados en el proyecto.

El capítulo tres, muestra los algoritmos que se desarrollaron así como la forma de implementar cada uno de los métodos descritos en el capitulo anterior.
El capítulo cuatro, muestra los resultados obtenidos que se comparan con los datos proporcionados por el experto.
En el capítulo cinco, se indican posibles mejoras así como los problemas detectados durante las pruebas y que deben ser tenidos en cuenta. Para terminar se presentan las conclusiones del trabajo realizado.
En el Anexo I, se detalla la utilización de la herramienta desarrollada y que implementa uno de los dos métodos investigados del capítulo tres. En el Anexo II, se incluye el código de la aplicación desarrollada.

1.2. Motivación y Objetivos.

Como se ha mencionado anteriormente, es necesario disponer de herramientas que ayuden a estudiar el comportamiento del plasma. Este proyecto nace de la necesidad de estudiar un fenómeno que se produce en las descargas de plasma. Se trata de localizar el instante temporal en el que aparecen los modos armónicos, así como la duración de los mismos. Éstos armónicos suelen aparecer en una banda frecuencial concreta, pero el objetivo es detectarlos en todo el periodo temporal de la descarga y en toda su banda frecuencial.

Para conseguir este objetivo se deben analizar ficheros que contienen las medidas de estas descargas. Las medidas que aquí se presentan se han realizado en el *stellarator* TJ-II, del Laboratorio Nacional de Fusión por Confinamiento Magnético, perteneciente al Centro de Investigaciones Energéticas Medioambientales y Tecnológicas (en adelante, **CIEMAT**).

A partir de estos ficheros se debe calcular el espectrograma de la imagen para posteriormente estudiar la aparición de armónicos.

A continuación se muestran los objetivos específicos del trabajo:

1) Aprender diferentes técnicas de visión por computador.
2) Analizar el problema y ver cuáles de los diferentes métodos anteriores o combinación de los mismos podrían aplicarse a la resolución del problema para la detección de armónicos en imágenes de espectrogramas.
3) Analizar las ventajas e inconvenientes de cada uno así como su problemática.

4) Identificar y diseñar un algoritmo mediante el cual se pueda dar una solución al problema lo más precisa posible.
5) Diseñar e implementar una herramienta que, de forma automática, sea capaz de analizar gran número de datos.

1.3. Metodología cronológica.

A continuación se exponen de forma cronológica los pasos que se han seguido para llevar a cabo la investigación junto con la actividad desarrollada en cada uno de ellos.

El proyecto se inicia en el marco apuntado previamente, donde el cálculo del espectrograma y búsqueda de armónicos aparece como un reto para dar respuesta a una necesidad. Esto resulta ser más complicado dado que este fenómeno no sigue una pauta común de comportamiento.

Una vez definidos y ajustados los parámetros que determina el espectrograma, el estudio se centra en la detección concreta del fenómeno. Se utilizan diferentes técnicas, se diseña un primer algoritmo y se consiguen los primeros resultados. La investigación se centra en la aplicación de diferentes filtros a la imagen binarizada del espectrograma para posteriormente aplicar la transformada de Hough. Este método consigue unos primeros resultados en base a trazar rectas que pasen por los picos de energía que forman los armónicos. Esto es posible dado que los armónicos detectados de forma manual tienen, en algún caso, gráficamente forma de recta. A pesar de conseguir buenos resultados el sistema resulta ser lento. La lentitud se debe fundamentalmente al cálculo de los espectrogramas y al tratamiento de gran número de imágenes. También aquí aparece el problema del rendimiento del sistema, por lo que es necesario recurrir a herramientas de medición como puede ser "*process explorer"* para detectar donde puede estar el foco del mal funcionamiento.

Para minimizar el impacto del tiempo se recurre a analizar estadísticamente las señales que presentan mejores resultados. Con este planteamiento se establece lo que posteriormente se nombrará como *criterio de análisis* y se implementará en la herramienta desarrollada. Un *criterio de análisis* de descargas se basa en analizar solamente las señales junto con sus canales que presentan más calidad. Una vez implementado este criterio los tiempos mejoran y las estadísticas de éxito también. Llegados a este punto, se estudia la posibilidad de

paralelizar las tareas. Se realizan pruebas con un modulo de Matlab y que consta de dos servidores pero esta línea se abandona para una ampliación futura del proyecto.

El último intento de mejorar los datos obtenidos ha consistido en investigar algunas de las medidas estadísticas de la imagen como puede ser métodos de clasificación estadística concretamente el *agrupamiento borroso*. Pero los resultados no llegan a ser adecuados para seguir investigando en esa dirección. En este punto se decide estudiar las imágenes y sus propiedades. Se centra el estudio en el análisis de las regiones de imagen. Se realizan diferentes pruebas y los resultados son mejores que el método anterior, por lo que se decide desarrollar una herramienta que implemente este algoritmo para que, de forma automática, detecte el fenómeno de modos multiarmónicos en todas las descargas.

2. REVISIÓN DE MÉTODOS.

2.1. Introducción.

En este capítulo se hace una revisión de las técnicas que posterioremente se aplicarán para tratar de conseguir el objetivo planteado. En primera instancia, se revisará el cálculo y realización del espectrograma de una señal. Se comprobará cómo la modificación de ciertas características puede hacer variar los resultados. También en dicho apartado se verán los resultados de las diferentes pruebas realizadas y los parámetros que al final son los que se han considerado óptimos para las fases posteriores. A continuación se entrará de lleno en el aspecto visual del proyecto; es decir, el tratamiento de imágenes.

La revisión comienza con el cálculo de transformada de Hough y su interpretación geométrica. Posteriormente, se da un repaso a los diferentes filtros utilizados en los tratamientos de imágenes, para terminar con las regiones de imágenes. Se denomina regiones de imágenes a toda área incluida en una imagen que resulta de interés. Se verán las propiedades que definen las regiones, haciendo hincapié en las que han sido aplicadas al proyecto.

Los armónicos son los componentes de una señal y se definen como las frecuencias secundarias que acompañan a una frecuencia fundamental o generadora. El armónico de una onda es un componente sinusoidal de una señal y su frecuencia es un múltiplo de la fundamental. La amplitud de los armónicos más altos es mucho menor que la amplitud de la onda fundamental y tiende a cero. La presencia de las señales armónicas, ocasiona que se produzcan interferencias en las señales y por ejemplo incremento de la temperatura

2.2. Espectrograma.

El análisis espectral es crucial para comprender las características de la señal, y puede aplicarse a cualquier tipo de señal, incluidas señales de radar, señales de audio, datos sísmicos, datos de valores financieros y señales biomédicas. Un espectrograma es la representación de la magnitud de la transformada de Fourier de una señal.

El espectro de una señal es la medida de la distribución de amplitudes de cada frecuencia. Es decir, representa cada frecuencia contenida en una señal y su intensidad. Matemáticamente, el análisis espectral está relacionado con una herramienta llamada transformada de Fourier o análisis de Fourier. Ese análisis puede llevarse a cabo para pequeños intervalos de tiempo, o menos frecuentemente para intervalos largos, o incluso puede realizarse el análisis espectral de una función determinista.

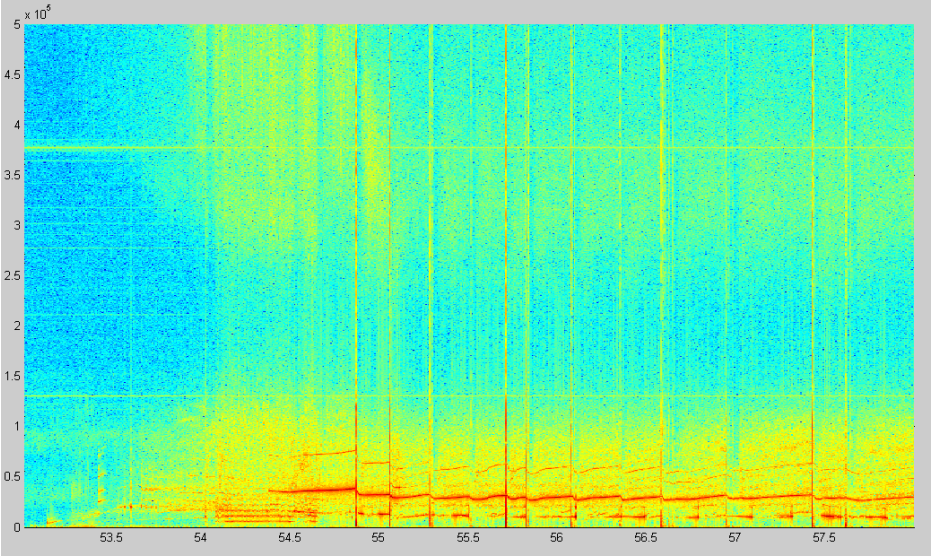


Figura 1: Espectrograma de la descarga 79455, señal PP8 canal 6.

Además la transformada de Fourier de una función, no sólo permite hacer una descomposición espectral de los formantes de una onda o señal oscilatoria sino que, con el espectro generado por el análisis de Fourier, incluso se puede reconstruir (*sintetizar*) la función original mediante la transformada inversa. Para poder hacer eso, la transformada no solamente contiene información sobre la intensidad de determinada frecuencia, sino también sobre su fase.

Esta información se puede representar como un vector bidimensional o como un número complejo. En las representaciones gráficas, frecuentemente sólo se representa el módulo al cuadrado de ese número, y el gráfico resultante se conoce como **espectro de potencia** o **densidad espectral de potencia**. La figura 1 muestra un ejemplo de espectrograma de una señal. Se puede comprobar la distribución de potencia en diferentes frecuencias.

Es importante recordar que la transformada de Fourier de una onda aleatoria, mejor dicho estocástica, es también aleatoria. Un ejemplo de este tipo de onda es el ruido ambiental. Por

tanto, para representar una onda de ese tipo se requiere cierto tipo de promediado para representar adecuadamente la distribución frecuencial.

La **transformada discreta de Fourier** o **DFT** (del inglés, *discrete Fourier transform*) es un tipo de transformada discreta utilizada en el análisis de Fourier. Transforma una función matemática en otra, obteniendo una representación en el dominio de la frecuencia, siendo la función original una función en el dominio del tiempo. Pero la DFT requiere que la función de entrada sea una secuencia discreta y de duración finita. Dichas secuencias se suelen generar a partir del muestreo de una función continua, como puede ser la voz humana. Al contrario que la transformada de Fourier en tiempo discreto (DTFT), esta transformación únicamente evalúa suficientes componentes frecuenciales para reconstruir el segmento finito que se analiza.

Utilizar la DFT implica que el segmento que se analiza es un único período de una señal periódica que se extiende de forma infinita; si esto no se cumple, se debe utilizar una **ventana** para reducir los espúreos del espectro. Por la misma razón, la DFT inversa (IDFT) no puede reproducir el dominio del tiempo completo, a no ser que la entrada sea periódica indefinidamente. Por estas razones, se dice que la DFT es una transformada de Fourier para análisis de señales de tiempo discreto y dominio finito. Las funciones sinusoidales base que surgen de la descomposición tienen las mismas propiedades.

La entrada de la DFT es una secuencia finita de números reales o complejos, de modo que es ideal para procesar información almacenada en soportes digitales. En particular, la DFT se utiliza comúnmente en procesado digital de señales y otros campos relacionados dedicados a analizar las frecuencias que contiene una señal muestreada, también para resolver ecuaciones diferenciales parciales, y para llevar a cabo operaciones como convoluciones o multiplicaciones de enteros largos. Un factor muy importante para este tipo de aplicaciones es que la DFT puede ser calculada de forma eficiente en la práctica utilizando el algoritmo de la transformada rápida de Fourier o FFT (Fast Fourier Transform).

Los algoritmos FFT se utilizan tan habitualmente para calcular DFTs que el término "FFT" muchas veces se utiliza en lugar de "DFT" en lenguaje coloquial. Formalmente, hay una diferencia clara: "DFT" hace alusión a una transformación o función matemática,

independientemente de cómo se calcule, mientras que "FFT" se refiere a una familia específica de algoritmos para calcular DFTs.

A nivel teórico se puede decir que los armónicos son los componentes de una señal que se definen como las frecuencias secundarias que acompañan a una frecuencia fundamental o generadora.

El armónico de una onda es un componente sinusoidal de una señal y su frecuencia es un múltiplo de la fundamental. La amplitud de los armónicos más altos es mucho menor que la amplitud de la onda fundamental y tiende a cero. A modo de ejemplo en audio los armónicos por encima del quinto o sexto generalmente son inaudibles.

La presencia de las señales armónicas, ocasiona que se produzcan: interferencia en las señales, incremento de la temperatura. La figura 1 muestra el espectrograma de la descarga número 79455, señal PP8 y canal 6. En el intervalo temporal comprendido entre 54 y 54.5 sg., se puede comprobar la presencia de armónicos. Se trata de las líneas paralelas que están en el intervalo frecuencial entre 5 KHz y 20 Khz.

2.3. Técnicas de preprocesado.

Las técnicas de procesado pretenden mejorar o realzar las propiedades de la imagen para facilitar las posteriores operaciones de la Visión Artificial, tales como las etapas de segmentación, extracción de las características y finalmente la interpretación automática de las imágenes. Todo esto lleva a que el fin último de la Visión Artificial es la interpretación automática de la imagen o una mejora de la calidad de la imagen.

Las técnicas de preprocesado se basan bien en técnicas derivadas del procesamiento digital de señales, o bien en un conjunto de procedimientos heurísticos que han dado resultados satisfactorios. Estos algoritmos se pueden catalogar en función de las pretensiones de sus transformaciones en alguna de las siguientes facetas:

- Realce o aumento del contraste (enhancement).
- Suavizado o eliminación del ruido (denoising)
- Detección de bordes (edge detection)

La técnica a aplicar de ahora en adelante será la del suavizado, como parte de los métodos desarrollados en la consecución del objetivo.

2.3.1. Suavizado.

Las técnicas de suavizado de las imágenes intentan eliminar el nivel de ruido presente en la imagen. En el proceso de formación de la imagen se ha generado ruido que se ha sumado a la señal. Este ruido se explicará como una variable aleatoria que sigue una función de densidad determinada. El origen del ruido es múltiple y existen tres tipos básicos: Gaussiano, impulsional y multiplicativo.

El ruido más común se origina en el proceso de captación al convertir los fotones en carga eléctrica, debido al carácter no determinista de esta transformación.

Este tipo de ruido se explica mediante el modelo de función de densidad normal o Gaussiana. El segundo tipo de perturbación se consigue como consecuencia de la saturación de carga que recibe un píxel ya sea por exceso o por defecto. A este tipo de ruido se le denomina de '*sal y pimienta'*. La saturación se produce debido a la sensibilidad de las cámaras al infrarrojo y al encontrar sobre el escenario objetos que están calientes. Este segundo tipo de comportamiento se modela con una función de densidad de tipo impulsional. Por último, hay ruido generado por la falta de iluminación uniforme sobre la escena capturada. Este error es como si la imagen estuviera formada por la multiplicación de dos imágenes, por un lado la imagen que refleja la intensidad de iluminación sobre los objetos y de otro la función de refracción de los cuerpos vistos en la escena. A esta categoría de ruido se le ha relacionado con los filtros homomórficos.

2.3.2. Suavizado binario.

Para eliminar el ruido básicamente consiste en evaluar una función booleana específica sobre un entorno de vecindad centrado sobre un pixel *p*, y dependiendo de la configuración espacial y los valores binarios de sus vecinos, asignarle a *p* un ´0´o bien un ´1´. La máscara que se propone de forma general tiene dimensiones de *3x3***.** Para resumir, el proceso de suavizado tiene los siguientes efectos: 1) rellena los pixel huecos en zonas oscuras. 2) rellena las pequeñas muescas de cortes en segmentos rectos. 3) elimina los unos aislados. 4) elimina las protuberancias en segmentos de lados rectos. 5) repone los puntos perdidos de las esquinas**.**El algoritmo de suavizado utilizado en el proyecto es el de Nagao y Matsuyana (1980). Las figuras 2 y 3 muestran el resultado.

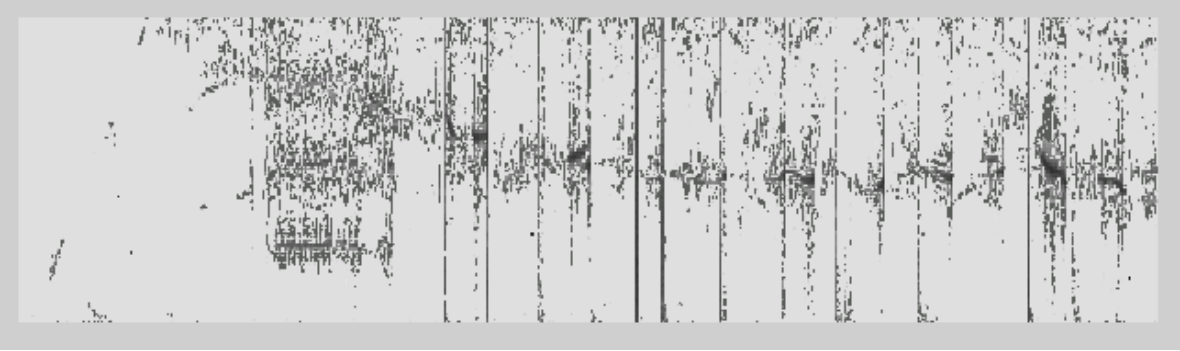


Figura 2: Imagen original donde se aprecia la gran cantidad de ruido

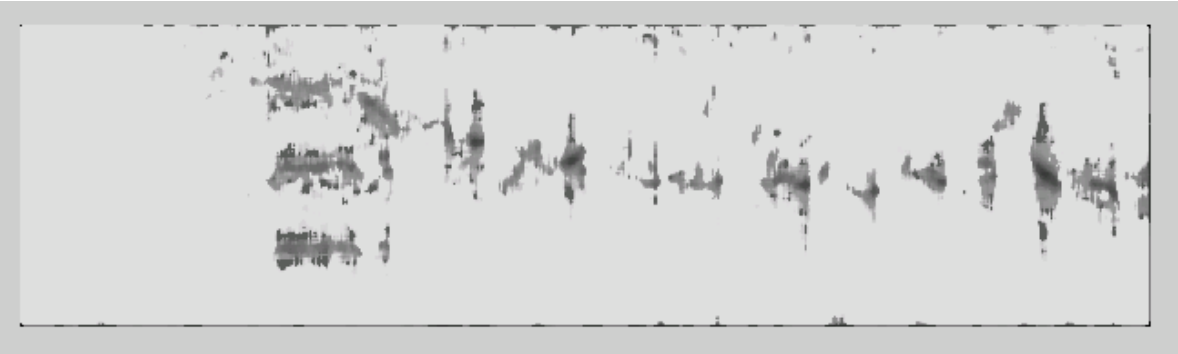


Figura 3: Imagen con filtro de la mediana aplicado con máscara de 8x8.

El algoritmo de Nagao y Matsuyama trata de preservar los bordes aunque se suavice la imagen. El objetivo es resolver el conflicto entre la eliminación del ruido y la degradación de los bordes. El algoritmo sigue los siguientes pasos:

1) Rotar una máscara rectangular alrededor de un punto (x,y).
2) Detectar la posición de la máscara donde su varianza del nivel de gris es mínima.
3) Asignar el valor medio del nivel de gris de la máscara a la posición seleccionada del punto (x,y).
4) Aplicar los pasos 1-3 a todos los puntos de la imagen.
5) Iterar el proceso anterior hasta que los niveles de gris de casi todos los puntos en la imagen no cambien o cuando se haya realizado un cierto número de iteraciones.

Para eliminar el ruido sin desdibujar los bordes, el promediado no debe aplicarse a un área que contenga un borde ya que esto tiende a desdibujarlo. Por tanto, se trata de encontrar la vecindad más homogénea alrededor de un punto para ser suavizado. Si un área contiene un

borde, la varianza del nivel de gris en dicha área se hace grande. Por tanto, se puede usar la varianza como una medida de homogeneidad de un área.

La mediana *M* de un conjunto de valores es tal que la mitad de los valores del conjunto son mayores que *M*. Con el objeto de realizar un filtrado de la mediana en el entorno de vecindad, seordenan las intensidades de la vecindad. La forma de ordenarlas es la siguiente; se toman todos los pixeles de la vecindad identificado como *ventana* y se reemplaza el valor del pixel central. Se organizan todos los pixeles de forma secuencial basándose en el valor del nivel de gris de los mismos. Los pixeles se ordenan de forma secuencial de menor a mayor valor. Por ejemplo dada una ventana *V* de dimensión *nxn* los valores de los pixeles se ordenarían de la siguiente forma:

$$f_1 \leq f_2 \leq f_3 \leq f_4 \leq f_5 \ldots\ldots f_n^2$$

Según este planteamiento y una vez ordenados todos los valores de los pixeles se determina la mediana y se asigna a la intensidad del pixel.

Por ejemplo, en un entorno de vecindad de *3x3* la mediana es el quinto valor más grande, en un entorno de *5x5* la mediana es el décimo tercer valor más grande, y así sucesivamente. Cuando varios pixeles en un entorno de vecindad tienen el mismo valor, se agrupan de la siguiente forma: por ejemplo se supone que un entorno de vecindad de *3x3* tiene los valores (20, 30, 10, 50, 50, 60, 50, 50, 30). Estos valores se ordenan como sigue (10, 20, 30, 30, 50, 50, 50, 60) obteniéndose el valor de 50 en la mediana. El objetivo del filtro de la mediana es hacer que los puntos con intensidades muy distintas se hagan muy parecidos a sus vecinos, eliminando así los picos de intensidad que aparezcan aislados en un área de la máscara del filtro.

Las figuras 2 y 3 muestran los efectos de la aplicación del filtro de la mediana. El objetivo es definir lo más posible las regiones objeto de estudio. Esto último se consigue con la aplicación de una máscara de *8x8*. En la figura 3 se aprecia cómo las zonas que, posteriormente se estudiarán, están mucho más definidas.

2.4. Transformada de Hough.

La transformada de Hough permite encontrar las formas básicas que definen una imagen a la que previamente se le ha aplicado algún detector de bordes. Estas formas pueden ser representadas mediante rectas, círculos o curvas.

Para un punto (x_i , y_i) se puede generar la ecuación de la recta que lo contiene:

$$y_i = a \cdot x_i + b \qquad [1]$$

Existen infinitas rectas que cumplen la ecuación [1] para diversos factores de *a* (pendiente) y *b* (ordenada al origen).

Examinando el espacio paramétrico de a y b (plano ab) se puede observar que un punto (a_i , b_i) representa una determinada recta en el plano xy:

$$b = - x_i \cdot a + y_i \qquad [2]$$

Por lo que para cada punto (x_i , y_i) perteneciente a una recta, se obtiene otra recta en el espacio paramétrico ab. Todas estas rectas se intersectan en un punto (a',b'). Dicho punto es la representación de la recta en el plano xy que contiene todos los puntos (x_i , y_i), colineales (que tienen pendiente a´y ordenada al origen b´).

El problema que surge de esta transformación es para aquellos casos donde la recta en el plano ab es vertical, puesto que la pendiente de la recta es infinita.

Para solucionar este problema Hough propuso un método alternativo que consiste en expresar la ecuación de la recta en coordenadas polares. Ésta es la llamada *Transformada de Hough.*

2.4.1. Método de transformación.

La ecuación básica para la transformada de Hough fue diseñada para detectar líneas rectas y curvas. Esto significa que cualquier línea recta en el espacio de la imagen *xy* es representada por **un solo punto** en el espacio de parámetros $\rho\Theta$, y cualquier parte de esta línea recta es transformada en el mismo punto.

$$\rho = x_i \cdot cos(\Theta) + y_i \cdot sen(\Theta)$$

El rango del ángulo Θ varia de $-\pi/2$ a $\pi/2$ medido con respecto al eje x. Y el rango de ρ varía de $-N\sqrt{2}$ a $N\sqrt{2}$ donde N es la dimensión de la imagen cuadrada. El método de Hough subdivide el espacio de parámetros en las denominadas *células acumulador*. La célula de coordenadas i, con valor acumulador A_{ij} corresponde al cuadrado asociado con las coordenadas

2.4.2. Detección de líneas.

Para este caso concreto y una vez obtenida la matriz binaria (BW) del espectrograma el objetivo es aplicar la transformada de Hough. De esta forma se está en condiciones de detectar las rectas principales que pasan por las líneas definidas por los amónicos. Posteriormente habrá que analizar estas rectas para ver si cumplen las condiciones de multiplicidad frecuencial.

A continuación, se explicará de forma analítica los pasos que sigue la Transformada de Hough mediante el siguiente ejemplo:

Supóngase la siguiente **matriz binaria (BW)** de la Figura 4, donde los píxeles de interés se encuentran en las siguientes coordenadas (X,Y): (5,1), (6,2), (7,3), (8,4), (9,5).

	1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	1	0
2	0	0	0	0	0	0	1	0	0
3	0	0	0	0	0	1	0	0	0
4	0	0	0	0	1	0	0	0	0
5	0	0	0	1	0	0	0	0	0
6	0	0	1	0	0	0	0	0	0
7	0	1	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0

Figura 4: Matriz binaria (BW).

Para representar la línea que definen estos píxeles de valor 1, se considera la ecuación de una recta a partir de los parámetros de y en coordenadas polares:

$$y = \left[\frac{-\cos\theta}{sen\ \theta}\right]x + \frac{\rho}{sen\ \theta}$$

Simplificando la ecuación se tendrá que;

$$\rho(\theta) = x \cdot sen\ \theta + y \cdot sen\ \theta$$

Dónde $\theta \epsilon [-90,90]$ grados.

Tal y como se muestra en la siguiente figura 5, se corresponde con la dirección (medida con respecto al eje x) en la que se evaluará cada uno de los píxeles de interés (para cada píxel se trazará una recta que corte perpendicularmente con esa dirección) y con el módulo del vector o de cada uno de los vectores que se formen en esa dirección (distancia desde el origen de coordenadas a la recta que atraviesa el píxel de interés).

Un dato importante a tener en cuenta sería la ubicación de este origen de coordenadas, ya que dependiendo de donde se sitúe, se obtendrán diferentes resultados para los valores de *y*.

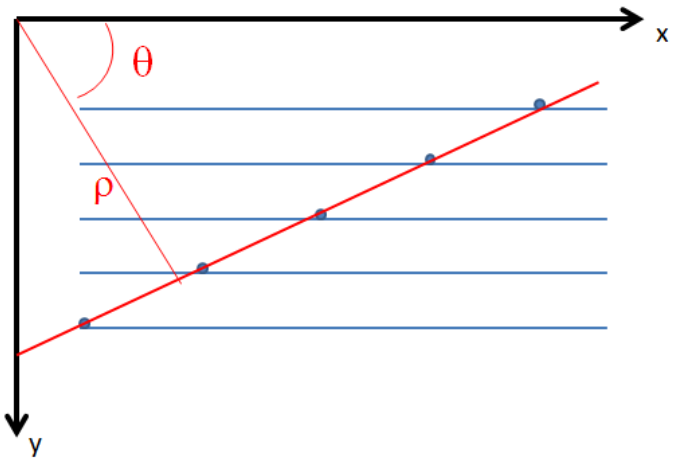


Figura 5: Representación de una recta.

En este ejemplo, se considera que el origen se encuentra en la parte superior derecha de la matriz binaria (BW).

Partiendo de la ecuación de la recta en coordenadas polares ($\rho(\Theta) = x \cdot cos\Theta + y \cdot sin\ \Theta$), para cada Θ se calcula la ρ correspondiente pero de solamente los pixeles de interés (los pixeles de interés son los que están a "1" en la matriz binaria BW) y se almacenan los resultados en un vector (figura 6).

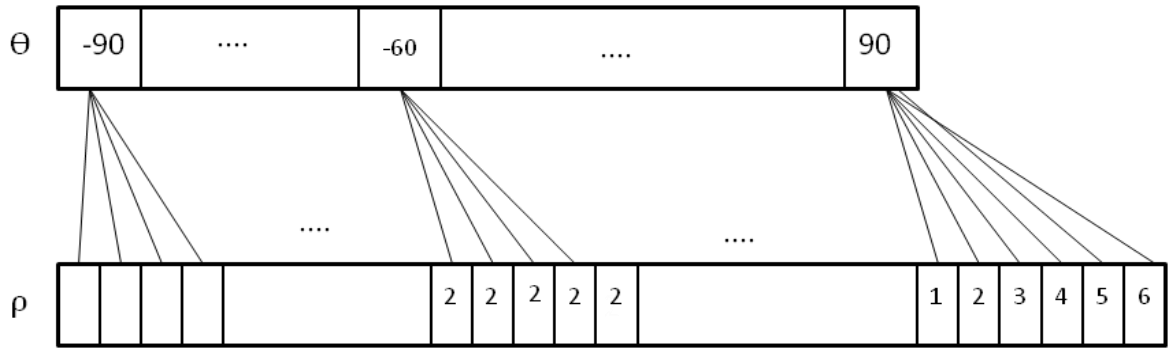


Figura 6: Vector que almacena ρ de los pixel de interés.

En este ejemplo se ve que la matriz binaria solo consta de 5 pixeles con valor '1' por lo tanto, para cada ángulo Θ solo obtendrán 5 posibles valores de ρ, se observa que para Θ=-60 grados, la ρ es la misma para los cinco pixeles; es decir los cinco pixeles están alineados y por lo tanto eso significa que se ha encontrado una línea principal. A diferencia de eso, para Θ=90 grados no hay ningún valor de ρ igual y, por lo tanto se deduce que los pixeles de interés no se encuentran alineados horizontalmente.

A partir de los vectores Θ y ρ de la figura anterior, se rellena la **matriz H** (matriz de Hough) que devuelve la función de Matlab *hough.m* (figura 7).

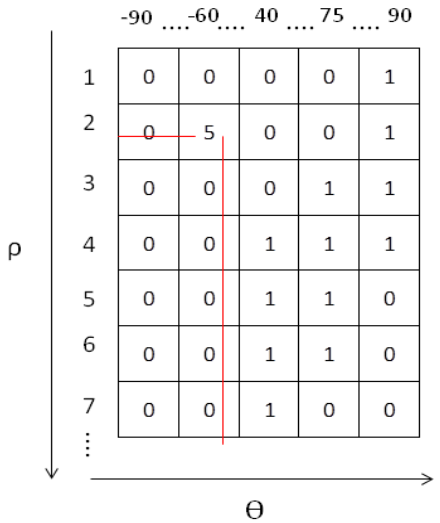


Figura 7: Matriz de Hough.

El valor de cada una de estas posiciones de esta matriz se corresponde con el número de pixeles que comparten un determinado valor de ρ, alineados en la dirección perpendicular a Θ. Por ejemplo, para el caso anterior, se obtendría el siguiente ejemplo.

La línea definida por Θ=-60 grados y ρ=2 atraviesa cinco pixeles de interés. El resto de líneas posibles tan solo atraviesan 1 pixel o ninguno. En general, los valores máximos de la matriz H se corresponden con las líneas detectadas. En este ejemplo se puede afirmar que se ha detectado una única línea.

Para terminar de ilustrar la técnica de Hough para la detección de líneas, se presenta la siguiente imagen binaria de dimensiones conocidas:

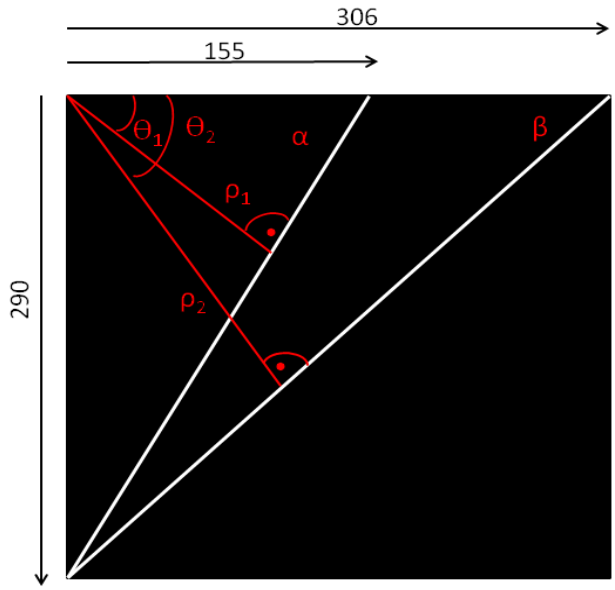


Figura 8: Rectas a analizar mediante transformada de Hough.

En primer lugar se realizan los cálculos analíticos reales de cada Θ y ρ de cada una de las rectas de la figura 8 con el objetivo de verificar que los resultados que se obtenga al aplicar la Transformada de Hough sean los correctos. Los cálculos son:

Cálculo para Θ_1, α_1 y ρ_1

$$\alpha_1=\arctan\frac{290}{155}=61.8; \qquad \Theta_1=90-\alpha_1=28.1; \qquad \rho_1=155\cdot\text{sen}\,\alpha_1=136.6$$

Cálculo para Θ_2, α_2 y ρ_2:

$$\alpha_2=\arctan\frac{290}{306}=43.46; \qquad \Theta_2=90-\alpha_2=46.5; \qquad \rho_1=306\cdot\text{sen}\,\alpha_2=210.4;$$

Una vez realizados los cálculos trigonométricos se aplica la transformada de Hough a la imagen binaria y se representa mediante MatLab la matriz de Hough. Los resultados se muestran en la figura 9.

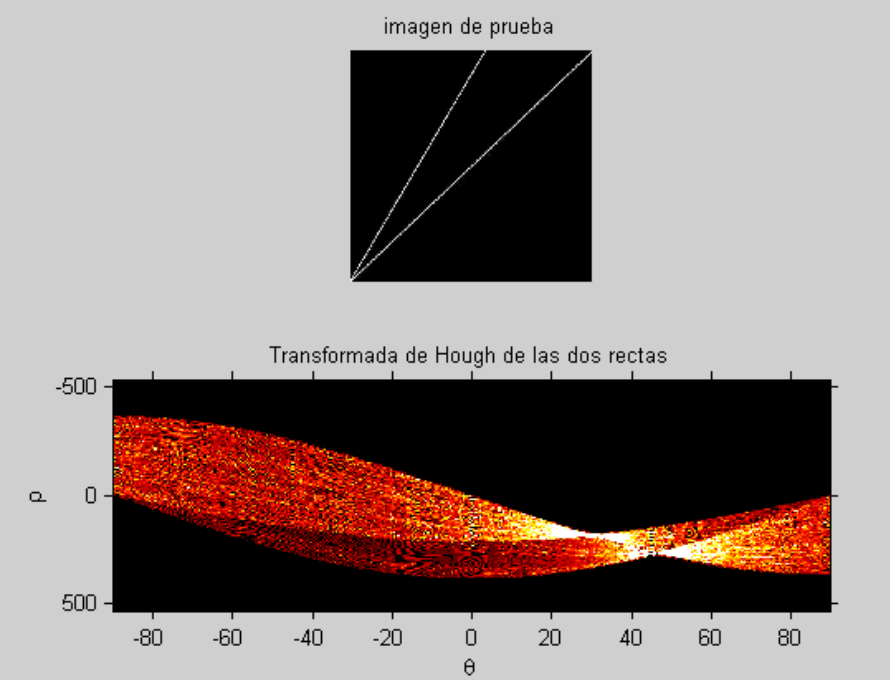


Figura 9: Resultado de Matlab de la matriz Hough rectas de la figura 8.

En la imagen inferior de la figura 9 se puede comprobar dos máximos encerrados en dos círculos azul que corresponden a las dos rectas. Aproximando la imagen para ver más de cerca estos máximos, se puede comprobar las magnitudes de Θ y ρ, que se expone en la figura 10.

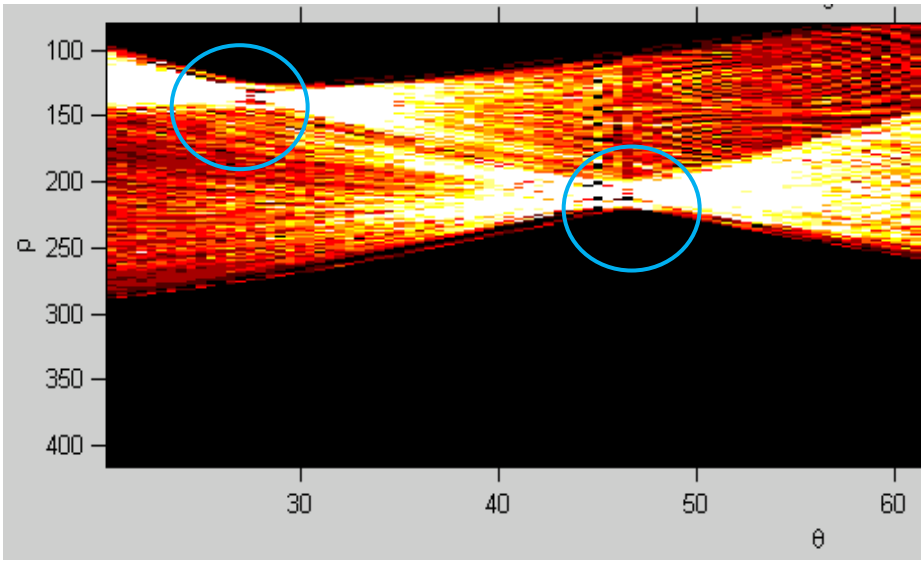


Figura 10: Zoom del resultado de Matlab de la matriz Hough rectas figura 8.

En la figura 10 se puede comprobar cómo los valores calculados para Θ_1, α_1, ρ_1, Θ_2, α_2 y ρ_2 se aproximan mucho al resultado gráfico. Puede haber alguna desviación en la precisión en la realización del trazado de las dos líneas puesto que se realizaron de forma manual.

Tal como se ha comentado anteriormente, los valores máximos de la Matriz H indican que en esas posiciones de han detectado líneas principales, cuyo número de pixeles es igual al valor encontrado en dicha posición. En la representación gráfica adquieren una elevada intensidad debido a la concentración de pixeles en esas coordenadas.

A partir de la función *houghpeaks.m* (disponible en el toolbox 'images\images' de MatLab) se calculan los máximos para asegurar que solo se detecten líneas principales. También en esta misma función se fijará un umbral de detección de los máximos de manera que solo se considerará máximo de Hough los valores de la matriz que sean iguales o superiores a dicho umbral (el resto de valores harán referencia a líneas de pocos pixeles, líneas no principales de la imagen binaria). Además, mediante esta función también se podrá especificar el número máximo de máximos de Hough que se desea detectar.

Una vez detectadas las coordenadas de los máximos de la matriz transformada de Hough, a partir de la función *houghlines.m* (disponible en el toolbox 'images\images' de MatLab) se representarán las líneas principales. Para ello, a esta función se le pasarán las coordenadas de los máximos como parámetro de entrada y de este modo se obtendrá como resultado un vector de líneas. Además como parámetro de entrada también se puede especificar la mínima longitud encontrada (despreciando las líneas que se encuentren por debajo de ese valor) y la mínima separación entre líneas. De tal manera que si se detectan dos líneas separadas una distancia menor que el valor especificado, tan solo se representará una única línea.

2.5. Operaciones morfológicas.

En visión artificial es frecuente utilizar la morfología para el tratamiento de regiones en el sentido de determinar cómo se pueden cambiar, contar o evaluar estas áreas.

La morfología se puede utilizar para las siguientes tareas:

- Suavizar los bordes de una región. Esto es útil por ejemplo si se necesita mejorar un borde, ya que usando la segmentación estándar, los bordes se presentan generalmente ruidosos debido tanto al proceso de captura como al proceso de segmentación apareciendo sobre ellos pequeños valles o protuberancias, que pueden suprimirse mediante transformaciones morfológicas.
- Separar determinadas regiones que el proceso de segmentación las presenta unidas.
- Unir regiones que han sido separadas durante la segmentación.
- Como consecuencia de los dos puntos anteriores facilitar el cómputo de regiones en una imagen.

Las operaciones matemáticas tienen su principal aplicación en imágenes binarias y los fundamentos matemáticos fueron concebidos desde el punto de vista de la posición (que es la base en el tratamiento binario) antes que desde la intensidad.

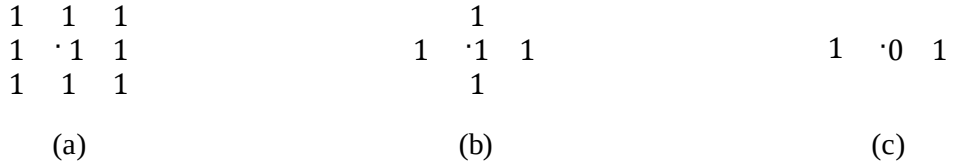
2.5.1. Principios y transformación.

La morfología matemática aprovecha las propiedades de los conjuntos de puntos, los resultados de la geometría integral y la topología. La premisa inicial consiste en suponer que las imágenes reales pueden ser modeladas utilizando conjuntos de puntos de cualquier dimensión (por ejemplo el espacio Euclídeo N-dimensional). El espacio Euclídeo 2D (E^2) y su sistema de subconjuntos es un dominio natural para la descripción de formas planas. Desde la perspectiva de conjuntos se consideran las operaciones habituales en ellos, a saber: inclusión, unión, intersección, complementario o conjunto vacio.

La visión por computador utiliza el equivalente digital el espacio Euclídeo, conjuntos cuyos elementos son pares de enteros para imágenes binarias o conjuntos de tripletes para imágenes de gris.

Un punto se representa por un par de enteros que se corresponden a las coordenadas de la imagen digital. Por tanto, una imagen binaria puede tratarse como un conjunto de puntos 2D. Los puntos que pertenecen a los objetos en la imagen representan un conjunto *X*. En imágenes binarias esos puntos son pixeles con valor binario igual a uno. Los puntos del conjunto complementario X^c corresponden al fondo con valores binarios igual a cero. Cualquier punto *x* de una imagen discreta *X* se puede considerar como un vector.

Una transformación morfológica ϕ viene dada por la relación de la imagen (conjunto de puntos de *X*) con otro pequeño conjunto de puntos *B*, llamado *elemento estructural***.** *B* se expresa con respecto a un origen local *O* (llamado punto representativo). Algunos elementos estructurales típicos (obviamente también desde el punto de vista binario) se muestran en la figura 11



(a) (b) (c)

Figura 11: Elementos estructurales.

El · indica el origen del elemento estructural. En la figura 11 se puede comprobar que el origen puede en alguna ocasión no pertenecer al elemento estructural.

La transformación morfológica $\phi(X)$ aplicada a la imagen *X* significa que el elemento estructural *B* se desplaza por toda la imagen. Suponiendo que *B* se posiciona sobre algún punto de la imagen, el pixel de la imagen correspondiente al punto representativo de *O* de *B* se denomina pixel actual. El resultado de la relación (que puede ser cero o uno) entre la imagen *X* y el elemento estructural *B* en la posición actual se almacena en el pixel actual de la imagen

2.5.2. Transformaciones morfológicas cuantitativas.

Entre las definiciones elementales se pueden encontrar la dualidad, traslación, complemento de un conjunto y diferencia de dos conjuntos. Una transformación se define cuantitativa si satisface los siguientes principios morfológicos:

a) *Compatibilidad con la traslación*: sea la transformación ϕ que depende de la posición del origen *O* del sistema de coordenadas, y se denota dicha transformación como ϕ_o . Si todos los puntos son trasladados por el vector –h se expresa como ϕ_{-h}. El principio de compatibilidad con la traslación esta dado por

$$\phi_o(X_h)=(\phi_{-h}(X))_h$$

Si ϕ no depende de la posición del origen *O* entonces la compatibilidad con el principio de traslación se reduce a la invarianza bajo traslación.

$$\phi(X_h)=(\phi(X))_h$$

b) *Compatibilidad con el cambio de escala*: suponiendo que λ es la homotecia de un conjunto de puntos *X*, por tanto las coordenadas de cada punto del conjunto se multiplican por alguna constante positiva λ. Esto es equivalente a cambiar de escala con respecto a algún origen. Sea ϕ_λ la transformación que depende del parámetro positivo λ (cambio de escala). La compatibilidad con el cambio de escala esta dada por:

$$\phi_\lambda = \lambda\phi\left[\frac{1}{\lambda}X\right]$$

Si ϕ no depende de la escala λ , entonces la compatibilidad con el cambio de escala se reduce a la invarianza del cambio de escala,

$$\Phi(\lambda X) = \lambda\phi[X]$$

c) *Conocimiento local*: el principio de conocimiento local considera la situación en la cual solo una parte de una determinada estructura puede examinarse. Este es siempre el caso en la realizada debido a la dimensión restringida de la imagen digital. La transformación morfológica ϕ posee el principio de conocimiento local si para cualquier conjunto de puntos restringido *Z´*en la transformación $\phi(X)$ existe un conjunto restringido *Z*, de forma que el conocimiento de *Z* es suficiente para

proporcionar ϕ. El principio de conocimiento local puede escribirse simbólicamente como

$$(\phi(X \cap Z) \cap Z' = \phi(X) \cap Z'$$

d) *Continuidad semi superior*: este principio dice que la transformación morfológica no exhibe ningún cambio abrupto. Las trasformaciones morfológicas más sencillas son la dilatación, erosión, apertura y cierre.

A continuación se expondrá la dilatación y erosión que son las que se aplicaron al análisis de los espectrogramas con el fin de aislar lo más posible las áreas bajo estudio.

a) Dilatación.

La transformación morfológica de la dilatación combina dos conjuntos utilizando la adición de vectores (o adición de conjuntos de Minkowski). La dilatación entre dos conjuntos es el conjunto de todas las posibles adiciones vectoriales de pares de elementos, uno de cada uno de los dos conjuntos *X* y *B*.

$$X \oplus B = \{\mathbf{d} \in E^2 : \mathbf{d} = \mathbf{x} + \mathbf{b} \ \textit{para cada} \ \mathbf{x} \in X \ \text{y} \ \mathbf{b} \in B\}$$

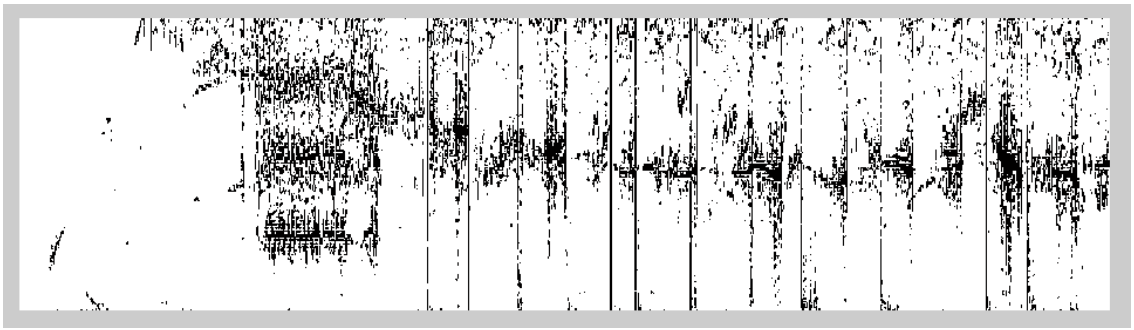


Figura 12: Imagen original antes de la dilatación.

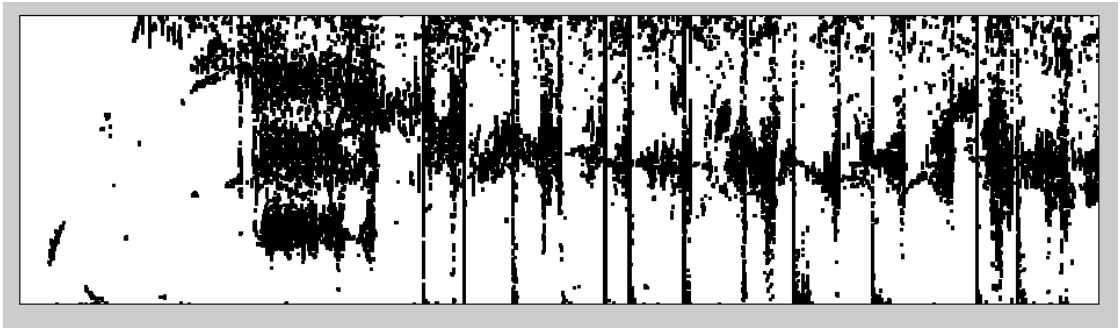


Figura 13: Imagen después de la dilatación.

En la figura 13 se puede comprobar el efecto de la dilatación. La figura 12 muestra la imagen original mientras que en la figura 13 se muestra el efecto de la dilatación por un elemento estructural de 3x3. La dilatación es isotrópica, es decir que se comporta igual en todas las direcciones, en ocasiones esta operación se denomina crecimiento o rellenado. La dilatación tiene algunas propiedades como pueden ser la conmutativa, asociativa y además es invariante a la traslación.

Gráficamente la dilatación se realiza como sigue: se va recorriendo la imagen por ejemplo de izquierda a derecha y de arriba abajo y donde se encuentre un '1' se situa el origen del elemento estructural. Sobre ese uno, es esa la posición donde se realiza la unión del elemento estructural con la parte de la imagen sobre la que se solapa dicho elemento; si todos los unos del elemento estructural coinciden con unos de la imagen entonces se marca el pixel de la imagen donde está el origen del elemento estructural con el valor 1

b) Erosión.

La transformación morfológica de la erosión combina dos conjuntos utilizando la substracción de vectores. Es dual de la dilatación. Ni la erosión ni la dilatación son transformaciones invertibles.

$$X \otimes B = \{\mathbf{d} \in E^2 : \mathbf{d}+\mathbf{b} \in X \ \ para\ cada \quad b \in B\}$$

Esta expresión dice que cada punto d del conjunto X, que será la imagen, es restado; el resultado de la erosión esta dado por los puntos d para los cuales todos los posibles $d+b$ están en X.

Gráficamente la erosión se realiza como sigue: se va recorriendo la imagen por ejemplo de izquierda a derecha y de arriba abajo y donde se encuentre un 1 se situará el origen del elemento estructural sobre ese 1, si todos los unos del elemento estructural coinciden con unos de la imagen, entonces se señala el pixel de la imagen donde está el origen del elemento estructural con el valor 1.

La figura 15 muestra el efecto de la operación de erosión. La figura 14 muestra la imagen original y la figura 15 muestra el resultado.

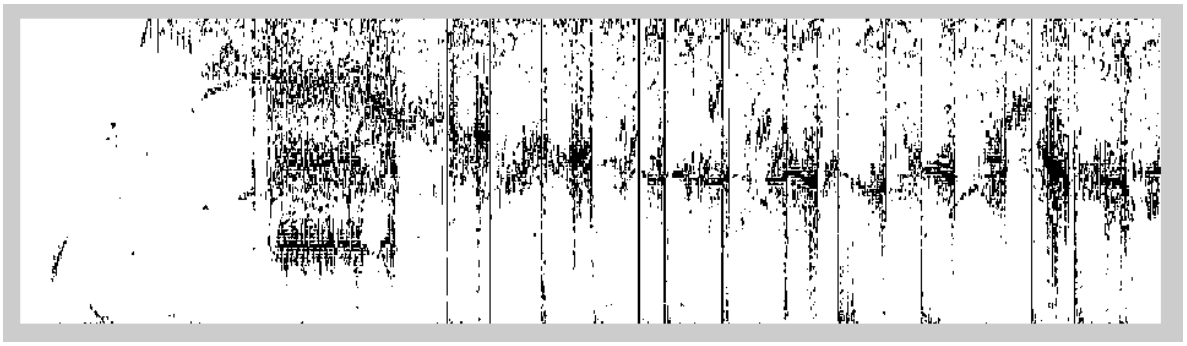


Figura 14: Imagen antes de la erosión.

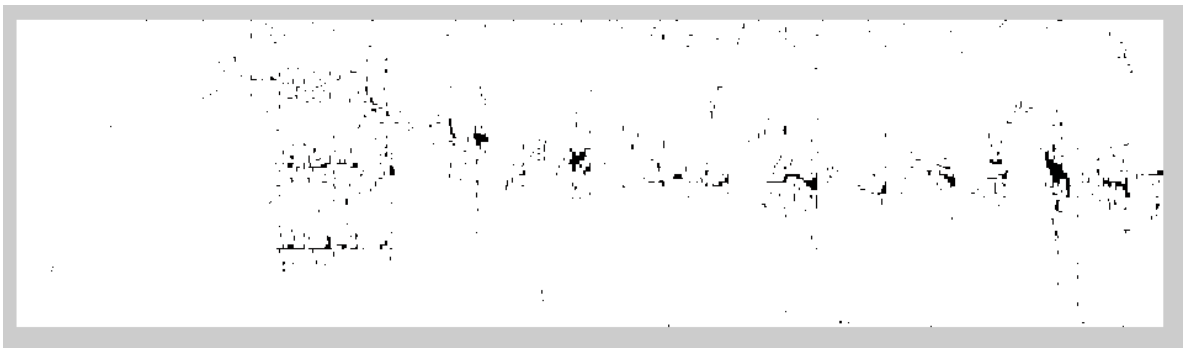


Figura 15: Imagen después de la erosión.

Se puede apreciar la desaparición de muchos contornos existentes en la imagen original. En algunas ocasiones se puede referir a esta operación como de reducción. La erosión es utilizada para simplificar la estructura de los objetos, ya que los objetos o partes de ellos que son pequeños (uno o dos pixeles) desaparecen, según esto objetos complicados pueden descomponerse en otros más simples.

Algunas propiedades de la erosión son:

a) Si el punto representativo es un miembro del elemento estructural la erosión es una transformación *anti-extensiva*, esto es, si $(0,0) \epsilon B$ entonces $X \otimes B \subseteq X$.
b) La erosión es también invariante a la traslación

$$X_h \otimes B = (X_h \otimes B)_h$$

$$X \otimes B_h = (X \otimes B)_{-h}$$

c) Es una transformación creciente si $X \subseteq Y$. Entonces $X \otimes B \subseteq Y \otimes B$.
d) Si B y D son elementos estructurales y D esta contenido en B, la erosión por B es más agresiva que por D; esto es, si $D \subseteq B$ entonces $X \otimes B \subseteq X \otimes D$

e) Anteriormente se ha comentado que la erosión y la dilatación son transformaciones duales lo que formalmente se escribe como:

$$(X \otimes Y)^c = X^c \otimes Y$$

Las transformaciones morfológicas básicas pueden usarse para encontrar los contornos de los objetos. Esto puede lograrse por ejemplo mediante la operación lógica XOR de la imagen original con la imagen dilatada(o erosionada).

c) Apertura y cierre.

Las dos operaciones anteriores han servido para ilustrar la operación que sigue a continuación. Se trata de la operación de cierre. La erosión y la dilatación son transformaciones no invertibles. Si una imagen es erosionada y luego dilatada, la imagen original no se recuperara. En efecto, el resultado es una imagen más simplificada y menos detallada que la imagen original.

La erosión seguida de una dilatación crea una transformación morfológica importante llamada *apertura*. La *apertura* de una imagen X por un elemento estructural B se denota por $X \circ B$ y se define como

$$X \circ B = (X \otimes B) \oplus B$$

La dilatación seguida de la erosión crea una transformación llamada *cierre*. El *cierre* de una imagen X por un elemento estructural B se denota por $X \bullet B$ y se define como,

$$X \bullet B = (X \oplus B) \otimes B$$

Si una imagen X permanece invariable por apertura con respecto al elemento estructural B se dice que es abierta con respecto a B. Análogamente, si una imagen X permanece invariable por cierre con respecto al elemento estructural B se dice que es cerrada con respecto a B.

La apertura y cierre con un elemento estructural isótropo se utiliza para eliminar detalles específicos de la imagen más pequeños que el elemento estructural.

La forma global de los objetos no se distorsiona. El cierre conecta objetos que están próximos entre sí, rellena pequeños huecos y suaviza el contorno del objeto rellenando los pequeños

valles, mientras que la apertura produce el efecto contrario. Los conceptos de pequeño y próximo están relacionados con la forma del elemento estructural.

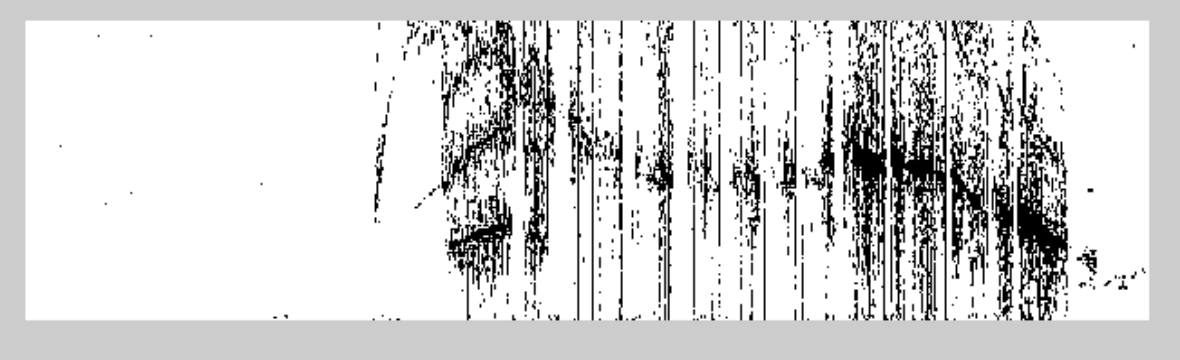


Figura 16: Imagen original.

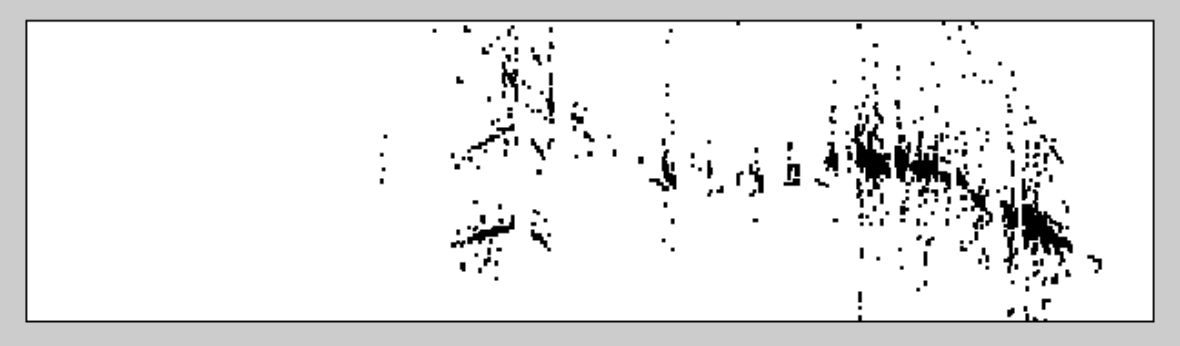


Figura 17: Operación de cierre sobre la figura 16.

El cierre se ilustra en las figuras 16 y 17. La figura 16 es la imagen original y la 17 el resultado del cierre. La apertura y cierre son invariantes a la traslación del elemento estructural. El hecho de que tanto la dilatación como la apertura sean transformaciones crecientes implica que tanto la apertura como el cierre también lo son.

La apertura es *anti-extensiva* y el ($X \circ B \subseteq X$) y el cierre es *extensivo* ($X \subseteq X \bullet B$). La apertura y el cierre a igual que la dilatación y la erosión, son transformaciones duales.

$$(X \bullet B)^c = X^c \circ B$$

Otro importante hecho es que la apertura y el cierre utilizadas iterativamente son *idempotentes*, lo que implica que la aplicación de esas transformaciones no cambia el resultado previo. Formalmente:

$$X \cdot B = (X \cdot B) \cdot B \qquad X \bullet B = (X \bullet B) \bullet B$$

2.6. Regiones.

Enmarcado en el entorno de una imagen, una región es un tipo de figura determinada. Si se considera que podemos interpretar como tal ciertas áreas de una imagen, entonces se podrá analizar sus propiedades para de esta forma investigar su descripción. Este capítulo trata precisamente de las propiedades que serán útiles como herramientas de investigación.

2.6.1. Propiedades de una imagen.

Una región se puede describir por la forma de su frontera o por sus características internas. De acuerdo al criterio de Duda y col. (2000) se trata de describir la forma de una figura que ha sido extraída del plano de la imagen.

Dependiendo del criterio o del punto de vista a utilizar una región puede describirse de distintas maneras. En efecto, puede ser vista como un conjunto de puntos conectados entre sí, es decir, partiendo de un punto de la región se puede llegar a otro punto de la misma sin abandonarla; o puede ser descrita por el número de huecos que presenta. Son en ambos casos propiedades *topológicas*. Por el contrario, si se pueden obtener valores tales como el área, perímetro, etc., se referirán a propiedades *métricas*. Pero también es posible fijarse en las *irregularidades* de las regiones para su diferenciación. Finalmente, el *esqueleto* de una región puede resultar una propiedad de interés.

2.6.2. Propiedades topológicas.

Una propiedad topológica se caracteriza por ser invariante a ciertas deformaciones de las figuras en la imagen. De forma gráfica, si el plano de la figura fuera representado por una hoja de papel deformable, con posibilidad de arrugarse, entonces la propiedad de la figura permanecería invariable ante una deformación de la hoja. Una distorsión del plano de esta forma se llama *homeomorfismo* del plano en si mismo.

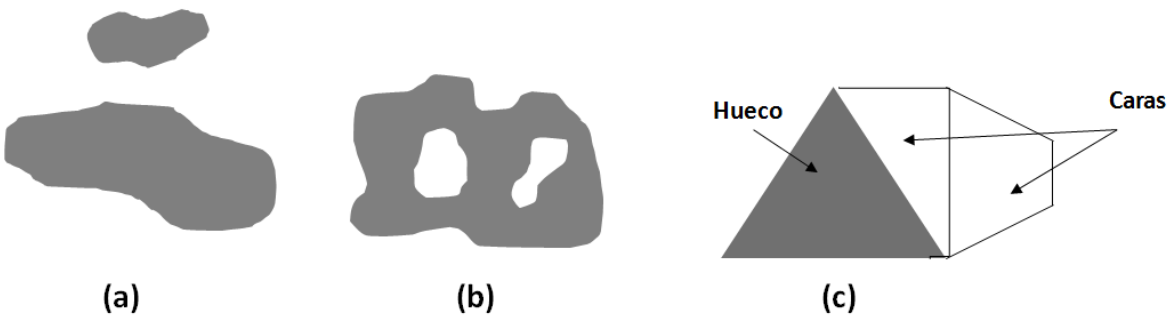


Figura 18: (a) figura con dos componentes conexas, (b) figura con dos huecos número Euler-1, (c) red poligonal.

Formalmente se define como una proyección continua uno a uno cuya inversa es también continua. Las propiedades topológicas no pueden involucrar nociones de distancia, puesto que las distancias son distorsionadas por proyecciones topológicas. Del mismo modo, tampoco puede involucrar nociones de distancia de forma indirecta, tales como áreas, paralelismo de curvas, perpendicularidad de líneas, etc.

Una de las descripciones topológicas de un conjunto más usadas normalmente es el número de sus *componentes conexas*. Una componente conexa de un conjunto es un subconjunto de dimensión máxima tal que cualesquiera dos de sus puntos pueden unirse por una curva continua perteneciente enteramente al subconjunto.

Una segunda propiedad topológica de interés es el número de *huecos* en la figura. Formalmente, el número de huecos en la figura es uno menos que el número de componentes conexas en el complemento de la figura. Si X es el número de componentes conexas de una figura y H el número de huecos, se define el numero de *Euler*: E =C-H, que también es una propiedad topológica. Como ejemplo, el número de Euler de las letras A y B son 0 y -1 respectivamente.

Si se restringe la atención a las figuras hechas con líneas rectas, se puede hablar de *redes poligonales* cuando un polígono está contenido en otro, y en ese caso, puede ser de interés el hecho de distinguir entre dos tipos de regiones de dicha red y llamar algunas de ellas *caras* y a otras *huecos*. La figura 18(c) muestra una red poligonal con dos componentes conexas, dos caras y un hueco. El número de Euler de una red poligonal puede describirse de una forma particularmente simple. Si V es el número de vértices, S el número de lados y F el número de caras, entonces la *formula de Euler* relacionando esas cantidades es:

$$V - S + F = C - H = E$$

En la figura 18(c), por ejemplo, 6-8+2=1-1=0. La primera parte de la igualdad se obtiene al considerar que V=6, S=8 y el número de caras es F=2. Además, el número de regiones conexas es C=1 (una región formada por las dos caras) y el número de huecos es H=1.

Las descripciones topológicas de las figuras encuentran aplicación en el análisis de la escena como un parámetro preliminar a otros descriptores más precisos. Por ejemplo, en reconocimiento de caracteres, si el número de Euler no coincide con un determinado patrón, el carácter ya no será comparado.

2.6.3. Métricas.

Existen figuras que dependen de una *métrica* determinada. Conceptualmente las métricas son generalizaciones de la distancia Euclídea, así una propiedad métrica cambiará si el plano de la figura se distorsiona. Entre las propiedades métricas más simples de las figuras destaca el *área*, el *perímetro* y el *centro de gravedad*. El área *A* es el número de pixeles contenidos dentro de su frontera. El perímetro *P* es la longitud de su frontera, que se puede obtener a partir del código de cadena de la frontera como sigue:

$$P = \sum_i \sqrt{(x_i + x_{i-1})^2 + (y_i + y_{i-1})^2}$$

Aunque a veces se usa el perímetro como descriptor, su aplicación más usual se da en la obtención de la compatibilidad de una región que se define como P^2/A. Este es un valor sin dimensiones que es mínimo para una región en forma de disco.

El centro de gravedad (*x*,*y*) es un único punto representativo de la región y se obtiene como sigue:

$$\bar{x} = \frac{1}{A}\sum_i x_i \qquad \bar{y} = \frac{1}{A}\sum_i y_i$$

Se debe observar que el valor no es entero. En el cálculo del centro de gravedad se ha dado el mismo peso específico a todos los pixeles que intervienen, sin embargo a veces puede resultar de interés calcular el centro de gravedad teniendo en cuenta el valor de la intensidad en cada punto en cuyo caso le ecuación anterior se modifica de la siguiente manera,

$$\bar{x} = \frac{\sum_i f(x_i, y_i) x_i}{\sum_i f(x_i, y_i)} \qquad \bar{y} = \frac{\sum_i f(x_i, y_i) y_i}{\sum_i f(x_i, y_i)}$$

El centro de gravedad se puede calcular también a partir de un código de cadena con *n* elementos representando el borde que limita la región, para ello se utilizan los pares de coordenadas x_i, y_i de cada punto en el borde donde x_0, y_0 y x_n, y_n son el mismo punto en el borde y por tanto el contorno debe ser cerrado,

$$\bar{x} = \frac{1}{A}\sum_i (x_i + x_{i-1})^2 (y_i + y_{i-1})^2 \qquad \bar{y} = \frac{1}{A}\sum_i (y_i + y_{i-1})^2 (x_i + x_{i-1})^2$$

En este caso el área se calcula como:

$$A = \frac{1}{2}\sum_i (y_i + y_{i-1})(x_i + x_{i-1})$$

Otra propiedad de interés es la *elongación* o *razón de aspecto*, que en el caso de un rectángulo es la razón de su longitud a su ancho. Un valor de uno seria un cuadrado y próximo a 1 daría la impresión de un rectángulo grueso. Para generalizar este concepto, a las regiones de cualquier tipo, se circunscribe un rectángulo a la figura y se calcula *la razón de aspecto* del rectángulo circunscrito.

Los ejes mayor y menor de una región se definen en términos de su frontera y son útiles para obtener la *orientación* de un objeto. El cociente de las longitudes de estos ejes llamado *excentricidad* de la región, es también un descriptor global importante de la forma del objeto.

La *redondez* de una región se define como la razón entre el área y el eje mayor al cuadrado y la *compacidad* como el cociente entre la raíz cuadrada del área y el eje mayor.

Finalmente, se puede ajustar una recta a la región mediante el procedimiento de la transformada de Hough, en esta línea determina también la *orientación* de la región, mientras que el ángulo de esta línea se puede obtener a partir del conjunto de ecuaciones siguiente:

$$S_x = \sum x_i,\ S_y = \sum y_i,\ \ S_{xx} = \sum x_i^2,\ \ S_{yy} = \sum y_i^2,\ \ S_{xy} = \sum x_i y_i$$

$$M_{xx} = S_{xx} - \frac{S_x^2}{A},\quad M_{yy} = S_{yy} - \frac{S_y^2}{A},\quad M_{xy} = S_{xy} - \frac{S_x S_y}{A}$$

Finalmente la orientación viene dada por:

$$\phi = tan^{-1}\left\{\frac{M_{xx} - M_{yy} + \sqrt{\left(M_{xx} - M_{yy}\right)^2 + 4{M_{xy}}^2}}{M_{xy}}\right\}$$

3. ALGORITMOS DESARROLLADOS.

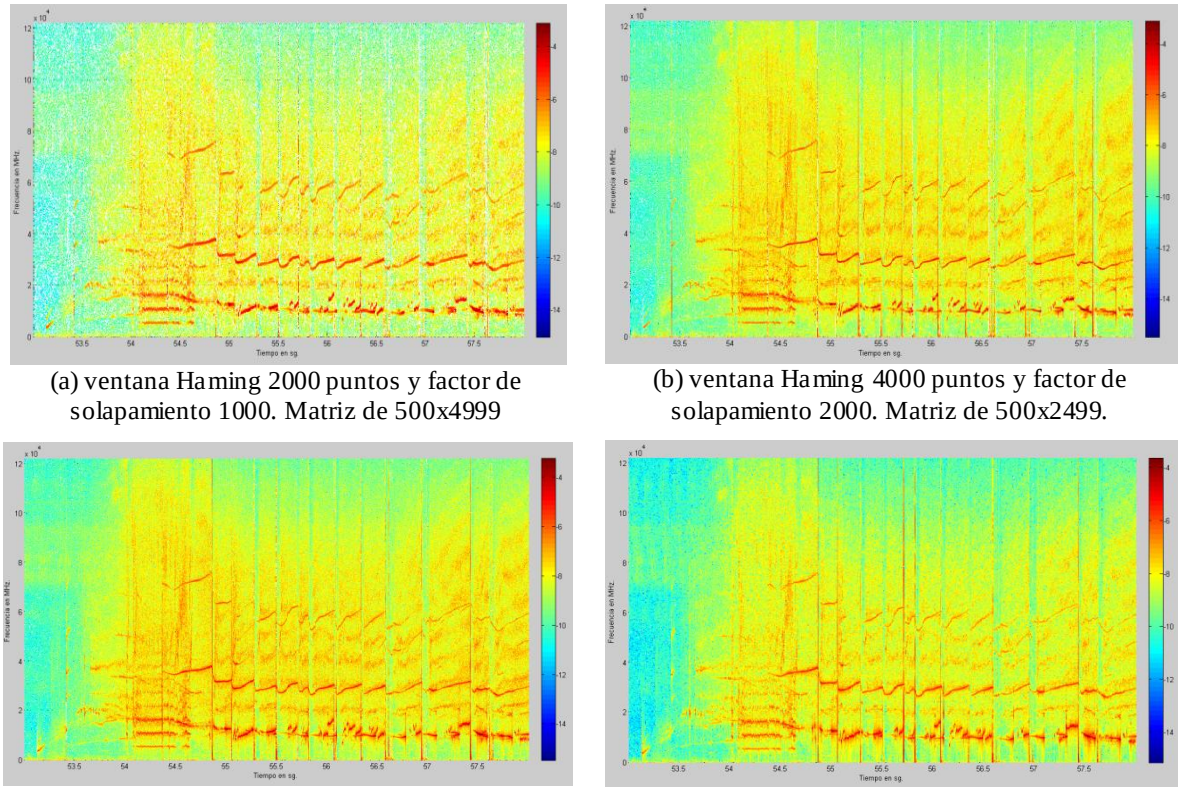
3.1. Introducción.

Este capítulo se centra básicamente en los dos algoritmos que se desarrollaron para la detección de armónicos en las descargas de plasma nuclear.

Se verá la aplicación práctica de todos los métodos vistos en el capitulo anterior.

El objetivo es básicamente detectar cual de los dos métodos ofrece más exactitud.

En un primer apartado se verá el cálculo del espectrograma. También aquí se comprobará cómo la variación de los parámetros lleva a resultados muy diferentes. Posteriormente se entrará directamente en la aplicación de los métodos que afecta directamente al tratamiento de imágenes y se detallará los dos algoritmos aplicados, finalizando con los resultados obtenidos por uno y otro algoritmo.



(a) ventana Haming 2000 puntos y factor de solapamiento 1000. Matriz de 500x4999

(b) ventana Haming 4000 puntos y factor de solapamiento 2000. Matriz de 500x2499.

(c) ventana Haming 6000 puntos y factor de solapamiento 3000. Matriz de 500x1665.

(d) ventana Haming 10000 puntos y factor de solapamiento 5000. Matriz de 500x999.

Figura 19: Cálculo de espectrogramas.

3.2. Cálculo del espectrograma.

El espectrograma es la base principal del análisis. Para que el objetivo pueda conseguirse es necesario que la imagen tenga la mayor definición posible. Esto solo se conseguirá mediante una imagen cuya matriz tenga el mayor número de líneas y columnas posible. Además se debe conseguir que el cálculo sea lo más rápido posible ya que el método está orientado a análisis de descargas masivas. Esto hace que el factor tiempo sea muy importante.

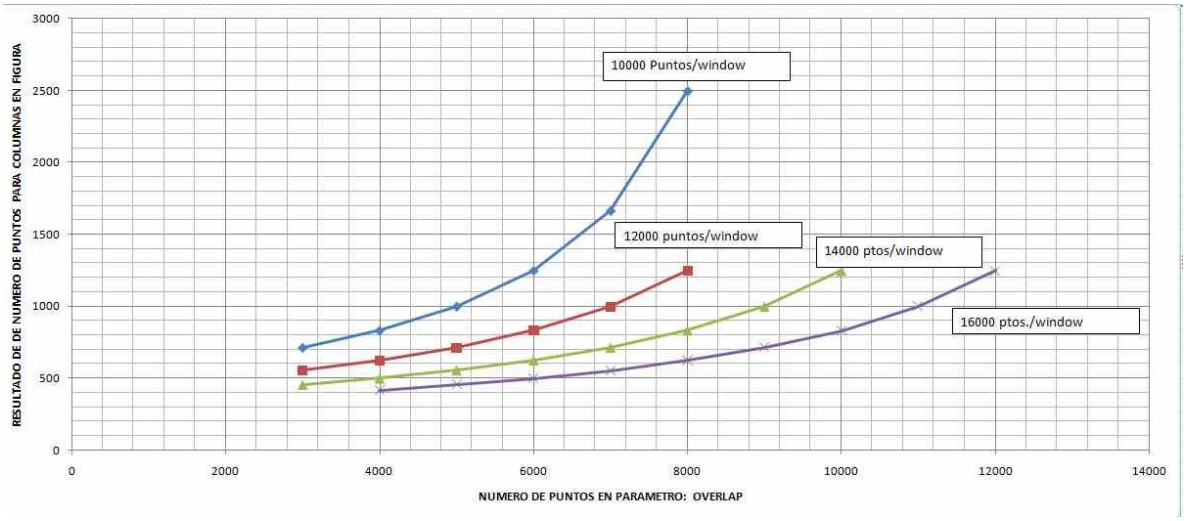


Figura 20: Relación entre los parámetros puntos por ventana y solapamiento.

El efecto de la ventana.

El principal efecto de la ventana en la transformada de Fourrier dependiente del tiempo es limitar la extensión de la secuencia que se va a transformar de forma que las características espectrales sean razonablemente estacionarias en el intervalo de duración de la ventana.

Cuanto más rápido cambien las características de la señal más corta deberá ser la ventana. A medida que la longitud de la ventana decrece también la resolución en frecuencia decrece. Por otra parte, a medida que decrece la longitud de la ventana, aumenta la potencia de resolver cambios en el tiempo. Por tanto aparece un compromiso en la selección de la longitud de la ventana, entre la resolución en el tiempo y en la frecuencia.

La definición del espectrograma debe ser alta pues el objetivo es analizar áreas donde pueda haber regiones de interés. En resumen, se ha de llegar a un compromiso entre definición de la

imagen y tiempo de computación. La figura 19 muestra el espectrograma de la descarga 79455, señal PP8 y canal 6. Aparentemente los cuatro espectrogramas son iguales, sin embargo la definición a nivel de columnas de la matriz de color es diferente.

Se puede comprobar que, para unos valores de ventana Hamming de 1000 puntos y un factor de solapamiento de 5000, se obtiene un espectrograma con buena definición. Por otro lado se debe tener en cuenta los tiempos de cómputo. Para el espectrograma 19(a) el tiempo de proceso es de 18.4531 segundos, 19(b) 12.5156 sg., 19(c) 11.3125 sg. y 19(d) 7.7813 sg.

Existe una diferencia de aproximadamente 11 segundos entre el modelo 19(a) y 19(c). Si se tiene en cuenta que en cada descarga aparecen cinco señales y por cada señal, una media de seis canales, resultarán entonces 840 ficheros x 11 seg./fichero=9240 seg.. Esto supone un retraso de 2 horas en un análisis de 28 descargas.

El espectrograma fue calculado mediante la función Matlab *spectrogram.m*. Esta función dispone de cuatro parámetros que son los que se han tenido que modificar hasta llegar a los valores que muestran un espectrograma bien definido tal como muestra la figura 19.

3.3. Algoritmo mediante la transformada de Hough.

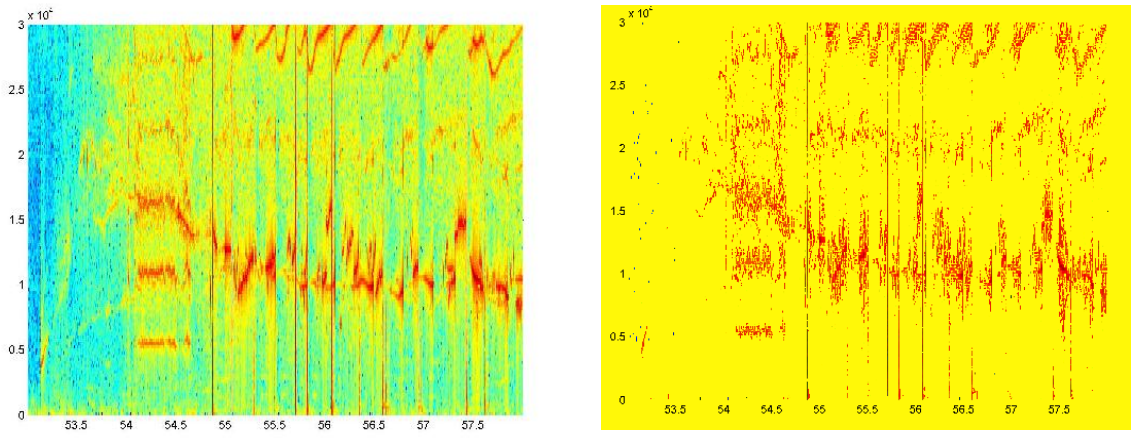
El algoritmo que se presenta a continuación tiene como comienzo el cálculo del espectrograma y tiene como base la aplicación de la transformada de Hough. Para aplicar la transformada a una imagen es preciso prepararla previamente. Los pasos del algoritmo fueron los siguientes:

1. Cálculo del espectrograma de la descarga.
2. Recortar imagen.
3. Convertir espectrograma a escala de grises.
4. Aplicar filtro para eliminar ruido.
5. Binarizar la imagen mediante método Otsu.
6. Relleno de huecos (apertura y cierre).
7. Aumento de las líneas horizontales mediante máscara.
8. Cálculo de la matriz de Hough.
9. Cálculo de los picos de Hough.

10. Cálculo de las líneas paralelas al eje x.

Para la realización del método se parte, como ya se ha indicado, del cálculo del espectrograma. Una vez obtenida la imagen se debe recortar la zona a estudiar. Esto es muy necesario, ya que de otra manera se corre el riesgo de perder definición debido a que la imagen tiende a "estirarse".

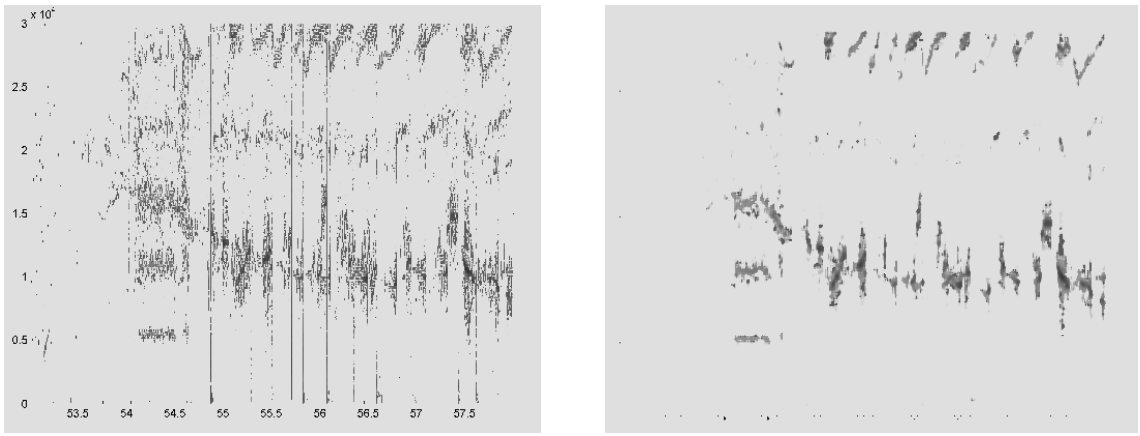
Con la imagen resultante, se inicia el tratamiento mediante filtros. En primer lugar se deben realzar las componentes de interés. En este caso, es la componente de alta potencia, es decir, la componente roja de la imagen. La figura 21(a) muestra la zona de interés y la figura 21(b) la componente roja separada del resto de componentes.



(a) (b)

Figura 21: (a) extracto del área bajo estudio, (b) realce de la componente roja para mejorar la binarización.

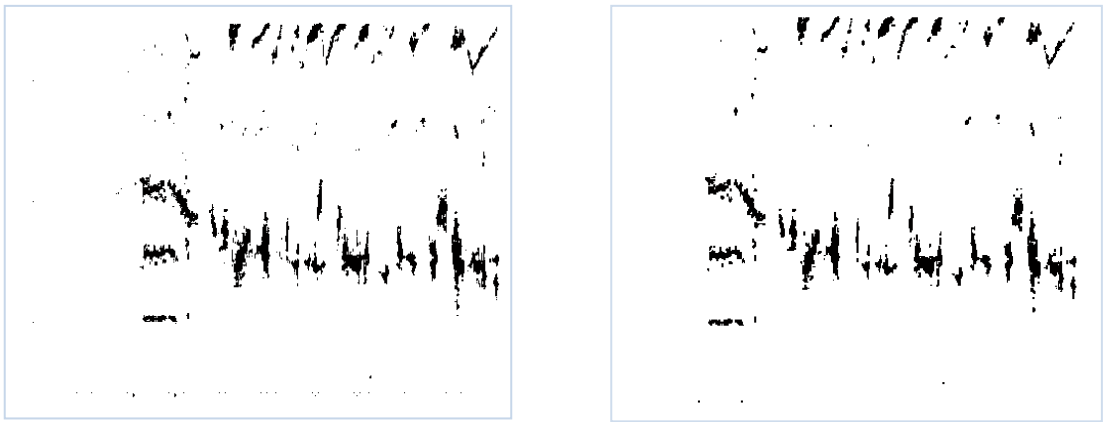
Aislar esta componente se hace necesario ya que al aplicar el filtro, éste convertirá zonas de alta intensidad (rojas, de interés) en zonas visibles, despreciando las de baja intensidad (componente azul y que no interesan).



(a) (b)

Figura 22: (a) imagen convertida a escala de grises, (b) misma imagen con filtro Nagao y Matsuyama.

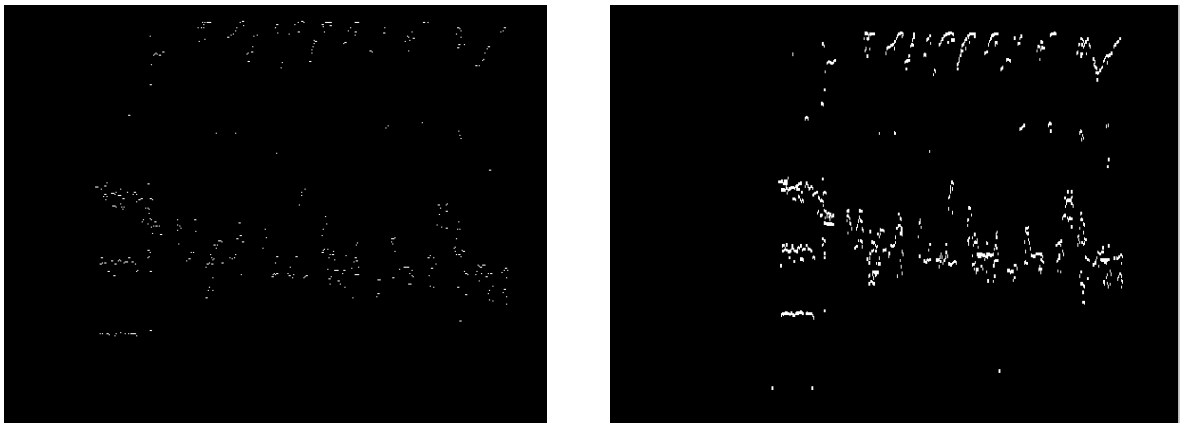
Una vez aislada la componente roja, se debe hacer un escalado de la imagen a grises. Posteriormente se debe aplicar un filtro para eliminar todo el ruido existente. La figura 22 muestra esta acción. En la figura 22(b) se ve la acción del filtro sobre la figura 22(a). Se trata del filtro basado en el algoritmo de Nagao y Matsuyama también llamado filtro sal y pimienta. El siguiente paso es Binarizar la imagen tratada. Para ello se utiliza el algoritmo de Otsu, que calculará el umbral de binarización. Posteriormente se rellenarán huecos tal como se puso de manifiesto en la sección operaciones morfológicas concretamente en el apartado de apertura o cierre. El resultado de estas dos acciones es el que se puede ver en la figura 23.



(a) (b)

Figura 23: a) binarización mediante el umbral de Otsu, (b) misma imagen pero con relleno de huecos.

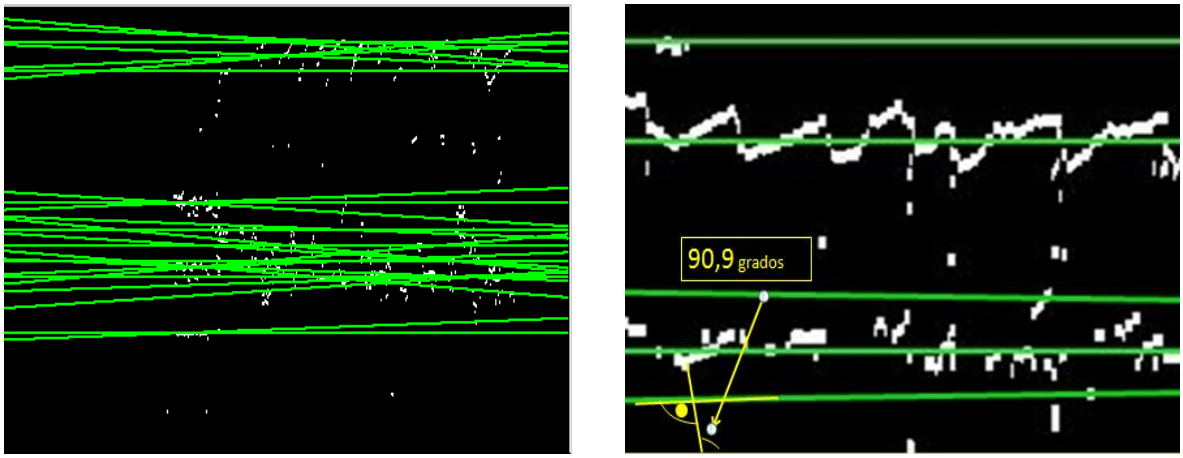
En la figura 23(a) se ve cómo la imagen se ha limpiado considerablemente. En figura 23(b) se han rellenado los huecos, para hacer más apreciable este hecho se rodea una zona donde esta acción se hace más visible.



(a) (b)

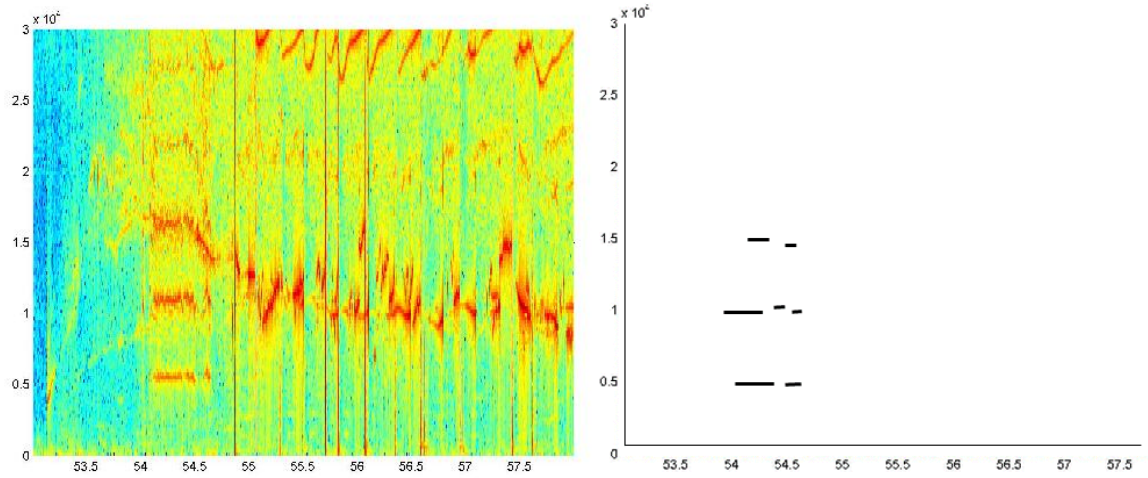
Figura 24: (a) imagen con los bordes superiores aislados, (b) misma imagen con la aplicación de una máscara para ampliar las zonas de estudio.

La figura 25 muestra la aplicación de la transformada de Hough. En la figura 25(a) se muestran pintadas de verde las rectas detectadas. En la figura 25(b) se muestra el criterio aplicado para tomar como recta positiva teniendo en cuenta la inclinación frente al eje de abcisas.



(a) (b)

Figura 25: (a) imagen con las líneas de Hough detectadas, (b) criterio seguido para tener en cuenta las rectas.



(a) (b)

Figura 26: (a) la imagen con el área bajo análisis, (b) extracción de armónicos.

No se impone paralelismo de las líneas de Hough respecto al eje de abcisas. Hay que darse cuenta que, tal como están dispuestos los picos de Hough, es muy probable que muchas de las líneas tengan cierta pendiente respecto a este eje.

Por este motivo se impone a las líneas un intervalo de inclinación. De esta forma, se considerará un umbral en grados que varía desde 89,1 a 90,9. La figura 25(b) muestra esta situación. Se admiten como correctas líneas que serán totalmente paralelas a eje de abcisas y las que tengan cierta inclinación. Una vez determinadas las rectas ya solo queda calcular las rectas cuya separación entre ellas sean múltiplos exactos. Esto dará los números de armónicos que se pueden encontrar. Una vez detectadas las rectas se debe calcular el inicio y fin de cada armónico. La figura 26 muestra esta idea. Para llegar a esta solución se deben calcular los picos de las rectas de Hough. Tomando como referencia cada recta, se deberá analizar cada pico por el que pase la recta. De tal forma que se considere como armónico cada longitud de pico a pico. Este cálculo deberá hacerse en la misma columna de tiempo en la cual se sitúa el armónico fundamental. No obstante, hay que mantener un umbral lateral en porcentaje de pixeles. No se puede ser estricto en este punto. En la figura 26(a) se ve que el fundamental comienza en 54 segundos, pero el primer armónico comienza en aproximadamente 53,7 segundos. Así pues, este umbral debe ser controlado y además se debe mantener a nivel de líneas de separación entre cada armónico ya que no siempre la multiplicidad de líneas es exacta. También aquí se debe mantener un porcentaje de error en cuanto a multiplicidad de líneas.

3.4. Algoritmo mediante la detección de regiones.

La detección de regiones es muy parecida al método anterior. En este caso en lugar de analizar rectas se analizan áreas de interés, que deben cumplir las mismas exigencias que las rectas de Hough en lo referente a multiplicidad de armónicos

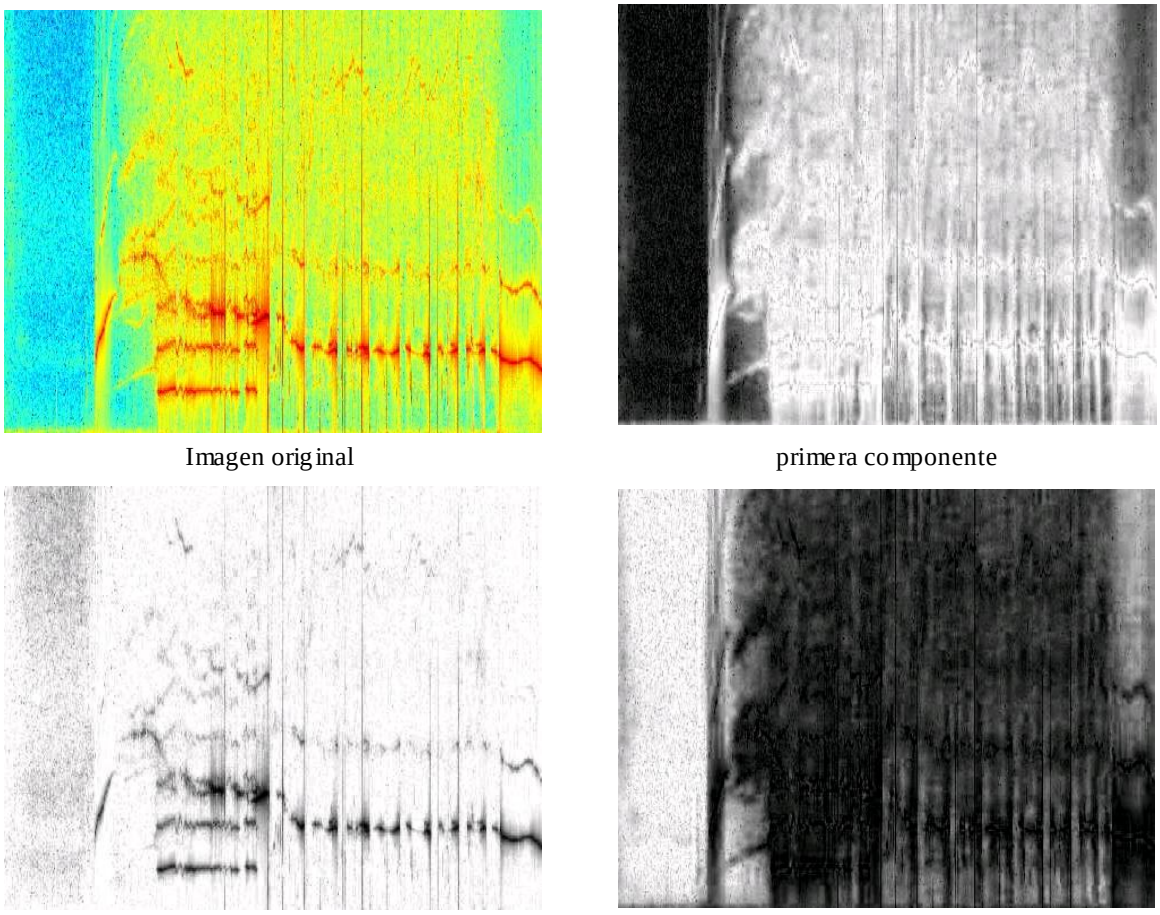


Imagen original | primera componente

Segunda componente | Tercera componente

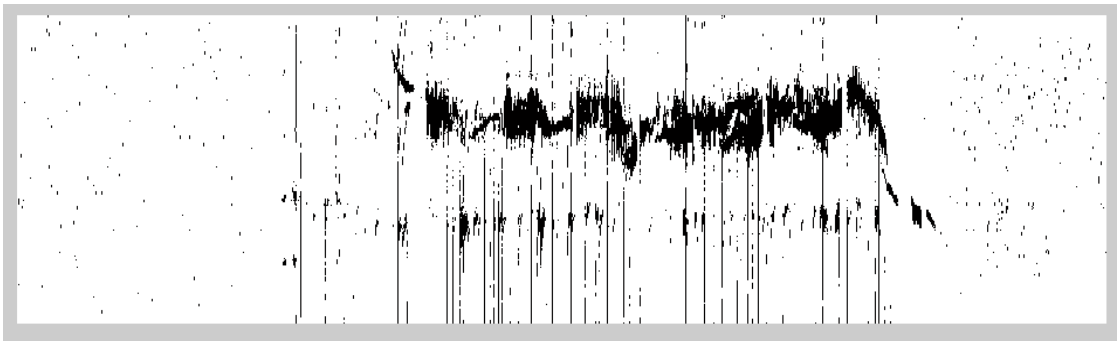
Figura 27: Imagen original y sus componentes.

Este método también parte del cálculo inicial del espectrograma. El Algoritmo es el siguiente:

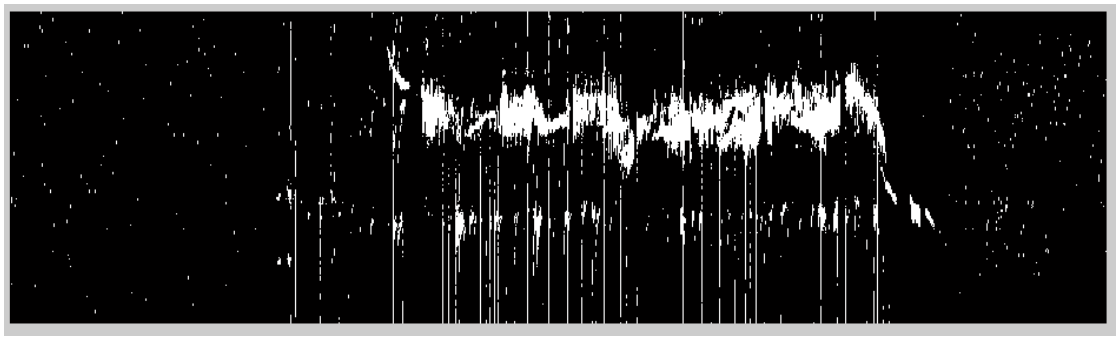
1. Cálculo del espectrograma de la descarga.
2. Extracción de la componente con mayor dispersión.
3. Binarización de la imagen anterior.
4. Aplicación operación XOR.
5. Eliminación de grandes áreas sin interés.
6. Asignación de etiquetas a cada área.

7. Eliminación mediante estudio de propiedades.
8. Cálculo de multiplicidades.

Se puede comprobar que este método tiene menos pasos que en el anterior. En principio es un método más rápido pero el cálculo de multiplicidades así como el control de la lateralidad de las áreas deben ser tenidas en cuenta. El paso 1 es como en el método anterior. Ya puestos en el paso dos se cambia de trayectoria. Toda imagen en color tiene tres componentes. Realmente una imagen en color es una matriz tridimensional la cual tiene asignado en cada pixel el valor de cada una de las tres componentes, a saber; rojo, verde y azul. Pues bien, como punto de partida se debe aislar la dimensión que más dispersión tiene (la que más variabilidad tenga de las tres).



(a)



(b)

Figura 28: (a) imagen binarizada, (b) la misma imagen aplicada operación XOR.

La figura 27 pone de manifiesto todo lo que se está explicando. En esta figura se puede ver la figura principal y después las tres componentes. En ella se puede apreciar cómo la segunda componente es la de mayor dispersión. La señal está más definida. En esta segunda

componente se puede apreciar más claramente las áreas a estudiar posteriormente. En este caso, el algoritmo seleccionaría la segunda componente para continuar con el estudio.

Una vez seleccionada la componente el siguiente paso es binarizar la imagen y aplicar una operación XOR. El resultado de estas dos operaciones es la figura 28. Este paso es necesario como preparatorio a la asignación de etiquetas a cada región. No obstante antes de comenzar con esta tarea se debe eliminar el mayor número de regiones que no sean de utilidad.

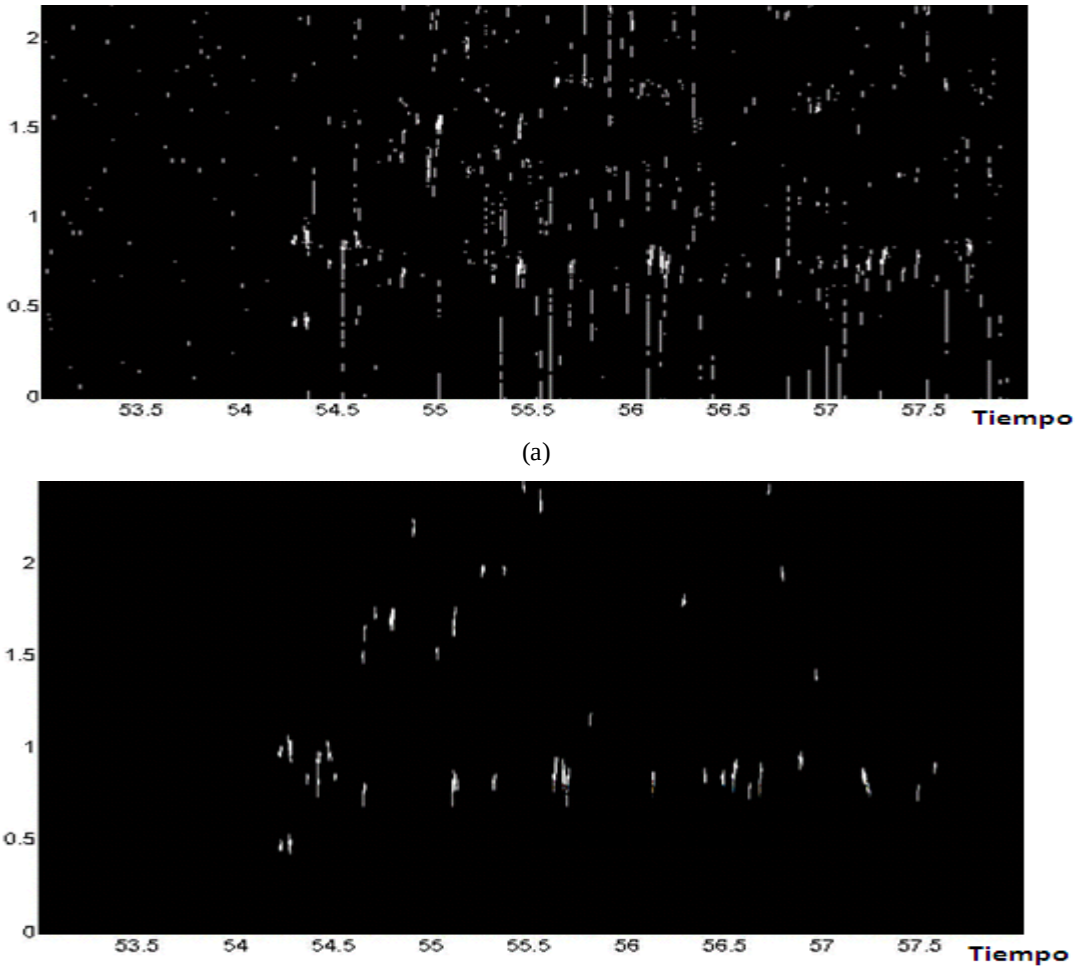


Figura 29: (a) imagen con ruido y zonas a eliminar, (b) la misma imagen limpia de áreas sin interés para el estudio del problema

La imagen 29 muestra el tratamiento habiendo eliminado las grandes áreas que carecen de interés para este caso. La figura 29(a) muestra este resultado pero la imagen aún tiene información a retirar como puede ser el ruido y las regiones perpendiculares al eje de abcisas. Estas áreas podrían dar lecturas erróneas, por ese motivo deben ser descartadas del estudio. Al final el análisis de multiplicidades de armónicos se centrará sobre la figura 29(b), donde se

ven como existen zonas que muy posiblemente correspondan a posibles armónicos; en particular, en la columna temporal de 54,3 segundos. Ahí se intuye un posible fundamental y un primer armónico. En este caso se debería analizar la multiplicidad frecuencial. El fundamental esta en 5 Khz y el primer armónico se encuentra en 10 Khz. Por tanto se puede concluir que se trata de armónicos.

4. ANÁLISIS DE RESULTADOS.

4.1. Objetivos del análisis.

En este punto se trata de verificar y validar las estrategias propuestas. Anteriormente se ha comentado que no existe investigación especifica en esta área. Esto es lo que motiva que no sea posible comparar estos métodos con los propuestos en otras estrategias.

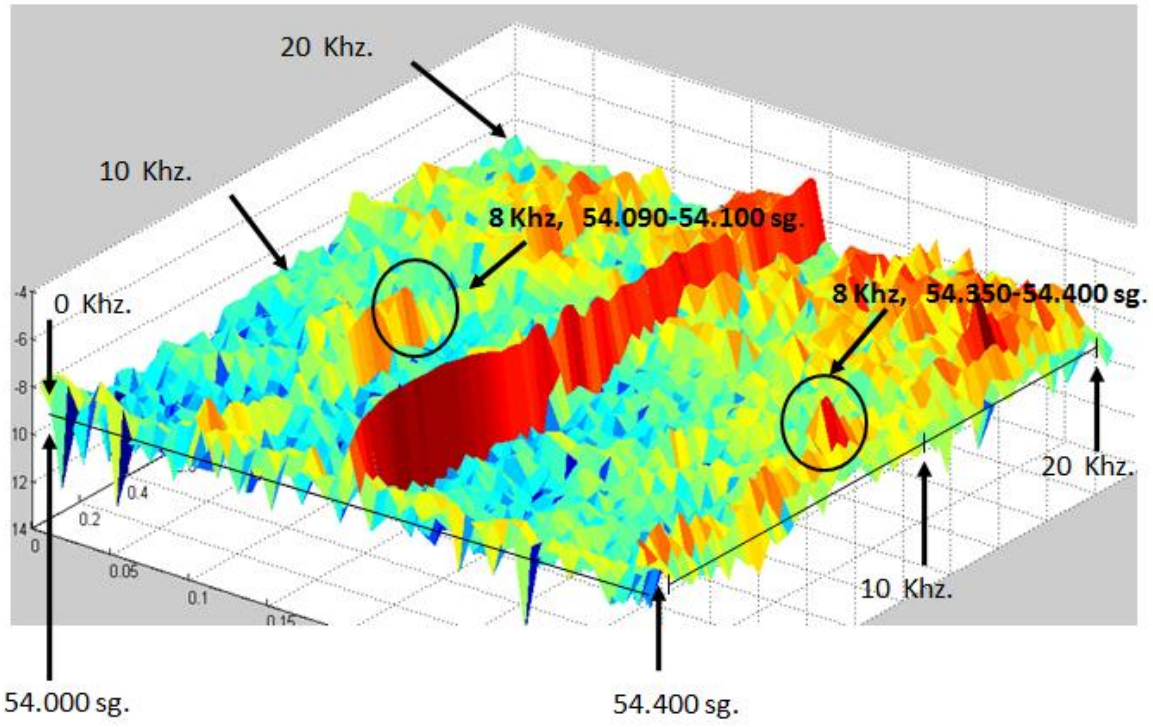


Figura 30: Análisis en detalle del área 0-20 Khz. de la descarga número 79461.

En cuanto a la precisión de los resultados se puede comprobar cual es el mejor de los dos desarrollados. La comprobación consiste en contrastar los resultados obtenidos frente a los proporcionados por el **CIEMAT**.

En lo referente a la validación resulta difícil de cuantificar pues hasta ahora el único método de validación conocido es el ojo humano.

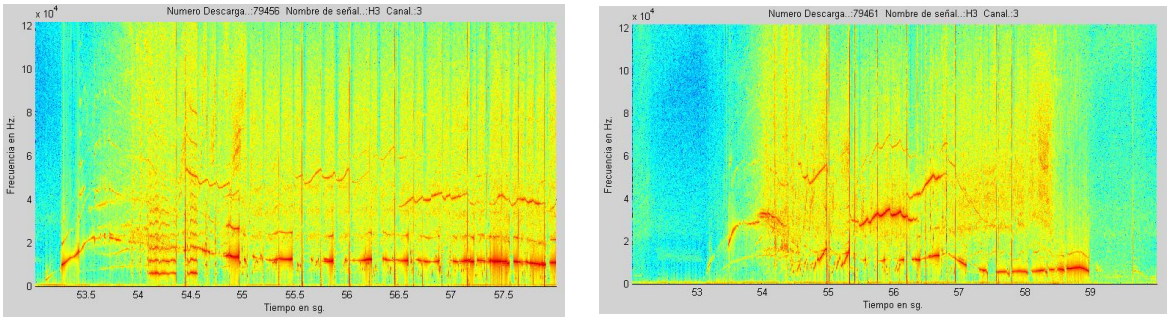
Las pruebas para analizar los resultados se realizaron con descargas proporcionadas por el **CIEMAT**. Se trata de descargas de plasma nuclear y en ellas, como ya se ha comentado, el objetivo es detectar la presencia de armónicos o mejor dicho de modos multiarmónicos. Los

ficheros de cada descarga están compuestos por dos variables. Una con los valores de energía y otra con los valores temporales. En el análisis se puede llegar a tener casos extremos como el de la figura 30 donde se comprueba la existencia de armónicos pero éstos son tan cortos que es posible que el sistema no los pueda detectar. El total de ficheros a analizar es de 929 repartidos en 28 descargas. Estas descargas están compuestas por seis señales (H3, PP8, PP4, I8, T0 y S40). Cada una de ellas puede contener hasta 7 canales diferentes. Cada fichero tiene un tamaño aproximado entre 10-12 millones de datos.

4.2. Tipología de los armónicos y descripción de los intervalos.

Primeras pruebas.

Para poder contrastar los resultados no se dispone de ningún método previo tal como se ha comentado anteriormente. En su lugar, se utilizan de las medidas que ha realizado el **CIEMAT** de forma manual que consisten en visualizar el espectrograma de la descarga y calcular los intervalos en los que se producen los armónicos.



(a) (b)

Figura 31: (a) espectrograma de la descarga 79456 señal H3, (b) espectrograma de la descarga 79461 señal H3.

Se nombra *intervalo* al espacio temporal donde se ha detectado la presencia de un modo y que a su vez cuenta con un mínimo de un armónico.

Todos los fenómenos de este tipo tienen una composición determinada. En la mayor parte de los casos se tiene un fundamental originado en las frecuencias más bajas. A continuación, y en múltiplos enteros de frecuencias, se detectan uno o dos armónicos.

En la figura 31(a) se comprueba cómo la descarga 79456 dispone de más de tres armónicos. Al contrario de ésta, en la figura 31(b) la descarga 79461 aparentemente no se muestra ningún

armónico. Aún así existe un fundamental en el espacio temporal que va desde 54.094 sg. hasta 54.370 sg.

Parte de las medidas realizadas por el **CIEMAT** son las que aparecen en la figura 32. Son cálculos realizados para la descarga 79455.

Intervalo temporal del fundamental localizados por el experto.		Intervalo temporal del fundamental localizados automáticamente.	
Inicio (sg.)	fin (sg.)	Inicio (sg.)	fin (sg.)
54.010	54.022	53.0269	53.0484
54.073	54.086	53.0682	53.0897
54.095	54.372	53.1872	53.2087
54.376	54.436	54.1268	54.3849
54.436	54.490	54.3872	54.5162
54.562	54.630	54.4515	54.5053
		54.5954	54.6492
		55.0882	55.0989
		55.2806	55.2914
		55.8200	55.8307
		56.5142	56.5895
		56.7266	56.7588

Figura 32: Tabla de medidas detectadas manualmente y automáticamente para la descarga 79455.

Revisando los datos que aparecen en la figura 32 se comprueba cómo la herramienta ha sido capaz de detectar algo en doce intervalos. Parte de estos intervalos son los mismos detectados de forma manual. Concretamente se trata del intervalo entre 54.01 seg. y 54.63 sg.. Según estos datos el porcentaje de éxito sería del 100%. En la página siguiente se muestra la tabla de la figura 33 donde se indica los números de descarga junto con el número de intervalos detectados de forma manual y el número de intervalos detectados de forma automática.

Número. Descarga	Número Intervalos manual	Número Intervalos aut.	Intervalo completo manual (sg.)		Intervalo ajustado automat. (sg.)	
75410	3	13	53.927	54.176	54.0499	54.1948
75411	3	20	53.986	55.376	53.9029	55.4872
75412	6	4	53.946	54.499	54.0168	54.3214
75414	1	24	55.932	56.180	55.9253	56.1193
75783	1	18	57.104	57.143	57.0674	57.1319
77078	5	17	54.073	54.486	53.8151	54.3800
77919	2	25	46.040	46.337	46.0514	46.3570
77920	3	33	45.907	46.538	45.9119	46.5415
77921	3	15	46.120	46.597	46.1452	46.6469
77925	2	33	46.140	46.340	46.1597	46.3693
77926	2	20	46.120	46.270	46.1334	46.2949
77929	1	18	46.435	46.445	47.2368	47.2626
77931	3	16	46.140	46.597	46.1563	46.6202
78008	2	13	54.136	54.301	52.2084	52.4553
78010	6	25	53.810	54.320	53.7131	54.3540
78011	1	15	53.790	54.257	53.4440	54.1421
78012	3	5	53.707	54.645	53.7660	54.6613
78014	6	9	53.752	54.640	53.8337	54.3980
78017	3	13	53.882	54.226	53.9356	53.9786
78018	7	12	53.892	54.444	53.9965	54.0395
78019	7	7	53.820	54.350	53.9668	54.0313
78130	3	10	54.210	54.475	54.2613	54.5015
79455	6	12	54.010	54.630	54.1268	54.6492
79456	6	5	53.920	54.730	53.0561	54.7566
79458	7	9	53.965	54.413	55.9700	56.0560
79459	6	15	54.120	54.868	54.1594	54.7399
79460	1	26	53.770	53.810	54.3253	54.3597
79461	2	16	54.094	54.370	54.4364	54.4537

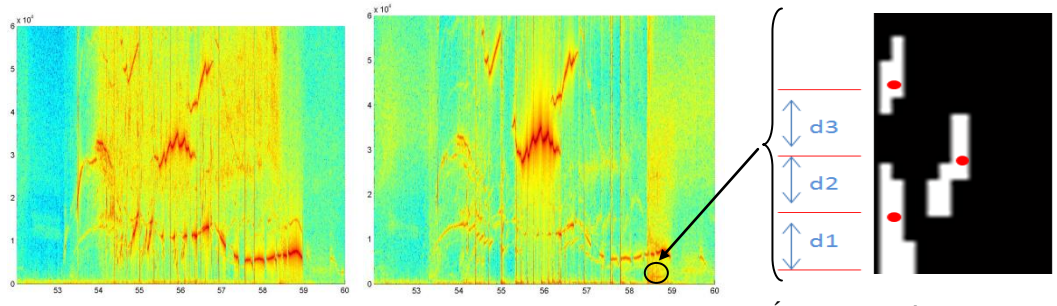
Figura 33: Intervalos encontrados por el medio manual y automático.

No se puede analizar siempre en la misma columna temporal pues si se observa la descarga 77931, encuentra armónicos en el intervalo 46.140 sg. – 46.507 sg. Esto contrasta con la columna predominante en muchas descargas. Es decir se encuentran armónicos a partir de 50 sg.. Este es uno de los motivos que obliga a que el análisis se efectúe en todo el espectrograma. La figura 33 muestra una tabla donde se relacionan datos obtenidos manualmente y datos obtenidos automáticamente por la herramienta. La tabla tiene la información que sigue; la primera columna indica el número de descarga. La segunda y tercera columna indican el número de intervalos encontrados de manual y automática respectivamente. La cuarta columna indica el intervalo completo encontrado de forma manual. La quinta columna indica el mejor ajuste encontrado de forma automática del total de los intervalos encontrados por la herramienta para cada descarga. Es de reseñar que el sistema encuentra más intervalos candidatos pero los intervalos que más se ajustan a los detectados manualmente son los que figuran en la tabla de la figura 33.

Si se hace un breve repaso a la tabla de la figura 33 se puede comprobar el alto porcentaje de ajuste entre los intervalos manuales y automáticos. Esto indica un resultado muy aceptable del automatismo.

Pruebas de negativos.

Hasta aquí las pruebas habían sido realizadas con ficheros de descargas que a priori se tenía certeza de la existencia de modos. Es decir, la aplicación estaba constatando la existencia de modos. No solamente era necesario identificar su existencia sino también localizar con la mayor exactitud los intervalos temporales donde éstos estaban situados.



(a) Señal H3 canal2 (b) Señal I8 canal 2 (c) Área de interés descarga 79461

Figura 34: Descarga con número79461.

Quedaba pues por realizar una segunda prueba que consistía en la confirmación de negativos. En base a un nuevo juego de ficheros de descargas, el objetivo a conseguir era que la aplicación no detectara nada donde en principio se suponía que no había modos. Este segundo conjunto de ficheros constaba de dieciocho descargas, diez menos que el conjunto inicial. Para que los resultados fueran determinantes se procedió a guardar la imagen del espectrograma de todas las señales junto con sus respectivos canales. Una vez revisadas manualmente todas las imágenes de los espectrogramas se procedió al contraste con los datos analíticos obtenidos por la aplicación y el resultado fue satisfactorio. Se debe indicar que se impone a la herramienta la búsqueda muy rigurosa de un fundamental mas dos armónicos. Teniendo en cuenta esto último los porcentajes de éxito para el primer conjunto de descargas fue de un 88.46 %, y para el segundo conjunto de descargas el porcentaje de éxito fue de un 88.88 %, casi el 90%.

Hay que reseñar que la aplicación no recibe ningún parámetro de partida. La figura 34 trata de mostrar lo que se explicará a continuación. En la figura 34(a) se ve el espectrograma de la descarga 79461 de la señal H3 y canal 2. La figura 34(b) muestra la misma descarga pero de la señal I8 donde se puede comprobar que, en el intervalo temporal 58 sg.- 59 sg., existe un área de alta intensidad que no existe en la figura 34(a). Después de aplicar el algoritmo de detección de regiones, el área de alta intensidad resulta quedar con tres regiones detectadas para su posterior estudio. La figura 34(c) muestra que los centroides de estas tres regiones son equidistantes. Es decir $d1=d2=d3$. Esta situación podría confundir y mostrar la existencia de un modo. No obstante se puede comprobar que hay solapamiento entre los ejes verticales de las regiones y este detalle puede ser válido para el posterior descarte de estas regiones. Además la figura 34(c) muestra que la región más cercana al eje de abscisas está unida a este último. Regiones que presentan esta distribución deben ser descartadas pues dan origen a falsos modos. Este tipo de situaciones añaden complejidad pues como se ha comentado anteriormente el proceso comienza sin unos valores de partida. La complejidad aumenta si se tiene en cuenta que el análisis se efectúa en toda la banda frecuencial así como la temporal. Una forma de optimizar el proceso sería focalizar el área de interés en un espacio frecuencial/temporal determinado mediante una parametrización inicial del área de interés.

5. CONCLUSIONES Y TRABAJO FUTURO.

Se han propuesto en este trabajo dos aproximaciones para tratar de resolver el problema de la identificación automática de modos multiarmónicos en espectrogramas.

De las dos técnicas propuestas, una de ellas es más precisa que la otra. Además otro detalle a tener en cuenta es la rapidez de procesamiento pues la carga computacional es alta.

La principal aportación que realiza el trabajo consiste en la combinación de técnicas de tratamiento de imágenes y presenta una buena base de partida para posteriores experimentos en este campo.

Fueron varios los problemas encontrados durante el estudio de los diferentes métodos aplicados a la resolución del problema. Sin embargo uno de los principales fue la dificultad de las imágenes y la localización del área de estudio. En lo referente al área de estudio, la principal dificultad es que ésta no se encuentra en la misma zona temporal.

Otro detalle a tener en cuenta es la pérdida de información de las imágenes a tratar. Este es un problema relevante pues aún con el mejor método siempre, de alguna u otra forma, se pierde información de la imagen. El área de estudio normalmente se centra en zonas de muy pocos pixel. En algunas ocasiones éstas regiones están constituidas por cinco pixel, esto muestra el nivel de dificultad existente.

En lo referente a la dificultad de los espectrogramas basta con ver la complejidad que representa incluso para el cerebro humano. Es necesario fijarse detenidamente para identificar donde puede encontrarse un modo. Si se añade la dificultad de procesar automáticamente miles de ficheros, se podrá considerar la envergadura del problema que se está tratando.

La velocidad de cálculo para grandes volúmenes de datos es importante. En este sentido se hicieron pruebas de programación paralela con Matlab, no obstante los resultados no se llegaron a implementar en la aplicación. Esta podría ser una línea a seguir como trabajo futuro de este proyecto.

6. ANEXO I.

6.1. Introducción.

La aplicación que se ve a continuación, se ha codificado en una vez realizadas todas las pruebas con los diferentes métodos propuestos. El lenguaje utilizado es Matlab. El algoritmo codificado es el de detección de regiones en imágenes. Los motivos han sido por rapidez en la detección y por los resultados obtenidos.

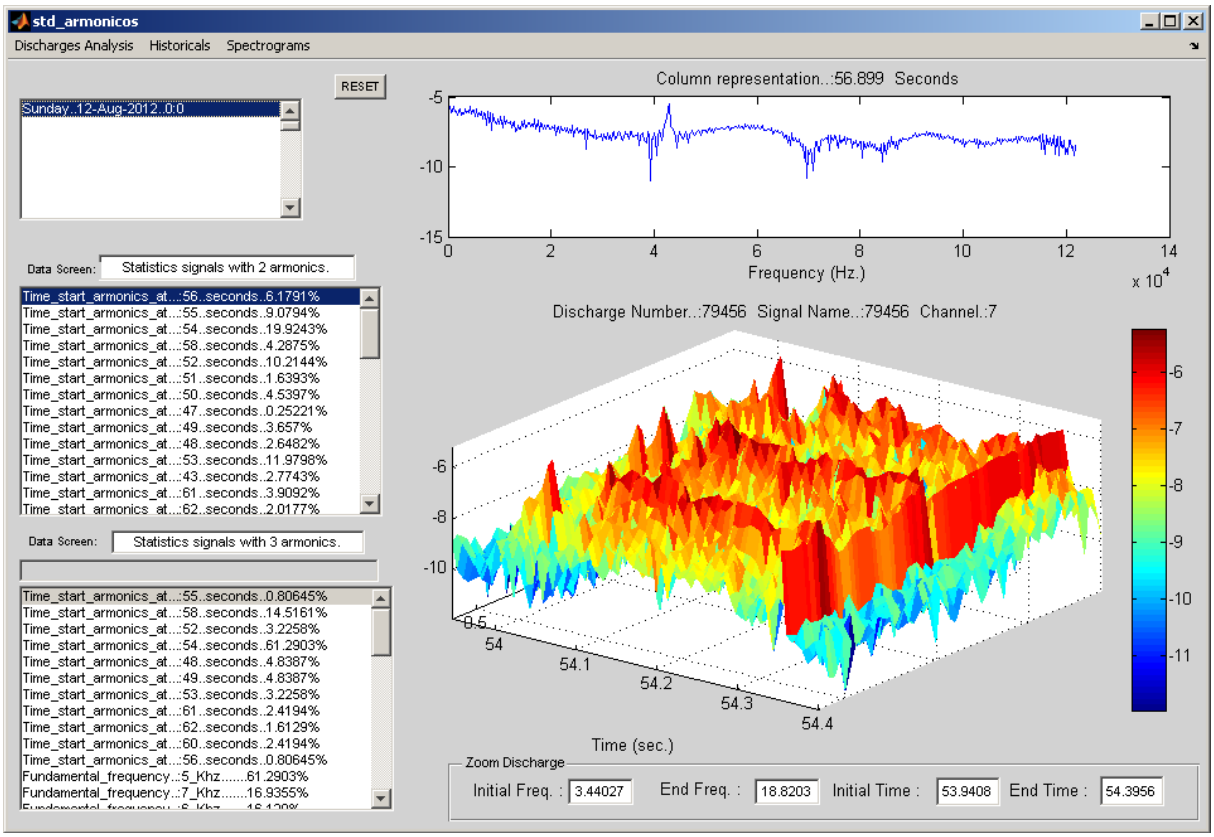


Figura 35: Consola principal del sistema std_armónicos.

La pantalla o consola principal del programa es la que se muestra en la figura 35. Se pueden apreciar a priori dos partes muy diferenciadas. A la izquierda están las ventanas donde se presentan todos los datos analíticos que son el resultado de los diferentes análisis. A la derecha, todo el entorno gráfico donde se pueden analizar descargas concretas así como áreas de más interés. La aplicación ofrece diferentes prestaciones que se exponen a continuación.

6.2. Discharges Analysis.

Este Menú es el que aparece en la figura 36. Permitirá las siguientes acciones:

Start Analysis: mediante esta opción se arranca todo el proceso de análisis de descargas. El proceso se ejecuta de forma dedicada. Esto supone que mientras se esté ejecutando el proceso, no se podrá hacer nada con la sesión Matlab en curso. En un futuro, se verá la posibilidad de ejecutar threads y de esta forma la sesión podrá quedar libre para realizar otras tareas. Con la versión actual del sistema se debería abrir otra sesión si es necesario realizar otras operaciones con la consola.

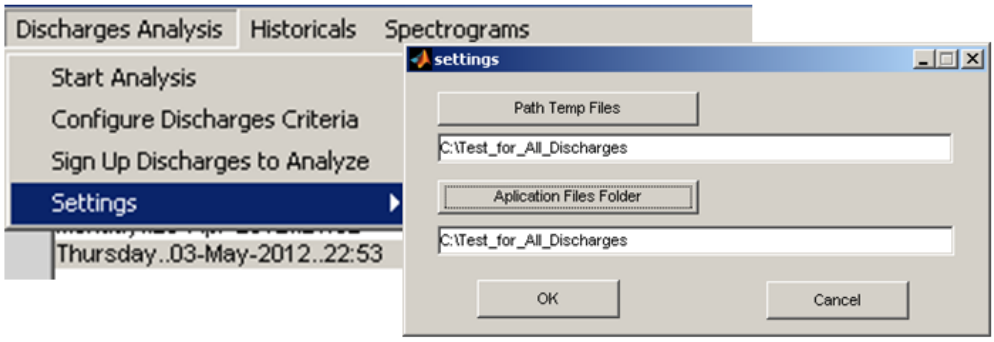


Figura 36: Menú Análisis de descargas.

Configure Discharges Criteria: se pueden seleccionar las señales que se precise analizar. Como ya se ha visto en los capítulos precedentes, cada señal tiene a su vez diferentes canales. Mediante esta opción se pueden seleccionar las señales, así como los canales a tratar. La ventana que se abre cuando se selecciona esta opción, es la que aparece en la figura 39(a).

Sign Up Discharges to Analyze: esta opción permite dar de alta números de descargas analizar. Las descargas se almacenan en un fichero de plano. También existe la posibilidad de cambiar el fichero por otro siempre y cuando el formato no cambie.

Settings: la aplicación necesita espacio en disco para almacenar datos temporales. Para disponer de una forma ordenada los datos es aconsejable tenerlos localizados. Esto se puede hace mediante la opción **Path temp files.** Además de esto último, la aplicación necesita ubicar los ficheros con la información obtenida resultado del análisis. También aquí se puede parametrizar mediante la opción **Aplication folder files**. Si no se inician estas rutas, la

aplicación tomara por defecto carpetas que creará en la carpeta donde este instalada la aplicación. La carpeta por defecto se llamará LOGS.

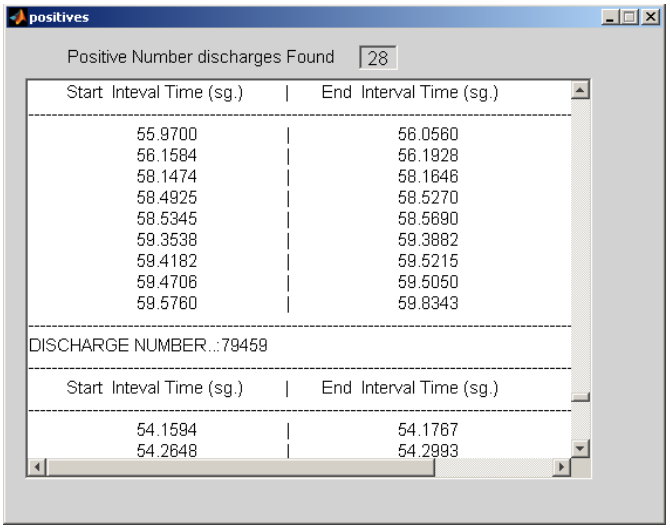
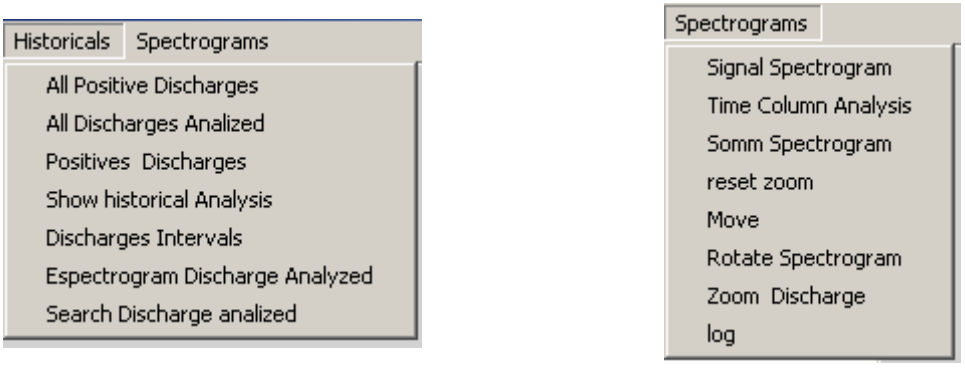


Figura 37: All positives Discharges.



(a) (b)

Figura 38: (a) menú de históricos, (b) menú de Espectrogramas

6.3. Historicals Menu.

All Positive Discharges: la figura 37 muestra la ventana que aparece al pulsar esta opción, que da la posibilidad de ver de una forma rápida todas las descargas en las que se ha encontrado algo. Muestra el intervalo temporal de comienzo y final de lo que ha encontrado.

All Discharges Analized: mediante esta opción se puede ver todas las descargas analizadas. Tanto en las que se ha encontrado algo como en las que no ha sido así.

Positives Discharges: con esta selección se vusualizarán los números de descarga que han dado positivos.

Discharges Intervals: además de la ventana anterior donde se indica de forma rápida todo lo que se ha encontrado, existe una alternativa y es poder ver las señales asi como los canales donde se ha encontrado. Esto lo muestra la figura 39(b).

Show historicals analysis. esta opción muestra los diferentes ejecuciones de análisis de descargas que se han realizado. Automáticamente aparecerán en las ventanas de la consola los porcentajes de los diferentes intervalos que se han encontrado. Esto se muestra para dos y tres armonicos encontrados.

Dischages Intervals: en esta opción se muestran las descargas que han sido analizadas en dichos procesos. Si se selecciona una de dichas descargas y se pulsa la opción "Espectrogram Discharge Analyzed", aparecerá el espectrograma de dicha descarga pero con los intervalos resaltados con bandas verticales de color negro. De esta forma, se ve claramente donde el sistema ha detectado algo. La figura 39(b) así lo muestra.

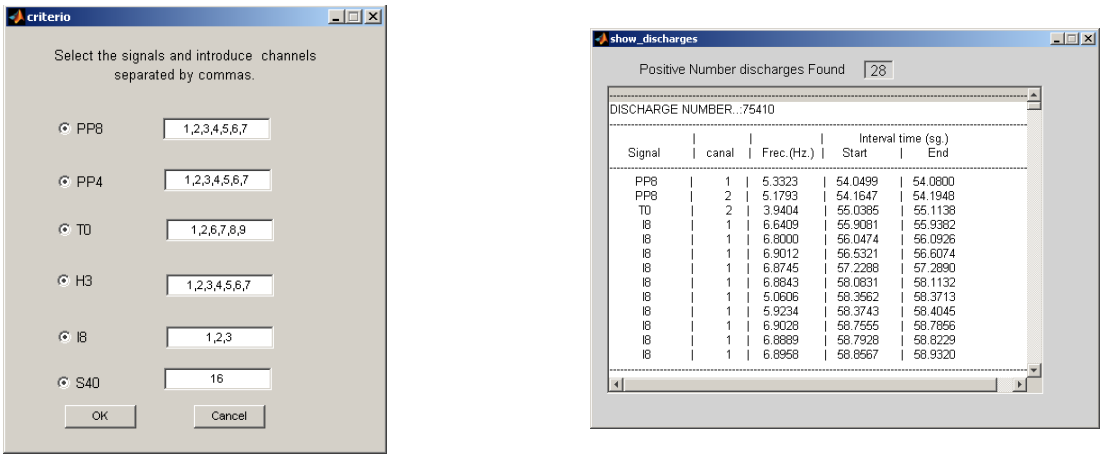
Spectrogram Discharge Analized: Si se selecciona el número de cualquier descarga mostrada mediante la selección *positives discharges* y después se pulsa esta opción, se calculará el espectrograma de la señal que mas armónicos haya obtenido. Además, para localizar rápidamente el resultado del análisis, se reseña con una líneas verticales negras el lugar donde se encuentra lo que la aplicación ha encontrado. En la figura 40 se puede ver un ejemplo de lo que se acaba de comentar.

6.4. Spectrogram Menu.

Signal spectrogram: permite calcular y mostrar el espectrograma de una señal genérica.

Time column analysis: permite extraer una columna de tiempo del espectrograma. Ésta se mostrará en la parte superior de la consola.

Zoom Discharge: existe la posibilidad de analizar una zona muy concreta del espectrograma. Mediante esta opción se puede hacer zoom de la zona acotada por medio de los campos *Frec. Inic., Frec. Final, Tiempo Inic. y Tiempo fin.* El resultado es una figura tridimensional. Posteriormente se puede rotar mediante las funciones del spectrogram menú. Un ejemplo del zoom se puede ver en la figura 35 que muestra la consola principal.



(a) (b)

Figura 39: (a) criterio de análisis, (b) intervalos temporales de armónicos.

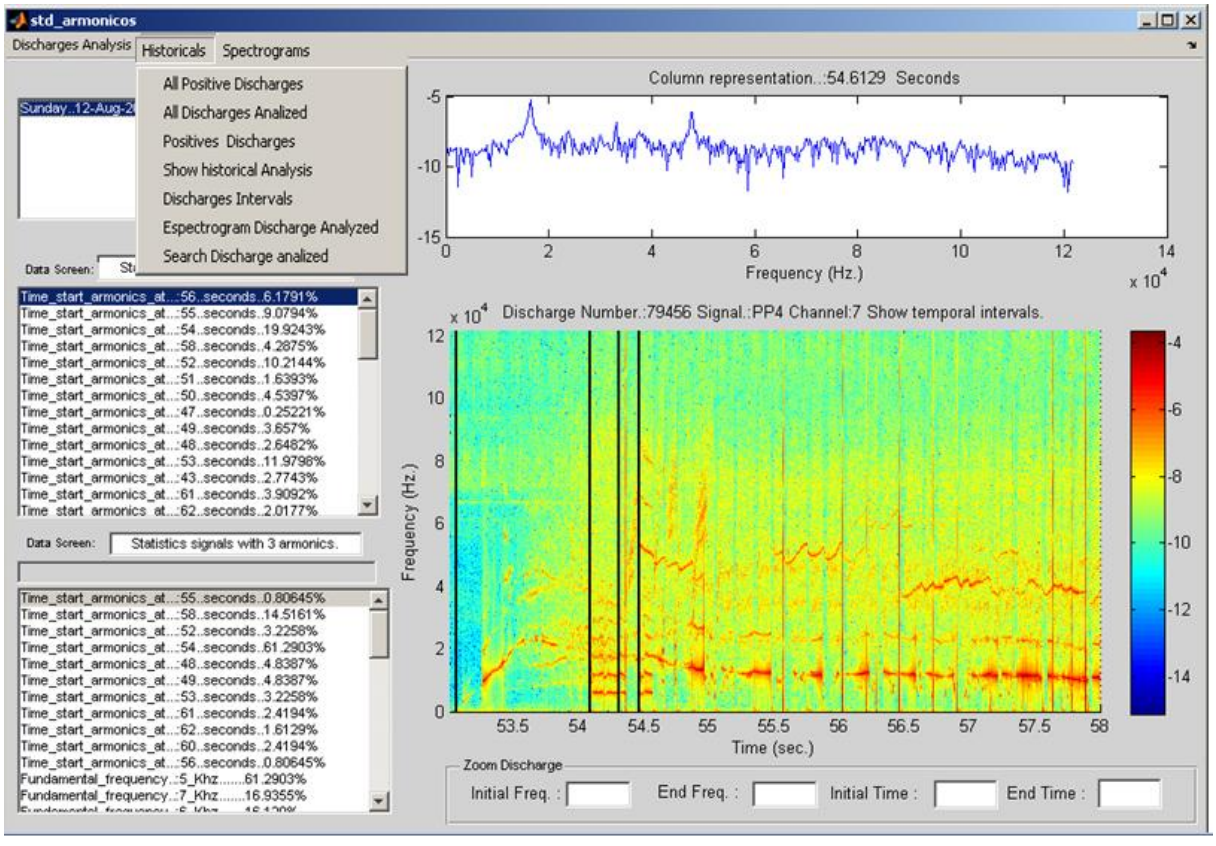


Figura 40: Menú de históricos.

7. ANEXO II.

7.1. Código de la aplicación.

```
% Aplicación que analiza señales provenientes de descargas de plasma
% nuclear. El objetivo es mostrar todos los intervaloes donde es posible
% que se ecuentren modos multiarmonicos. Ademas de localizar  los modos es
% posible mostrar el espectrograma de la señal asi como los intervalos
% donde éstos se encuatran.
function varargout = std_armonicos(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @std_armonicos_OpeningFcn, ...
                   'gui_OutputFcn',  @std_armonicos_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
function std_armonicos_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;
guidata(hObject, handles);
function varargout = std_armonicos_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
global carpeta_actual rotar;
rotar=false;
carpeta_actual=pwd;
disp(carpeta_actual);
% --------------------------------------------------------------------
function Arranque_analisis_Callback(hObject, eventdata, handles)
search_arm1;
% --------------------------------------------------------------------
function Ver_fichero_frecini_Callback(hObject, eventdata, handles)
function configurar_arranque_Callback(hObject, eventdata, handles)
criterio;
function configuracion_1_Callback(hObject, eventdata, handles)
set(handles.listbox2,'String',' ');
set(handles.listbox3,'String',' ');
set(handles.listbox4,'String',' ');
set(handles.datos2,'String',' ');
set(handles.datos4,'String',' ');
set(handles.datos7,'String',' ');
function ver_historico_analisis_Callback(hObject, eventdata, handles)
ruta=pwd;
fich_log=fullfile(ruta,'ficheros_cfg.txt');
fid4=fopen(fich_log,'r');
[ruta_logs]=fgetl(fid4);  % leo dos veces orque en el sgunda linea esta el ficheros para
intervalos
fclose(fid4);
fichero_log3=strcat(ruta_logs,'\LOG_GEN.txt');
fid=fopen(fichero_log3,'r');
i=1;
matriz=[];
[datos]=fgetl(fid);
datoss=struct([]);
global matriz;
matriz =cell(50,1);
i=1;
if ischar(datos)
            while  datos>0
```

```
                        if findstr(datos,'300')==1
                                [datos]=fgetl(fid);
                                [fecha] = strread(datos,'%s');
                                 matriz(i)=cellstr(fecha);
                                i=i+1;
                        end
                        [datos]=fgetl(fid);
                        if ~ischar(datos),   break,   end
            end
  end; %del if
  fclose(fid);
 matriz2=cell(1,1);
 matriz2(1)= matriz(1);
 clc;
 for i=1:20
     kk(i)=cellstr('as');
 end;
 set(handles.listbox3,'String',matriz,'Value',1);
function ver_estadisticas_Callback(hObject, eventdata, handles)
function Estadisticas_II_armonicos_Callback(hObject, eventdata, handles)
global fecha_find;
global matriz;
matriz =cell(50,1);
%
ruta=pwd;
fich_log=fullfile(ruta,'ficheros_cfg.txt');
fid4=fopen(fich_log,'r');
[ruta_logs]=fgetl(fid4);  % leo dos veces orque en el sgunda linea esta el ficheros para
intervalos
fclose(fid4);
fichero_log3=strcat(ruta_logs,'\LOG_GEN.txt');
%
fid=fopen(fichero_log3,'r');
i=1;
matriz2 =cell(50,1);
[datos]=fgetl(fid);
i=1;
bucle=true;
if ischar(datos)
            while  datos>0
                        % estoy buscando la fecha
                        if findstr(datos,'300')==1
                                [datos]=fgetl(fid);
                                [fecha] = strread(datos,'%s');
                                 ok=strcmp(fecha,fecha_find);
                                 if ok,
                                    while datos>0
                                        if findstr(datos,'500')==1
                                           while  bucle
                                           [datos]=fgetl(fid);
                                                if findstr(datos,'510')==1
bucle=false,break,end
                                                    % [fecha] = strread(datos2,'%s');
                                                    [fecha] = strread(datos,'%s');
                                                    matriz2(i)=cellstr(fecha);
                                                    i=i+1;
                                            end
                                        end
                                        [datos]=fgetl(fid);
                                        if (~ischar(datos)) | (~bucle),   break,   end
                                    end
                                end
                        end
                        [datos]=fgetl(fid);
                        if (~ischar(datos)) | (~bucle),   break,   end
            end
  end; %del if
  fclose(fid);
set(handles.listbox2,'String',matriz2,'Value',1);
set(handles.datos2,'String','Señales con 2 armonicos','Value',1);
function Estadisticas_III_armonicos_Callback(hObject, eventdata, handles)
```

```
global fecha_find;
global matriz;
 matriz =cell(50,1);
%
 ruta=pwd;
fich_log=fullfile(ruta,'ficheros_cfg.txt');
fid4=fopen(fich_log,'r');
[ruta_logs]=fgetl(fid4);  % leo dos veces orque en el sgunda linea esta el ficheros para
intervalos
fclose(fid4);
fichero_log3=strcat(ruta_logs,'\LOG_GEN.txt');
%
fid=fopen(fichero_log3,'r');
i=1;
matriz2 =cell(50,1);
[datos]=fgetl(fid);
i=1;
bucle=true;
if ischar(datos)
            while  datos>0
                        % estoy buscando la fecha
                        if findstr(datos,'300')==1
                                [datos]=fgetl(fid);
                                [fecha] = strread(datos,'%s');
                                 ok=strcmp(fecha,fecha_find);
                                 if ok,
                                       [datos]=fgetl(fid);
                                       [datos]=fgetl(fid);
                                       [datos]=fgetl(fid);
                                       [datos]=fgetl(fid);
                                       [datos]=fgetl(fid);
                                    if findstr(datos,'400')==1
                                        while  bucle
                                           [datos]=fgetl(fid);
                                            if findstr(datos,'410')==1 bucle=false,break,end
                                           [fecha] = strread(datos,'%s');
                                           matriz2(i)=cellstr(fecha);
                                           i=i+1;
                                        end
                                    end
                                end
                        end
                        [datos]=fgetl(fid);
                        if (~ischar(datos)) | (~bucle),   break,   end
            end
  end; %del if
  fclose(fid);
set(handles.listbox2,'String',matriz2,'Value',1);
set(handles.datos2,'String','Señales con 3 armonicos','Value',1);
function Espectrograma_frecini_analizada_Callback(hObject, eventdata, handles)
global fecha_find descarga senal canal;
global matriz;
try,
    index_selected_arm = get(handles.listbox4,'Value');
    file_list_arm = get(handles.listbox4,'String');
    texto=char(file_list_arm(index_selected_arm,1))
    lon=length(texto);
    pos=findstr(texto,'....');
    descarga=texto(1:pos(1)-1);
    senal=texto(pos(1)+4:pos(2)-1);
    canal=texto(pos(2)+4:lon)
    file=strcat(canal,'-',descarga,'.mat');
    matriz =cell(50,1);
    %
    ruta=pwd;
    fich_log=fullfile(ruta,'ficheros_cfg.txt');
    fid4=fopen(fich_log,'r');
    [ruta_logs]=fgetl(fid4);  % leo dos veces orque en el sgunda linea esta el ficheros para
intervalos
    fclose(fid4);
    fichero_log3=strcat(ruta_logs,'\LOG_GEN.txt');
```

```
    %
    fid=fopen(fichero_log3,'r');
    i=1;
    matriz2 =cell(50,1);
    [datos]=fgetl(fid)
    i=1;
    bucle=true;
    if ischar(datos)
     while  datos>0
         if findstr(datos,'300')==1
              [datos]=fgetl(fid);
               [fecha] = strread(datos,'%s');
               ok=strcmp(fecha,fecha_find);
                if ok,
                    [datos]=fgetl(fid);
                    selecciona=true;
                    pathname=fullfile(char(datos),descarga,senal);
                    set(handles.tiempoini,'string',senal);
                    set(handles.tiempofin,'string',canal);
                    set(handles.frecini,'string',descarga);
                    h = waitbar(0,'Espere por favor...');
                    waitbar(20/100);
                    edit2=fullfile(pathname,file);
                    edit5=file;
                    selecciona=true;
                    fichero_imagen_temp=strcat(pathname,'\img_temp.jpg');
                    load (edit2,'Y','X');
                    waitbar(40/100);
                    win=10000;
                    noverlap=3000;
                    Fs=1e+6;
                    num_ptos=4096;
                    [y,f,t,p]=spectrogram(Y,win,noverlap,num_ptos,Fs,'yaxis');
                    waitbar(60/100);
                    [fila,col]=size(y);
                    matriz=log10(abs(p)+eps);
                    ff=f(1:500);                                    %ZOOM  de la escala de
frecuencias
                    matrizB=matriz(1:500,:);
                    waitbar(80/100);
                    waitbar(100/100);
                    close(h);
                    xlmax=length(X);
                    [fila,colu]=size(t);
                    tiempo=[];
                    for a=1:colu
                            tiempo(a)=X(fix((xlmax*a)/colu));
                    end
                    axes(handles.axes2);
                    surf(tiempo,ff,matrizB,'EdgeColor','none');
                    axis xy; axis tight; colormap(jet); view(0,90);
                    ylabel('Frecuencia en Hz.');xlabel('Tiempo en sg.');colorbar
                    texto=strcat('Numero Descarga..:',descarga,'  Nombre de
señal..:',descarga,'  Canal.:',canal);
                    title(texto);
                    guidata(hObject,handles);
                    clear('matriz','Y','ff');
                end;
              end;
              [datos]=fgetl(fid);
              if ~ischar(datos),   fclose(fid); break,   end
         end;
  end; %del if
catch,
            [msgstr, msgid] = lasterr;
            err=lasterror;
            st1 = err.stack(1,1);
            texto=strcat(msgstr,msgid);
            errordlg('Ver Descarga',texto);
end;
% --------------------------------------------------------------------
```

```
function espectrograma_de_senal_Callback(hObject, eventdata, handles)
global matrizB tiempo ff y f t p selecciona tiempo pathname filename;
global  descarga senal canal;
selecciona=true;
    [filename, pathname] = uigetfile( ...
      {'*.*', 'TODOS Ficheros-ASCII (*.*)'; ...
           '*.*','TODOS los Ficheros (*.*)'}, ...
    'Seleccione  la descarga a analizar');
    if isequal(filename,0)
          return;
    end
    c1=(findstr(pathname,'\'));
    lon=length(c1)
    senal=pathname(c1(lon-1)+1:c1(lon)-1);
    canal=filename(1:1);
    descarga=pathname(c1(lon-2)+1:c1(lon-1)-1);
    h = waitbar(0,'Leyendo datos, espere por favor...');
    waitbar(20/100);
    edit2=fullfile(pathname,filename);
    edit5=filename;
    selecciona=true;
    fichero_imagen_temp=strcat(pathname,'\img_temp.jpg');
    load (edit2,'Y','X');
    win=10000;
    noverlap=3000;
    Fs=1e+6;
    num_ptos=4096;
    [y,f,t,p]=spectrogram(Y,win,noverlap,num_ptos,Fs,'yaxis');
    delete(h);
    h = waitbar(0,'Ajustando datos calculados, espere por favor...');
    waitbar(60/100);
    [fila,col]=size(y);
    matriz=log10(abs(p)+eps);
    ff=f(1:500);                                    %ZOOM  de la escala de frecuencias
    matrizB=matriz(1:500,:);
    close(h);
    xlmax=length(X);
    [fila,colu]=size(t);
    tiempo=[];
     for a=1:colu
            tiempo(a)=X(fix((xlmax*a)/colu));
     end
    axes(handles.axes2);
    surf(tiempo,ff,matrizB,'EdgeColor','none');
    axis xy; axis tight; colormap(jet); view(0,90);
    ylabel('Frecuencia en Hz.');xlabel('Tiempo en sg.');colorbar
    texto=strcat('Numero Descarga..:',descarga,'  Nombre de señal..:',senal,'
Canal.:',canal);
    title(texto);
    guidata(hObject,handles);
    clear('matriz','Y','ff');
function analizar_columna_tiempo_Callback(hObject, eventdata, handles)
global matrizB y f t p selecciona tiempo;
if selecciona==false
         errordlg('Debe generar previamente el espectrograma de la señal.');
    return;
end
try,
[x,y]=ginput(1);
[fi,col]=size(matrizB);
x=fix(x*1000);
x=x-52036;
long_panta=6942;  % esta es la long en pixel de columnas de la pantalla de espectrograa
 colum=fix( (x*col)/long_panta);
 colstr=num2str(colum);
 columna=matrizB(:,colum);
 frecuen=f(1:500);
 axes(handles.axes3);
 plot(columna,'XData',frecuen);
 save('c:\pepe.mat','columna');
 xlabel('Frecuencia en Hz.');
```

```
 segundos=tiempo(colum);
 texto=strcat('Representacion de columna..:',num2str(segundos),'  Segundos');
 title(texto);
 guidata(hObject,handles);
 clear('colstr','texto','columna');
catch,
    errordlg('La columna seleccionada debe estar en el interior del espectrograma. Seleccione
otra columna.');
end
function zoom_on_Callback(hObject, eventdata, handles)
zoom on;
%
function zoom_off_Callback(hObject, eventdata, handles)
zoom off;
%
function reset_zoom_Callback(hObject, eventdata, handles)
%
function Analisis_descargas_Callback(hObject, eventdata, handles)
%
function Historicos_2_Callback(hObject, eventdata, handles)
%
function NUEVO_Callback(hObject, eventdata, handles)
global descarga senal canal pathname filename matrizB tiempo ff ;
    frecfin=str2num(get(handles.frecfin,'string'));
    frecini=str2num(get(handles.frecini,'string'));
    tiempoini=str2num(get(handles.tiempoini,'string'));
    tiempofin=str2num(get(handles.tiempofin,'string'));
     f1=find(ff>frecini);
     f2=find(ff>frecfin);
     tt1=find(tiempo>tiempoini);
     tt2=find(tiempo>tiempofin);
     tiempo1=tiempo(tt1(1):tt2(1));
     ff2=ff(f1(1):f2(1));
     matrizC=matrizB(f1(1):f2(1),tt1(1):tt2(1));
     axes(handles.axes2);
    surf(tiempo1,ff2,matrizC,'EdgeColor','none');
    axis xy;
    axis tight;
    colormap(jet);
    ylabel('Frecuencia en Hz.');xlabel('Tiempo en sg.');colorbar
    texto=strcat('Numero Descarga..:',descarga,'  Nombre de señal..:',descarga,'
Canal.:',canal);
    title(texto);
    guidata(hObject,handles);
    clear('matriz','Y','ff');
function pushbutton1_Callback(hObject, eventdata, handles)
function pushbutton2_Callback(hObject, eventdata, handles)
function listbox3_Callback(hObject, eventdata, handles)
global matriz;
global fecha_find;
index_selected = get(handles.listbox3,'Value');
file_list = get(handles.listbox3,'String');
clc;
fecha_find=file_list(index_selected,1);
matriz =cell(50,1);
%
ruta=pwd;
fich_log=fullfile(ruta,'ficheros_cfg.txt');
fid4=fopen(fich_log,'r');
[ruta_logs]=fgetl(fid4);  % leo dos veces orque en el sgunda linea esta el ficheros para
intervalos
fclose(fid4);
fichero_log3=strcat(ruta_logs,'\LOG_GEN.txt');
%
fid=fopen(fichero_log3,'r');
i=1;
matriz2 =cell(50,1);
matriz3=cell(50,1);
[datos]=fgetl(fid)
i=1;
bucle=true;
```

```
if ischar(datos)
            while  datos>0
                        % estoy buscando la fecha
                        if findstr(datos,'300')==1
                                [datos]=fgetl(fid);
                                [fecha] = strread(datos,'%s');
                                 ok=strcmp(fecha,fecha_find);
                                 if ok,
                                      [datos]=fgetl(fid);
                                       [datos]=fgetl(fid);
                                       [datos]=fgetl(fid);
                                       [datos]=fgetl(fid);
                                       [datos]=fgetl(fid);
                                   if findstr(datos,'400')==1
                                       while  bucle
                                          [datos]=fgetl(fid);
                                           if findstr(datos,'410')==1 bucle=false,break,end
                                          [fecha] = strread(datos,'%s');
                                          matriz2(i)=cellstr(fecha);
                                          i=i+1
                                        end
                                   end
                                   [datos]=fgetl(fid);
                                   bucle = true;
                                   i=1;
                                   while  bucle
                                          [datos]=fgetl(fid);
                                           if findstr(datos,'510')==1 bucle=false,break,end
                                          [fecha] = strread(datos,'%s');
                                          matriz3(i)=cellstr(fecha);
                                          i=i+1
                                   end
                                end
                        end
                        [datos]=fgetl(fid);
                        if (~ischar(datos)) | (~bucle),   break,   end
            end
  end; %del if
  fclose(fid);
set(handles.listbox2,'String',matriz2,'Value',1);
set(handles.listbox4,'String',matriz3,'Value',1);
set(handles.datos4,'String','Señales con 2 armonicos','Value',1);
set(handles.datos2,'String','Señales con 3 armonicos','Value',1);
function listbox3_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function listbox4_Callback(hObject, eventdata, handles)
function listbox4_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function listbox2_Callback(hObject, eventdata, handles)
function listbox2_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function datos4_Callback(hObject, eventdata, handles)
function datos4_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function datos2_Callback(hObject, eventdata, handles)
function datos2_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function positivos_1_Callback(hObject, eventdata, handles)
global fecha_find;
global matriz;
matriz =cell(50,1);
```

```
%
ruta=pwd;
fich_log=fullfile(ruta,'ficheros_cfg.txt');
fid4=fopen(fich_log,'r');
[ruta_logs]=fgetl(fid4);  % leo dos veces orque en el sgunda linea esta el ficheros para
intervalos
fclose(fid4);
fichero_log3=strcat(ruta_logs,'\LOG_GEN.txt');
%
fid=fopen(fichero_log3,'r');
i=1;
matriz2 =cell(50,1);
[datos]=fgetl(fid)
i=1;
bucle=true;
if ischar(datos)
            while  datos>0
                        % estoy buscando la fecha
                        if findstr(datos,'300')==1
                                [datos]=fgetl(fid);
                                [fecha] = strread(datos,'%s');
                                 ok=strcmp(fecha,fecha_find);
                                 if ok,
                                     [datos]=fgetl(fid);
                                     [datos]=fgetl(fid);
                                     fid1=fopen(datos,'r');
                                     i=1;
                                     [datos1]=fgetl(fid1)
                                     [descarga,senal,canal,frec,ppio,final] =
strread(datos1,'%s %s %s %f %f %f');
                                     canalb=canal;

matriz2(i)=cellstr(strcat(descarga,'....',senal,'....',canal));
                                     while  datos1>0
                                                [descarga,senal,canal,frec,ppio,final] =
strread(datos1,'%s %s %s %f %f %f');
                                                if isequal(canalb,canal)
                                                     [datos1]=fgetl(fid1);
                                                else

matriz2(i)=cellstr(strcat(descarga,'....',senal,'....',canal));
                                                      i=i+1;
                                                      [datos1]=fgetl(fid1);
                                                      canalb=canal;
                                                end
                                                if ~ischar(datos1),   fclose(fid1); break,
end
                                     end;  %del while
                                end;
                        end;
                        [datos]=fgetl(fid);
                         if ~ischar(datos),   fclose(fid); break,   end
            end;
  end; %del if
set(handles.listbox4,'String',matriz2,'Value',1);
set(handles.datos4,'String','ficheros con positivos','Value',1);
% --------------------------------------------------------------------
function negativos_1_Callback(hObject, eventdata, handles)
global fecha_find;
global matriz;
matriz =cell(50,1);
%
ruta=pwd;
fich_log=fullfile(ruta,'ficheros_cfg.txt');
fid4=fopen(fich_log,'r');
[ruta_logs]=fgetl(fid4);  % leo dos veces orque en el sgunda linea esta el ficheros para
intervalos
fclose(fid4);
fichero_log3=strcat(ruta_logs,'\LOG_GEN.txt');
%
fid=fopen(fichero_log3,'r');
```

```
i=1;
matriz2 =cell(50,1);
[datos]=fgetl(fid)
i=1;
bucle=true;
if ischar(datos)
            while  datos>0
                        % estoy buscando la fecha
                        if findstr(datos,'300')==1
                                [datos]=fgetl(fid);
                                [fecha] = strread(datos,'%s');
                                 ok=strcmp(fecha,fecha_find);
                                 if ok,
                                     [datos]=fgetl(fid);
                                     [datos]=fgetl(fid);
                                     [datos]=fgetl(fid);
                                       [datos]=fgetl(fid);
                                     fid1=fopen(datos,'r');
                                     i=1;
                                     [datos1]=fgetl(fid1)
                                     [descarga,senal,canal] = strread(datos1,'%s %s %s');

matriz2(i)=cellstr(strcat(descarga,'....',senal,'....',canal));
                                     while  datos1>0
                                                [descarga,senal,canal] = strread(datos1,'%s %s
%s');

matriz2(i)=cellstr(strcat(descarga,'....',senal,'....',canal));
                                                i=i+1;
                                                [datos1]=fgetl(fid1);
                                                if ~ischar(datos1),   fclose(fid1); break,
end
                                     end;  %del while
                                 end;
                        end;
                        [datos]=fgetl(fid);
                         if ~ischar(datos),   fclose(fid); break,   end
            end;
  end; %del if
set(handles.listbox4,'String',matriz2,'Value',1);
set(handles.datos4,'String','ficheros con negativos','Value',1);
function ver_intervalos_Callback(hObject, eventdata, handles)
function frecini_Callback(hObject, eventdata, handles)
function frecini_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function tiempoini_Callback(hObject, eventdata, handles)
function tiempoini_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function tiempofin_Callback(hObject, eventdata, handles)
function tiempofin_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function mover_on_Callback(hObject, eventdata, handles)
pan on;
function mover_off_Callback(hObject, eventdata, handles)
pan off;
function frecfin_Callback(hObject, eventdata, handles)
function frecfin_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function rotar_zoom_descarga_Callback(hObject, eventdata, handles)
global rotar;
if rotar
   rotate3d off;
   rotar=false;
```

```
else
    rotar=true;
    rotate3d on;
end
function II_armonicos_Callback(hObject, eventdata, handles)
global fecha_find;
global matriz;
index_selected_arm = get(handles.listbox4,'Value');
file_list_arm = get(handles.listbox4,'String');
texto=char(file_list_arm(index_selected_arm,1))
lon=length(texto);
pos=findstr(texto,'....');
descarga=texto(1:pos(1)-1);
senal=texto(pos(1)+4:pos(2)-1);
canal=texto(pos(2)+4:lon)
file=strcat(canal,'-',descarga,'.mat');
matriz =cell(50,1);
%
ruta=pwd;
fich_log=fullfile(ruta,'ficheros_cfg.txt');
fid4=fopen(fich_log,'r');
[ruta_logs]=fgetl(fid4);  % leo dos veces orque en el sgunda linea esta el ficheros para
intervalos
fclose(fid4);
fichero_log3=strcat(ruta_logs,'\LOG_GEN.txt');
%
fid=fopen(fichero_log3,'r');
i=1;
matriz2 =cell(50,1);
[datos]=fgetl(fid)
i=1;
bucle=true;
if ischar(datos)
            while  datos>0
                        % estoy buscando la fecha
                        if findstr(datos,'300')==1
                                [datos]=fgetl(fid);
                                [fecha] = strread(datos,'%s');
                                 ok=strcmp(fecha,fecha_find);
                                 if ok,
                                     [datos]=fgetl(fid);
                                     [datos]=fgetl(fid);
                                     fid1=fopen(datos,'r');
                                     i=1;
                                     [datos1]=fgetl(fid1)
                                       [descarga1,senal1,canal1,frec,ppio,final] =
strread(datos1,'%s %s %s %f %f %f');
                                     while  datos1>0
                                                 if isequal(descarga,char(descarga1)) &
isequal(senal,char(senal1))& isequal(canal,char(canal1))

matriz2(i)=cellstr(strcat(descarga1,'...',senal1,'...',canal1,'...Frec.:',...

num2str(frec),'...Inic.:',num2str(ppio),'...Final.:',num2str(final)));
                                                 i=i+1;
                                                 end;
                                                  [datos1]=fgetl(fid1);
                                                   if ~ischar(datos1),   fclose(fid1); break,
end
                                                   [descarga1,senal1,canal1,frec,ppio,final] =
strread(datos1,'%s %s %s %f %f %f');

                                     end;  %del while
                                end;
                        end;
                        [datos]=fgetl(fid);
                         if ~ischar(datos),   fclose(fid); break,   end
            end;
  end; %del if
  set(handles.listbox2,'String',matriz2,'Value',1);
set(handles.datos2,'String','intervalos positivos II armonicos','Value',1)
```

```
function espectrogramas_Callback(hObject, eventdata, handles)
function Espectrograma_descarga_analizada_Callback(hObject, eventdata, handles)
global fecha_find descarga canal senal pathname file tiempo ff matrizB f;
global matriz;
try,
index_selected_arm = get(handles.listbox4,'Value');
file_list_arm = get(handles.listbox4,'String');
texto=char(file_list_arm(index_selected_arm,1))
lon=length(texto);
pos=findstr(texto,'....');
descarga=texto(1:pos(1)-1);
senal=texto(pos(1)+4:pos(2)-1);
canal=texto(pos(2)+4:lon)
file=strcat(canal,'-',descarga,'.mat');
matriz =cell(50,1);
%
ruta=pwd;
fich_log=fullfile(ruta,'ficheros_cfg.txt');
fid4=fopen(fich_log,'r');
[ruta_logs]=fgetl(fid4);  % leo dos veces orque en el sgunda linea esta el ficheros para
intervalos
fclose(fid4);
fichero_log3=strcat(ruta_logs,'\LOG_GEN.txt');
%
fid=fopen(fichero_log3,'r');
i=1;
matriz2 =cell(50,1);
[datos]=fgetl(fid)
i=1;
bucle=true;
if ischar(datos)
  while  datos>0
  % estoy buscando la fecha
       if findstr(datos,'300')==1
         [datos]=fgetl(fid);
         [fecha] = strread(datos,'%s');
          ok=strcmp(fecha,fecha_find);
         if ok,
                 [datos]=fgetl(fid);
                 selecciona=true;
                 pathname=fullfile(char(datos),descarga,senal);
                 h = waitbar(0,'Leyendo datos, espere por favor...');
                 waitbar(20/100);
                 edit2=fullfile(pathname,file);
                edit5=file;
                selecciona=true;
                fichero_imagen_temp=strcat(pathname,'\img_temp.jpg');
                load (edit2,'Y','X');
                delete(h);
                h=waitbar(0,'Calculando Espectrograma, espere por favor...');
                waitbar(40/100);
                win=10000;
                noverlap=3000;
                Fs=1e+6;
                num_ptos=4096;
                [y,f,t,p]=spectrogram(Y,win,noverlap,num_ptos,Fs,'yaxis');
                  waitbar(60/100);
                [fila,col]=size(y);
                matriz=log10(abs(p)+eps);
                ff=f(1:500);                                    %ZOOM  de la escala de
frecuencias
                matrizB=matriz(1:500,:);
                delete(h);
                h=waitbar(0,'Ajustando datos, espere por favor...');
                waitbar(80/100);
                waitbar(100/100);
                 close(h);
                  xlmax=length(X);
                [fila,colu]=size(t);
                tiempo=[];
                 for a=1:colu
```

```
                        tiempo(a)=X(fix((xlmax*a)/colu));
                    end
                    axes(handles.axes2);
                    surf(tiempo,ff,matrizB,'EdgeColor','none');
                    axis xy; axis tight; colormap(jet); view(0,90);
                    ylabel('Frecuencia en Hz.');xlabel('Tiempo en sg.');colorbar
                    texto=strcat('Numero Descarga..:',descarga,'  Nombre de señal..:',descarga,'
Canal.:',canal);
                    title(texto);
                    guidata(hObject,handles);
                    clear('matriz','Y','ff');
                 end;
              end;
              [datos]=fgetl(fid);
              if ~ischar(datos),   fclose(fid); break,   end
            end;
    end; %del if
catch,

                [msgstr, msgid] = lasterr;
              err=lasterror;
              st1 = err.stack(1,1);
              texto=strcat(msgstr,msgid);
              errordlg(texto,'Ver Descarga');
end;


% --------------------------------------------------------------------
function III_armonicos_Callback(hObject, eventdata, handles)
global fecha_find;
global matriz;
try,
        index_selected_arm = get(handles.listbox4,'Value');
        file_list_arm = get(handles.listbox4,'String');
        texto=char(file_list_arm(index_selected_arm,1))
        lon=length(texto);
        pos=findstr(texto,'....');
        descarga=texto(1:pos(1)-1);
        senal=texto(pos(1)+4:pos(2)-1);
        canal=texto(pos(2)+4:lon)
        file=strcat(canal,'-',descarga,'.mat');
        matriz =cell(50,1);
        %
        ruta=pwd;
        fich_log=fullfile(ruta,'ficheros_cfg.txt');
        fid4=fopen(fich_log,'r');
        [ruta_logs]=fgetl(fid4);  % leo dos veces orque en el sgunda linea esta el ficheros
para intervalos
        fclose(fid4);
        fichero_log3=strcat(ruta_logs,'\LOG_GEN.txt');
        %
        fid=fopen(fichero_log3,'r');
        i=1;
        matriz2 =cell(50,1);
        [datos]=fgetl(fid)
        i=1;
        bucle=true;
        if ischar(datos)
            while  datos>0
                        % estoy buscando la fecha
                        if findstr(datos,'300')==1
                                [datos]=fgetl(fid);
                                [fecha] = strread(datos,'%s');
                                 ok=strcmp(fecha,fecha_find);
                                 if ok,
                                     [datos]=fgetl(fid);
                                     [datos]=fgetl(fid);
                                     [datos]=fgetl(fid);
                                     fid1=fopen(datos,'r');
                                     i=1;
                                     [datos1]=fgetl(fid1)
```

```
                                        [descarga1,senal1,canal1,frec,ppio,final] =
strread(datos1,'%s %s %s %f %f %f');
                                    while  datos1>0
                                            if isequal(descarga,char(descarga1)) &
isequal(senal,char(senal1))& isequal(canal,char(canal1))

matriz2(i)=cellstr(strcat(descarga1,'...',senal1,'...',canal1,'...Frec.:',...

num2str(frec),'...Inic.:',num2str(ppio),'...Final.:',num2str(final)));
                                            i=i+1;
                                            end;
                                             [datos1]=fgetl(fid1);
                                              if ~ischar(datos1),   fclose(fid1); break,
end
                                             [descarga1,senal1,canal1,frec,ppio,final] =
strread(datos1,'%s %s %s %f %f %f');

                                    end;  %del while
                                end;
                        end;
                        [datos]=fgetl(fid);
                        if ~ischar(datos),   fclose(fid); break,   end
            end;
  end; %del if
set(handles.listbox2,'String',matriz2,'Value',1);
set(handles.datos2,'String','intervalos positivos III armonicos','Value',1)
catch,

    [msgstr, msgid] = lasterr;
            err=lasterror;
            st1 = err.stack(1,1);
            texto=strcat(msgstr,msgid);
            errordlg('III Armonicos',texto);
end;


% --------------------------------------------------------------------
function con_dos_armonicos_Callback(hObject, eventdata, handles)
global fecha_find;
global matriz;
i=1;
matriz =cell(1000,1);
%
ruta=pwd;
fich_log=fullfile(ruta,'ficheros_cfg.txt');
fid4=fopen(fich_log,'r');
[ruta_logs]=fgetl(fid4);  % leo dos veces orque en el sgunda linea esta el ficheros para
intervalos
fclose(fid4);
fichero_log3=strcat(ruta_logs,'\LOG_GEN.txt');
%
carpeta_actual=pwd;
fid=fopen(fullfile(carpeta_actual,'intervalos','positivas2.txt'),'r');
[datos]=fgetl(fid)
if ischar(datos),
     [descarga] = strread(datos,'%s');
     matriz(i)=cellstr(descarga);
     [datos]=fgetl(fid)
     while ischar(datos)
            [descarga] = strread(datos,'%s');
            if ~strcmp(matriz(i),cellstr(descarga))
                i=i+1;
                matriz(i)=cellstr(descarga);
            end
            [datos]=fgetl(fid)
      end
end
fclose(fid);
set(handles.listbox4,'String',matriz,'Value',1);
set(handles.datos4,'String','Descargas positivas II Arm.','Value',1);
clear('matriz');
```

```
matriz =cell(1000,1);
fid=fopen(fullfile(carpeta_actual,'intervalos','negativas2.txt'),'r');

[datos]=fgetl(fid)
if ischar(datos),
 [descarga] = strread(datos,'%s');
 i=1;
 matriz(i)=cellstr(descarga);
 [datos]=fgetl(fid)
 while ischar(datos)
            [descarga] = strread(datos,'%s');
            if ~strcmp(matriz(i),cellstr(descarga))
                i=i+1;
                matriz(i)=cellstr(descarga);
            end
            [datos]=fgetl(fid)
 end
end
  fclose(fid);
 set(handles.listbox2,'String',matriz,'Value',1);
set(handles.datos2,'String','Descargas Negativas.','Value',1);

% --------------------------------------------------------------------
function Untitled_6_Callback(hObject, eventdata, handles)
function con_III_armonicos_Callback(hObject, eventdata, handles)
global fecha_find;
global matriz;
i=1;
matriz =cell(1000,1);
%
ruta=pwd;
fich_log=fullfile(ruta,'ficheros_cfg.txt');
fid4=fopen(fich_log,'r');
[ruta_logs]=fgetl(fid4);  % leo dos veces orque en el sgunda linea esta el ficheros para
intervalos
fclose(fid4);
fichero_log3=strcat(ruta_logs,'\LOG_GEN.txt');
%
carpeta_actual=pwd;
fid=fopen(fullfile(carpeta_actual,'intervalos','positivas3.txt'),'r');
[datos]=fgetl(fid)
if ischar(datos),
     [descarga] = strread(datos,'%s');
     matriz(i)=cellstr(descarga);
     [datos]=fgetl(fid)
     while ischar(datos)
            [descarga] = strread(datos,'%s');
            if ~strcmp(matriz(i),cellstr(descarga))
                i=i+1;
                matriz(i)=cellstr(descarga);
            end
            [datos]=fgetl(fid)
    end
end
fclose(fid);
set(handles.listbox4,'String',matriz,'Value',1);
set(handles.datos4,'String','Descargas positivas III Arm.','Value',1);
clear('matriz');
matriz =cell(1000,1);
fid=fopen(fullfile(carpeta_actual,'intervalos','negativas2.txt'),'r');
[datos]=fgetl(fid)
if ischar(datos),
 [descarga] = strread(datos,'%s');
 i=1;
 matriz(i)=cellstr(descarga);
 [datos]=fgetl(fid)
 while ischar(datos)
            [descarga] = strread(datos,'%s');
            if ~strcmp(matriz(i),cellstr(descarga))
                i=i+1;
                matriz(i)=cellstr(descarga);
```

```
            end
            [datos]=fgetl(fid)
 end
end
  fclose(fid);
 set(handles.listbox2,'String',matriz,'Value',1);
set(handles.datos2,'String','Descargas Negativas.','Value',1);
%
function intervalos_descargas_Callback(hObject, eventdata, handles)
global fecha_find;
global matriz;
global senal_itv;
global canal_itv;
global ppio_itv;
index_selected_arm = get(handles.listbox4,'Value');
file_list_arm = get(handles.listbox4,'String');
texto=char(file_list_arm(index_selected_arm,1));
%
ruta=pwd;
fich_log=fullfile(ruta,'ficheros_cfg.txt');
fid4=fopen(fich_log,'r');
[ruta_logs]=fgetl(fid4);
[ruta_logs]=fgetl(fid4);  % leo dos veces orque en el sgunda linea esta el ficheros para
intervalos
fclose(fid4);
%
i=1;
fichero=strcat(ruta_logs,'\positivas2.txt');
matriz =cell(100,1);
fid=fopen(fichero,'r');
[datos]=fgetl(fid);
while ischar(datos)
    [descarga1] = strread(datos,'%s');
    matriz2(i)=cellstr(descarga1);
    i=i+1;
    [datos]=fgetl(fid);
end;
matriz=matriz2;
set(handles.listbox2,'String',matriz2,'Value',1);
texto2=strcat('intervalos de II armonicos de..:',texto);
set(handles.datos2,'String','Discharges with 2 armonics ','Value',1)
set(handles.datos7,'String','Select discharge to see spectrogram&intervals','Value',1)
return;
%
function datos7_Callback(hObject, eventdata, handles)
%
function datos7_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function Espectrograma_Descarga_Analizada_Callback(hObject, eventdata, handles)
global fecha_find descarga canal senal pathname file tiempo ff matrizB f;
global matriz;
global matriz2;
global canal_itv;
global senal_itv;
global ppio_itv;
try,
        axes(handles.axes3);
        cla reset;
        axes(handles.axes2);
        cla reset;
        index_selected_arm = get(handles.listbox2,'Value');
        file_list_arm = get(handles.listbox2,'String');
        descarga=char(file_list_arm(index_selected_arm,1))
        ruta=pwd;
        fich_log=fullfile(ruta,'ficheros_cfg.txt');
        fid4=fopen(fich_log,'r');
        [ruta_logs]=fgetl(fid4);
        [ruta_logs]=fgetl(fid4);  % leo dos veces orque en el sgunda linea esta el ficheros
para intervalos
```

```
        fclose(fid4);
        fichero=strcat(ruta_logs,'\itv',descarga,'.txt');
        matriz =cell(100,1);
        % preparo los datos para llegado el caso visualizar el espectrograma y por
        % ello debo saber el nombre de señal y canal donde mejor se ve los
        % intervalos. lo leo otra vez para poder dimensionar cell
        fid=fopen(fichero,'r');
        senal_itv=cell(100,1);
        canal_itv=cell(100,1);
        ppio_itv=[];
        [datos]=fgetl(fid);
        j=1;
        num_reg=0;
        while ischar(datos)
            [descarga1,senal1,canal1,frec,ppio,final] = strread(datos,'%s %s %s %f %f %f');
            %cargo estos valores para despues poder marcar la perpendicular de color negro
donde
            %comienza cada tramo del fundamental
            senal_itv2(j)=senal1;
            canal_itv2(j)=canal1;
            ppio_itv(j)=ppio;
            j=j+1;
            [datos]=fgetl(fid);
            num_reg=num_reg+1;
        end;
        fclose(fid);
        matriz=matriz2;
        senal_itv=cell(num_reg,1);
        canal_itv=cell(num_reg,1);
        senal_itv=senal_itv2(1:num_reg);
        canal_itv=canal_itv2(1:num_reg);
        [a,b,c]=unique(canal_itv)
        [a1,b1,c1]=unique(senal_itv)
        senal=a1(1);
        canal=a(1);
        texto2=strcat('Se mostrará la señal junto con canal que mas calidad ha tenido.
Seleccione carpeta donde esta la descarga ....:'...
                ,descarga);
        directorio=uigetdir('Buscar carpeta',texto2);
        if directorio==0 return; end;
         file=char(strcat(canal,'-',descarga,'.mat'));
        pathname_file=strcat(directorio,'\',descarga,'\',senal,'\',file);
        selecciona=true;
        h = waitbar(0,'Leyendo datos, espere por favor...');
        waitbar(20/100);
        selecciona=true;
        load (char(pathname_file),'Y','X');
        delete(h);
        h=waitbar(0,'Calculando Espectrograma, espere por favor...');
        waitbar(40/100);
        win=10000;
        noverlap=3000;
        Fs=1e+6;
        num_ptos=4096;
        [y,f,t,p]=spectrogram(Y,win,noverlap,num_ptos,Fs,'yaxis');
          waitbar(60/100);
        [fila,col]=size(y);
        matriz=log10(abs(p)+eps);
        ff=f(1:500);                                    %ZOOM  de la escala de frecuencias
        matrizB=matriz(1:500,:);
        delete(h);
        h=waitbar(0,'Ajustando datos, espere por favor...');
        waitbar(80/100);
        waitbar(100/100);
         close(h);
          xlmax=length(X);
        [fila,colu]=size(t);
        tiempo=[];
         for a=1:colu
                tiempo(a)=X(fix((xlmax*a)/colu));
         end
```

```
         cla reset;
        axes(handles.axes2);
        surf(tiempo,ff,matrizB,'EdgeColor','none');
        axis xy; axis tight; colormap(jet); view(0,90);
        ylabel('Frecuencia en Hz.');xlabel('Tiempo en sg.');colorbar
        texto=strcat('Discharge Number.:',char(descarga),' Signal.:',char(senal),'
Chn.:',char(canal),' Show intervals with 2 armonics');
        title(texto);
        hold(gca);
        y=[200 121620];
        cont=length(ppio_itv);
        for u=1:cont
            x(1:2)=ppio_itv(u);
            line(x,y,'Color','k','LineWidth',1.5);
        end
        guidata(hObject,handles);
        clear('matriz','Y','ff');
catch,

            [msgstr, msgid] = lasterr;
            err=lasterror;
            st1 = err.stack(1,1);
            texto=strcat(msgstr,msgid);
            errordlg(texto,'Ver Descarga');
end;
function Registrar_Descargas_Analizar_Callback(hObject, eventdata, handles)
%
function reset_Callback(hObject, eventdata, handles)
set(handles.listbox2,'String',' ');
set(handles.listbox3,'String',' ');
set(handles.listbox4,'String',' ');
set(handles.datos2,'String',' ');
set(handles.datos4,'String',' ');
set(handles.datos7,'String',' ');
axes(handles.axes2);
cla reset;
axes(handles.axes3);
cla reset;
%
function all_positive_discharges_Callback(hObject, eventdata, handles)
positives;
function positive_discharges_Callback(hObject, eventdata, handles)
show_discharges;
function trazar_linea_Callback(hObject, eventdata, handles)
function settings_Callback(hObject, eventdata, handles)
settings;
function path_files_temp_Callback(hObject, eventdata, handles)
function aplication_files_folder_Callback(hObject, eventdata, handles)
function positive_1_Callback(hObject, eventdata, handles)
```

Procedimiento *settings.m*

```
% Este programa realiza la seleccion de las carpetas para almacenar los
% ficheros temporales y logs de la herramienta.
function varargout = settings(varargin)
%
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @settings_OpeningFcn, ...
                   'gui_OutputFcn',  @settings_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
```

```
    gui_mainfcn(gui_State, varargin{:});
end
%
function settings_OpeningFcn(hObject, eventdata, handles, varargin)
%
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
global path;
global aplication;
carpeta_actual=pwd;
fichero_criterio=fullfile(carpeta_actual,'ficheros_cfg.txt');
fid2=fopen(fichero_criterio,'r');
[path]=fgetl(fid2);
if path==-1 path=' '
else [aplication]=fgetl(fid2);
     if aplication==-1  aplication=' '; end
end;
fclose(fid2);
set(handles.temp_files,'String',path);
set(handles.aplication_files,'String',aplication   );
function varargout = settings_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
function temp_files_Callback(hObject, eventdata, handles)
function temp_files_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function aplication_files_Callback(hObject, eventdata, handles)
%
function aplication_files_CreateFcn(hObject, eventdata, handles)
%
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes on button press in aplication.
function aplication_Callback(hObject, eventdata, handles)
global aplication;
aplication=uigetdir;
 set(handles.aplication_files,'String',aplication   );
%
function path_Callback(hObject, eventdata, handles)
%
global path;
path=uigetdir;
 set(handles.temp_files,'String',path);
function ok_Callback(hObject, eventdata, handles)
global aplication;
global path;
ruta=pwd;
fichero=fullfile(ruta,'ficheros_cfg.txt');
fid=fopen(fichero,'w');
fprintf(fid,'%-200s\r\n',path);
fprintf(fid,'%-200s\r\n',aplication);
fclose(fid);
delete(handles.figure1);
function cancel_Callback(hObject, eventdata, handles)
delete(handles.figure1);
```

Procedimiento *show_discharges.m*

```
% programa que visualiza todos los intervalos encontrados en todas las
% descargas analizadas
function varargout = show_discharges(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @show_discharges_OpeningFcn, ...
                   'gui_OutputFcn',  @show_discharges_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
```

```
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
%
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
function show_discharges_OpeningFcn(hObject, eventdata, handles, varargin)
%
handles.output = hObject;
guidata(hObject, handles);
global fecha_find;
global matriz;
global senal_itv;
global canal_itv;
global ppio_itv;
global num_des;
num_des=0;
linea(1:120)='-';
k=1;
total=struct('descarga',{});
ruta=pwd;
fich_log=fullfile(ruta,'ficheros_cfg.txt');
fid4=fopen(fich_log,'r');
[ruta_logs]=fgetl(fid4);
[ruta_logs]=fgetl(fid4);  % leo dos veces orque en el sgunda linea esta el ficheros para
intervalos
fclose(fid4);
ficherogen=strcat(ruta_logs,'\positivas2.txt');
%ficherogen=fullfile(ruta_logs,'positivas2.txt');
fidgen=fopen(ficherogen,'r');
[datosgen]=fgetl(fidgen);
% bucle general
while ischar(datosgen)
   [texto] = char(strread(datosgen,'%s'));
    num_des=num_des+1;
    fichero=strcat(ruta_logs,'\itv',texto,'.txt');
    %fichero=fullfile(pwd,'intervalos',strcat('itv',texto,'.txt'));
    fid=fopen(fichero,'r');
    total(k).descarga=linea;
    k=k+1;
    total(k).descarga=strcat('DISCHARGE NUMBER..:',texto);
    k=k+1;
    total(k).descarga=linea;
    k=k+1;
    total(k).descarga='                          |               |                    |
Interval time (sg.)  ';
    k=k+1;
    total(k).descarga='     Signal         |   canal    |   Frec.(Hz.)   |     Start          |
End  ';
    k=k+1;
    total(k).descarga=linea;
    k=k+1;
    [datos]=fgetl(fid);
    while ischar(datos)
        punt=1;
         [descarga1,senal1,canal1,frec,ppio,final] = strread(datos,'%s %s %s %f %f %f');
        [descarga2,senal2,canal2,frec2,ppio2,final2] = strread(datos,'%s %s %s %5s %5s %5s');
        l=length(char(senal1));
        lon_frec=length(num2str(frec));
        lon_ppio=length(num2str(ppio));
        if l==2
            if strcmp(senal1,'I8')
                total(k).descarga(punt:punt+10)=sprintf('%11s',char(senal1));
                punt=punt+11;
            else
                total(k).descarga(punt:punt+9)=sprintf('%10s',char(senal1));
                punt=punt+10;
```

```
            end
            total(k).descarga(punt:punt+11)='           |';
            punt=punt+12;
            total(k).descarga(punt:punt+9)=sprintf('%10s',char(canal1));
            punt=punt+10;
            total(k).descarga(punt:punt+4)='    |';
            punt=punt+5;
            total(k).descarga(punt:punt+9)=sprintf('%10s',num2str(frec,'%1.4f'));
            punt=punt+10;
            total(k).descarga(punt:punt+7)='       |';
            puntero=punt+7;
        else
            total(k).descarga(punt:punt+9)=sprintf('%10s',char(senal1));
            punt=punt+10;
            total(k).descarga(punt:punt+9)='         |';
             punt=punt+10;
            a=sprintf('%10s',char(canal1));
            total(k).descarga(21:30)=a;
            total(k).descarga(31:35)='    |';
            total(k).descarga(36:45)=sprintf('%10s',num2str(frec,'%1.4f'));
            if lon_frec==5
                total(k).descarga(46:56)='        |  ';
                puntero=56;
            else
                total(k).descarga(46:53)='       |';
                puntero=53;
            end
        end
        total(k).descarga(puntero+1:puntero+10)=sprintf('%10s',num2str(ppio,'%2.4f'));
        puntero=puntero+11;
        total(k).descarga(puntero:puntero+6)='      |';
        puntero=puntero+7;
        total(k).descarga(puntero:puntero+9)=sprintf('%10s',num2str(final,'%2.4f'))
        k=k+1;
        [datos]=fgetl(fid);
  end;
fclose(fid);
[datosgen]=fgetl(fidgen);
end
fclose(fidgen);
str = {total.descarga};
set(handles.listbox1,'String',str,'Value',1);
set(handles.num_desc,'String',num2str(num_des),'Value',1);
function varargout = show_discharges_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
%
function listbox1_Callback(hObject, eventdata, handles)
function listbox1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function num_desc_Callback(hObject, eventdata, handles)
function num_desc_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

Procedimiento *positives.m*

```
%Programa que revisa todos los ficheros de las descargas analizadas y emite
% un informe z traves de una venta con todo lo que ha encontrado pero sin
% sacar duplicados de intervalos ,hace union e intersección.
function varargout = show_discharges(varargin)
global error;
error=false;
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @show_discharges_OpeningFcn, ...
```

```
                   'gui_OutputFcn',  @show_discharges_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    if error return; end;
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
function show_discharges_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;
guidata(hObject, handles);
try,
global fecha_find;
global matriz;
global senal_itv;
global canal_itv;
global ppio_itv;
global num_des;
global error;
num_des=0;
%global matriz2;
linea(1:120)='-';
k=1;
ruta=pwd;
fich_log=fullfile(ruta,'ficheros_cfg.txt');
fid4=fopen(fich_log,'r');
[ruta_logs]=fgetl(fid4);
[ruta_logs]=fgetl(fid4);  % leo dos veces orque en el sgunda linea esta el ficheros para
intervalos
fclose(fid4);
total=struct('descarga',{});
ficherogen=strcat(ruta_logs,'\positivas2.txt');
fichero_salida=strcat(ruta_logs,'\positives.txt');
fidgen=fopen(ficherogen,'r');
fid1=fopen(fichero_salida,'a');
[datosgen]=fgetl(fidgen);
% bucle general
while ischar(datosgen)
    [texto] = char(strread(datosgen,'%s'));
    num_des=num_des+1;
    fichero=strcat(ruta_logs,'\itv',texto,'.txt');
    fid=fopen(fichero,'r');
    total(k).descarga=linea;
    k=k+1;
    total(k).descarga=strcat('DISCHARGE NUMBER..:',texto);
    k=k+1;
    total(k).descarga=linea;
    k=k+1;
    total(k).descarga='        Start  Inteval Time (sg.)         |       End  Interval Time
(sg.)  ';
    k=k+1;
    total(k).descarga=linea;
    k=k+1;
    [datos]=fgetl(fid);
    while ischar(datos)
        puntero=0;
        [descarga1,senal1,canal1,frec,ppio,final] = strread(datos,'%s %s %s %f %f %f');
         l=length(char(senal1));
        lon_frec=length(num2str(frec));
        lon_ppio=length(num2str(ppio));
        total(k).descarga(puntero+1:puntero+30)=sprintf('%30s',num2str(ppio,'%2.4f'));
         puntero=puntero+31;
        total(k).descarga(puntero:puntero+19)='                    |';
        puntero=puntero+20;
        total(k).descarga(puntero:puntero+29)=sprintf('%30s',num2str(final,'%2.4f'));
```

```
        fprintf(fid1,'%10s %10s %10s
\r\n',texto,num2str(ppio,'%2.4f'),num2str(final,'%2.4f'));
        k=k+1;
        [datos]=fgetl(fid);
    end;
    fclose(fid);
    [datosgen]=fgetl(fidgen);
end
fclose(fidgen);
fclose(fid1);
str = {total.descarga};
set(handles.listbox1,'String',str,'Value',1);
set(handles.num_desc,'String',num2str(num_des),'Value',1);
catch,
            [msgstr, msgid] = lasterr;
            err=lasterror;
            st1 = err.stack(1,1)
            error=true;
            texto=strcat(msgstr,msgid,'fichero..:',
st1.file,'Nombre..:',st1.name,'sentencia..:',num2str(st1.line)  );
            errordlg(texto,'Error en lectura de ficheros');
            return;
end;
function varargout = show_discharges_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
function listbox1_Callback(hObject, eventdata, handles)
function listbox1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function num_desc_Callback(hObject, eventdata, handles)
function num_desc_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

Procedimiento *criterio.m*

```
% programa que configura el criterio a seguir para analizar las descargas.
% La utilidad viende dada pues desde aqui se seleccionan las señales y
% canales a estudiar.
function varargout = criterio(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @criterio_OpeningFcn, ...
                   'gui_OutputFcn',  @criterio_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
 function criterio_OpeningFcn(hObject, eventdata, handles, varargin)

handles.output = hObject;
guidata(hObject, handles);
carpeta_actual=pwd;
fichero_criterio=fullfile(carpeta_actual,'criterio.txt');
fid2=fopen(fichero_criterio,'r');
[datos2]=fgetl(fid2)
if datos2~=-1
    [senal,canales] = strread(datos2,'%s %s');
    while datos2>0
```

```
        if strcmp(senal,'PP8')
              set(handles.pp8,'Value',true);
              set(handles.pp8canal,'String',canales);
        end
        if strcmp(senal,'PP4')
              set(handles.pp4,'Value',true);
              set(handles.pp4canal,'String',canales);
        end
          if strcmp(senal,'T0')
              set(handles.t0,'Value',true);
              set(handles.t0canal,'String',canales);
          end
           if strcmp(senal,'I8')
              set(handles.i8,'Value',true);
              set(handles.i8canal,'String',canales);
           end
           if strcmp(senal,'H3')
              set(handles.h3,'Value',true);
              set(handles.h3canal,'String',canales);
           end
       [datos2]=fgetl(fid2);
        if ~ischar(datos2),   fclose(fid2); break,   end
        [senal,canales] = strread(datos2,'%s %s');
    end
end
function varargout = criterio_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;

function pp8_Callback(hObject, eventdata, handles)

function pp4_Callback(hObject, eventdata, handles)

function t0_Callback(hObject, eventdata, handles)

function h3_Callback(hObject, eventdata, handles)

function i8_Callback(hObject, eventdata, handles)

function pp8canal_Callback(hObject, eventdata, handles)

function pp8canal_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function pp4canal_Callback(hObject, eventdata, handles)
function pp4canal_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function t0canal_Callback(hObject, eventdata, handles)
function t0canal_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function h3canal_Callback(hObject, eventdata, handles)
function h3canal_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function i8canal_Callback(hObject, eventdata, handles)
function i8canal_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function OK_Callback(hObject, eventdata, handles)
matriz=struct();
i=1;
try,
catch, end;
```

```
if get(handles.pp8,'Value');
    pp8canal=get(handles.pp8canal,'String');
    matriz(i).senal='PP8';
    matriz(i).canal=char(pp8canal);
    i=i+1;
end;
if get(handles.pp4,'Value');
    pp4canal=get(handles.pp4canal,'String');
    matriz(i).senal='PP4';
    matriz(i).canal=char(pp4canal);
    i=i+1;
end;
if get(handles.t0,'Value');
    t0canal=get(handles.t0canal,'String');
    matriz(i).senal='T0';
    matriz(i).canal=char(t0canal);
    i=i+1;
end;
if get(handles.h3,'Value');
    h3canal=get(handles.h3canal,'String');
    matriz(i).senal='H3';
    matriz(i).canal=char(h3canal);
    i=i+1;
end;
if get(handles.i8,'Value');
    i8canal=get(handles.i8canal,'String');
    matriz(i).senal='I8';
    matriz(i).canal=char(i8canal);
    i=i+1;
end;
carpeta_actual=pwd;
fichero_criterio=fullfile(carpeta_actual,'criterio.txt');
fid=fopen(fichero_criterio,'w');
cont=length(matriz);
for i=1:cont
        fprintf(fid,'%5s %10s\r\n',matriz(i).senal,matriz(i).canal);
end
fclose(fid);
 delete(handles.figure1)
function cancel_Callback(hObject, eventdata, handles)
delete(handles.figure1)
```

Procedimiento *prb_regiones.m*

```
% programa que aplica el algoritmo seleccionado. Se basa en la búsqeda de
 % regiones de interés. Este programa analiza la existencia de DOS armonicos
 % pasos del algoritmo:
 % 1.   Cálculo del espectrograma de la descarga.
 % 2.   Extracción de la componente con mayor dispersión.
 % 3.   Binarización de la imagen anterior.
 % 4.   Aplicación operación XOR.
 % 5.   Eliminación de grandes áreas sin interés
 % 6.   Asignación de etiquetas a cada área
 % 7.   Eliminación mediante estudio de propiedades.
 % 8.   Cálculo de multiplicidades.
function [positivas,negativas,positivo3,negativo3]=prb_regiones(ruta,fichero,parametro);
positivas=false;
negativas=false;
positivo3=false;
negativo3=false;
global rectas_arm_gen;
global rectas_arm;
global ff;
global cc;
try,
      pack;  % para optimizar memoria
      carpeta_actual=pwd;
      % en este fchero guardo el fichero temporal para calculos de 2 armonicos
      fich_temp2=fullfile(carpeta_actual,'temp_2.txt');
      c1=(findstr(ruta,'\'));
```

```
      p1=length(c1);
      p2=length(ruta);
      senal=ruta(c1(p1)+1:p2);
      descarga=ruta(c1(p1-1)+1:c1(p1)-1);
      c1=(findstr(fichero,'-'));
      canal=fichero(1:c1-1);
      clear ('c1','p1','p2');
      fichero_datos=strcat(ruta,'\',fichero);
      [rutaf,nombref,extf]=fileparts(fichero_datos);
      if extf~= '.mat' return; end     %controlamos que sea un fichero de datos
      load (fichero_datos,'Y','X','-mat');
      texto=strcat('Paso 1. Calculando espectrograma para descarga....:', fichero_datos);
      disp(fichero_datos);
      [y,f,t,p]=spectrogram(Y,10000,3000,4096,1e+6,'yaxis');
      clear spectrogram;
      fff=fix(f/1000);
      % pongo por defecto que tome hasta la frec de 60 para hacer la grafica
      % mas legible
      indice=find(fff==60); %encuentro la fila donde esta la frec que busco
      f2=f(1:indice(1));
      pp=p(1:indice(1),:);
      % calculo los tiempos para el eje de abcisas
      %  este bule se puede obviar porque solo sirve para crear el vector de los tiempos
      xlmax=length(X);
      [f,colu]=size(t);
      for a=1:colu
            tiempo(a)=X(fix((xlmax*a)/colu));
      end
      matriz=log10(abs(p)+eps);
      matriz2=log10(abs(pp)+eps);
    j=figure('Visible','off');
    surf(tiempo,f2,matriz2,'EdgeColor','none');
    axis xy; axis tight; colormap(jet); view(0,90);
    % Se ajusta el modo Renderer para obtener mejor definición en gráficos
    set(gcf,'Renderer','Zbuffer');
    fichero_work=strcat(ruta,'\img_work.jpg');
    saveas(j,fichero_work,'jpg');
    xlabel('Segundos');ylabel('Hz');
    clear('Y','matriz2','fff','matriz','pp');
    delete(j);
     %  parte  II
    global ok_filtro;
    numero_arm=0;
    pack;
    umbral=20;
    [ORIGINAL,MAP] = imread(fichero_work,'jpg');
    delete(fichero_work);
    clear('MAP');
    disp('1');
    imagen = imcrop(ORIGINAL,[156 550 930 248]);
    ORIGINAL=imagen;
 %  parte  III     estudio de regiones
    dispersion=[];
    Verde = ORIGINAL(:,:,2);   %se toma la segunda componente  y no la que tiene mas
dispersion
    clear('dispersion','c','indice','Y');
    a=im2bw(Verde);
    [fi,col]=size(a);
    aa=ones(fi,col);
    aa=xor(aa,a);
    BWb2=bwareaopen(aa,250,4);
    aaa=aa-BWb2;
    [Etiquetas, N]=bwlabel(aaa,8);
    Prop
=regionprops(Etiquetas,'FilledArea','Centroid','MinorAxisLength','Orientation','BoundingBox','
SubarrayIdx');
    %
    % selecciono las areas que tienen mas de 6 pixel de area
    % las otras dos selecciones es para evitar las rayas verticales que
    % aparecen en los espectrogramas.
    idx = find(  ([Prop.FilledArea] > 6) & ([Prop.MinorAxisLength] > 1.9)...
```

```
            & (           (  ([Prop.Orientation]< 88)      &          ([Prop.Orientation]>-88)
)...
               |          (  ([Prop.Orientation]>92)       &          ([Prop.Orientation]< -
92)  )   ));
    BW2 = ismember(Etiquetas,idx);
    cont=length(idx);
    % creamos una tabla donde metemos todas los centroides para ordenarles. Es
    % decir debemso ordenar las filas donde estan los armonicos de mayor a
    % menor para mas tarde calcular los multiplos en filas dode estan las regiones que a su
    % seran los armonicos
    regiones_sort=[];
    for i=1:cont
        regiones_sort(i,1)=idx(i);  % meto el numero de region
        regiones_sort(i,2)=fix(Prop(idx(i)).Centroid(2));  % meto la fila del centroide  de la
region
        regiones_sort(i,3)=fix(Prop(idx(i)).Centroid(1));  % meto la columna del centroide  de
la region
    end
    regiones_sort=sortrows(regiones_sort,-2);  %ordeno en descendente por la segunda columna
    % ahora voy a calcular las regiones que pueden llegar a ser multimplos
    % entre ellas en funcion de las filas.
    regiones_ok=[]; %aqui voy a meter en la primera columna el numero del arm fdtal y en la
segunda col el primer arm.Sol vamos a
    k=1;                % buscar un fundamental y su primer armonico
    for i=1:cont-1
    distancia=fi-regiones_sort(i,2);  % miro a cuantas filas está del 0 es decir del origen
     %para calcular el armonico 1 tendremos en cuanta un tolerancia de +/-
     %  +/- 10  filas
     dist1=distancia-10;
    dist2=distancia+10;
    lateral1=regiones_sort(i,3)-15;  %tolerancia de -15 columnas
    lateral2=regiones_sort(i,3)+15;  %tolerancia del +15 columnas
    for jj=i+1:cont
        if
(regiones_sort(i,2)>=(regiones_sort(jj,2)+dist1))&(regiones_sort(i,2)<=(regiones_sort(jj,2)+di
st2))
            if (regiones_sort(jj,3)>=lateral1)&(regiones_sort(jj,3)<=lateral2)
                regiones_ok(k,1)=regiones_sort(i,1);  % meto la col de la imagen del ppio del
arm
                regiones_ok(k,2)=regiones_sort(jj,1); % meto la col de la imagen del fin del
arm
                regiones_ok(k,3)=regiones_sort(i,3); % meto la fila para calc la fec.
                k=k+1;
            end
        end
       end
    end
    [cont2,x]=size(regiones_ok);
    lon=length(tiempo);
    %
    % Extraigo datos relatvos a la descarga
    %
    c1=(findstr(ruta,'\'));
    p1=length(c1);
    p2=length(ruta);
    senal=ruta(c1(p1)+1:p2);
    descarga=ruta(c1(p1-1)+1:c1(p1)-1);
    c1=(findstr(fichero,'-'));
    canal=fichero(1:c1-1);
    clear ('c1','p1','p2');
    fndtal=[];
    % en esta variable voy a anotar donde comienza y donde termina las columnas
    % del fundamental
    for i=1:cont2;
        columnas=Prop(regiones_ok(i,1)).SubarrayIdx(2);  % elijo las columnas de este
parametro
        longi=length(columnas{1});
        fndtal(i,1)=columnas{1}(1);
        fndtal(i,2)=columnas{1}(longi);
    end;
    fndtal_2=struct();
```

```
    [filas,colus]=size(regiones_ok);
    % si no encuentra nada anotamos en fichero de error
    fid0=fopen(fich_temp2,'a');
    %fid=fopen(parametro.fichero_log,'a');
    for i=1:cont2
        %  en la variable regiones_ok tengo los numeros de regiones NO los
        %  numeros de fila ó columna
        region=(Etiquetas == regiones_ok(i,1))+(Etiquetas == regiones_ok(i,2));
        %figure(),imshow(region);
        %pause;
        eje=round(Prop(regiones_ok(i,1)).BoundingBox(3)/2);
        tpo=Prop(regiones_ok(i,1)).Centroid(1);             % nos da la columna del centroide
donde esta
        fila=Prop(regiones_ok(i,1)).Centroid(2);            % esto nos da la fila del
centroide donde esta el fundamental
        g=length(f2);
        frec=(fila*5)/180;
        %  frec=f2(round((fila*247)/732));
        % calculo una regla de tres para aber a donde corresponde la columna del
        % arm fdtal. La regla de tres se hace con la long del vctor de tiempo,
        % la lon en columnas de la imagen y la columna de la region del arm
        % fdtal
        tpo_ppio=round((lon*(tpo-eje))/col);
        tpo_fin=round((lon*(tpo+eje))/col);
        % calculo mediante una media aritmetica el tiempo que le corresponde a cada armonico
        % esta info la extraigo de X
        lon_X=length(X);
        periodo=round(lon_X/(col-1));
        tpo_ppio2=round(1+((tpo-eje)-1)*periodo);     % Media para encontrar el tiempo en el
vector de tiempos
        tpo_fin2=round(1+((tpo+eje)-1)*periodo);
        fndtal_2(i).col1=fndtal(i,1);
        fndtal_2(i).col2=fndtal(i,2);
        fndtal_2(i).desc=descarga;
        fndtal_2(i).senal=senal;
        fndtal_2(i).canal=canal;
        fndtal_2(i).frec=num2str(frec);
        fndtal_2(i).ppio=num2str(X(tpo_ppio2));
        fndtal_2(i).final=num2str(X(tpo_fin2));
        texto=strcat('  Descarga..  ',descarga,'  senal..:  ',senal,'  canal..:  ',canal,'
frec.: ',num2str(frec),'  Inic.: ',num2str(X(tpo_ppio2)),'   Fin.:  ',num2str(X(tpo_fin2)));
        disp(texto);
        fprintf(fid0,'%8s %5s %3s %3.4f %3.4f
%3.4f\r\n',num2str(descarga),senal,num2str(canal),double(frec),double(X(tpo_ppio2)),double(X(t
po_fin2)));
    end
fclose(fid0);
h=1;
datosDes=struct([]);
bb=[];
fid0=fopen(fich_temp2,'r');
% pasamos a escribir en el fichero de positivos o negativos
try,
        [datos1]=fgetl(fid0);

[datosDes(h).descarga,datosDes(h).senal,datosDes(h).canal,datosDes(h).frec,datosDes(h).ppio,da
tosDes(h).final]...
            = strread(datos1,'%s %s %s %f %f %f');
        bb(h)=datosDes(h).ppio;
        h=h+1;
        [datos1]=fgetl(fid0);
        while (ischar(datos1))

[datosDes(h).descarga,datosDes(h).senal,datosDes(h).canal,datosDes(h).frec,datosDes(h).ppio,da
tosDes(h).final]...
                    = strread(datos1,'%s %s %s %f %f %f');
                if ~find(bb==datosDes(h).ppio)
                    bb(h)=datosDes(h).ppio;
                    h=h+1;
                end
                [datos1]=fgetl(fid0);
```

```
        end
        fid=fopen(parametro.fichero_log,'a');
        for i=1:h
            fprintf(fid,'%8s %5s %3s %3.4f %3.4f
%3.4f\r\n',char(datosDes(i).descarga),char(datosDes(i).senal),char(datosDes(i).canal),...
            double(datosDes(i).frec),double(datosDes(i).ppio),double(datosDes(i).final));
        end
        fclose(fid);
        positivas=true;
catch,
        %si no encontramos nada escribimos en el fichero de negativos
        negativas=true;
end;
% hasta aqui hemos buscado armonicos que tengan el fundamental y un primer
% armonico. En la variable regiones_ok  tengo todo lo necesario para buscar
% el fundamental y dos armonicos. de esto se encarga prb_regiones_2
fclose(fid0);
parametro.tiempo=tiempo;
[positivo3,negativo3]=prb_regiones_2(imagen,fndtal_2,parametro.fichero_log2);
clear('prb_regiones2');
return;
catch,
            disp('Desde.........: prb_regiones');
            fclose('all');
            [msgstr, msgid] = lasterr;
            err=lasterror;
            st1 = err.stack(1,1)
            %st2=err.stack(2,1)
            texto=strcat('Error prb_regiones.........:',msgstr,msgid);
            texto=strcat(msgstr,msgid,'fichero..:',
st1.file,'Nombre..:',st1.name,'sentencia..:',num2str(st1.line)  );
            %errordlg(texto,'Error en lectura de ficheros');
            disp(msgstr);
           disp(msgid);
           disp(['fichero..:',
st1.file]);disp(['Nombre..:',st1.name]);disp(['sentencia..:',num2str(st1.line)]);
           return;
end
```

Procedimiento *search_arm1.m*

```
% Este programa es el modulo ppal de la busquea de armonicos. Se encarga de
% leer de forma secuencial los ficheros de las descargas e ir llamando a
% todos los modulos
%=========================================================================
function search_arm1;
     %warning off;
     pack;
     global  matriz_log;
     global h;
     matriz_log=struct([]);
     texto='*************************************************************';
     nom_direc_dst = uigetdir('C:\','Seleccione la ruta donde está el fichero -
descargas.txt');
     %  en fichero estan todos los nombres de señales y numero de
     %  descargas que deben ser analizadas
     fichero=strcat(nom_direc_dst,'\descargas.txt');
     scrsz = get(0,'ScreenSize');
     % genero pantalla que mostrara el log del proceso
     h = figure('NumberTitle','off','Name','Análisis de Descargas','Position',[200 100 800
620],'Visible','off');
     ss='';
     j= uicontrol('Parent',h,...
                  'Units','normalized',...
                  'Position',[0.05,0.05,0.93,0.93],...
                  'Style','edit',...
                  'Max',100,...
                  'FontSize',13,...
                  'HorizontalAlignment','left',...
                  'Enable','inactive',...
                  'String',ss,'Visible','on');
```

```
    nom_direc_dst = uigetdir('C:\','Seleccione la ruta donde están las descargas');
    datos='str';
    matriz_log(1).texto=texto;
    matriz_log(2).texto=strcat('Análisis de ficheros *mat en la ruta....:', nom_direc_dst);
try,
    % leo ruta ficheros log
    ruta=pwd;
    fich_log=fullfile(ruta,'ficheros_cfg.txt');
    fid4=fopen(fich_log,'r');
    [ruta_logs]=fgetl(fid4);
    fclose(fid4);
    % por cada nombre de señal que encuentre en -fichero- tendre que
    % analizar todos los ficheros que contienen los canales
      fichero_log=strcat(ruta_logs, '\LOG_2ARM_',datestr(now,30),'.txt');
      fichero_log2=strcat(ruta_logs,'\LOG_3ARM_',datestr(now,30),'.txt');
      fichero_log4=strcat(ruta_logs,'\LOG_ERR_',datestr(now,30),'.txt');
       fichero_log5=strcat(ruta_logs,'\LOG_GEN_DESCARGAS.txt');
      fichero_log3=strcat(ruta_logs,'\LOG_GEN.txt');
      fid3=fopen(fichero_log3,'a');
      hora=clock;
      fprintf(fid3,'%s\r\n',strcat('300 **********    COMIENZO NUEVO
ANALISIS________:',date,'_______Hora__:',num2str(hora(4)),':',num2str(hora(5)) ));
      [x,dia]=weekday(date,'long');
      fprintf(fid3,'%s\r\n',strcat(dia,'..',date,'..',num2str(hora(4)),':',num2str(hora(5))));
      fprintf(fid3,'%s\r\n', nom_direc_dst);
      fprintf(fid3,'%s\r\n',fichero_log);
      fprintf(fid3,'%s\r\n',fichero_log2);
      fprintf(fid3,'%s\r\n',fichero_log4);
      fclose(fid3);
      clear('fid3');
      fid=fopen(fichero,'r');
      j=0;
      mat_desc=[];
      mat_desc_anz=[];
      k=1;
% Voy a calcular de forma aleatoria las descargas a procesar. Lo que se analizara
% son 16 descargas del total. Una vez realizados los analisis se probaran
% con 5 mas desacargas para despues analizar todas las descargas del total.
% en la variable mat_desc tendre todas las descargas de la carpeta
% seleccionada
parametro=[];
descargas=dir(nom_direc_dst);
cont1=length(descargas)
parametro.fichero_log=fichero_log;
parametro.fichero_log2=fichero_log2;
parametro.fichero_log4=fichero_log4;
parametro.fichero_log5=fichero_log5;
parametro.fich_interv=' ';
parametro.nom_direc_dst=nom_direc_dst;
% hago lectura adelantada del fichero
num_descarga=false;
[datos]=fgetl(fid);fclose(fid);
datos_str=num2str(datos);
while  datos>0

           num_descarga=false;
          for j=3:cont1
              if descargas(j).name==datos_str
                   parametro.num_desc=descargas(j).name;
                   search_arm2(parametro);
                    clear search_arm2;
                   num_descarga=true;
              end
              if num_descarga  break,end
          end
          fid3=fopen(parametro.fichero_log,'a');
          fclose(fid3);
          fid=fopen(fichero,'r');
          bucle=true;
          [datos]=fgetl(fid);
```

```
            while bucle |(datos~=-1)
                 tempr=num2str(datos);
                 if tempr==datos_str
                     [datos]=fgetl(fid); %leo la sigiente linea y me quedo con ella
                     datos_str=num2str(datos);
                     fclose(fid);
                     bucle=false;
                     break;
                 else
                    [datos]=fgetl(fid);
                 end
            end
            if datos==-1,
                %fclose(fid);
                break,end

end;
          fid3=fopen(fichero_log3,'a');
          fprintf(fid3,'%s\r\n','400 ******');fclose(fid3);
         % este pgm realiza anotaciones de tres armonicos
         pgm920(fichero_log3,fichero_log,fichero_log2);
         fid3=fopen(fichero_log3,'a');
          fprintf(fid3,'%s\r\n','410 ******');
          fprintf(fid3,'%s\r\n','500 ******');fclose(fid3);
         % este pgm realiza anotaciones de dos armonicos
         pgm930(fichero_log3,fichero_log,fichero_log2);
         fid3=fopen(fichero_log3,'a');
         fprintf(fid3,'%s\r\n','510 ******');
        %fid2=fopen(fichero_log2,'a');
         hora=clock;
         fprintf(fid3,'%s\r\n',strcat('310 **********    FIN NUEVO
ANALISIS________:',date,'_______Hora__:',num2str(hora(4)),':',num2str(hora(5))
),'**********');
         fclose(fid3);
          return;
    catch,
           disp('Desde.........: search_arm1');

             [msgstr, msgid] = lasterr;
             err=lasterror;
             st1 = err.stack(1,1)
             texto=strcat('Error  search_arm1.........:',msgstr,msgid);
             disp(msgstr);
             disp(msgid);
             disp(['fichero..:',
st1.file]);disp(['Nombre..:',st1.name]);disp(['sentencia..:',num2str(st1.line)]);
             return;
end
```

Procedimiento *search_arm2.m*

```
     function search_arm2(parametro);
  positivas=false;
  negativas=false;
  positivo3=false;
  negativo3=false;
 try,
 senales=dir(strcat(parametro.nom_direc_dst,'\',parametro.num_desc));
 %leemos las señales que estan puestas como criterio a buscar%
 cont=length(senales);
  ruta=pwd;
    fich_log=fullfile(ruta,'ficheros_cfg.txt');
    fid4=fopen(fich_log,'r');
    [ruta_logs]=fgetl(fid4);
    [ruta_logs]=fgetl(fid4);  % leo dos veces orque en el sgunda linea esta el ficheros para
intervalos
    fclose(fid4);
 carpeta_actual=pwd;
 fich_criterio=fullfile(carpeta_actual,'criterio.txt');
 senal_analisis=struct([]);
```

```
 % en este fchero guardo el fichero temporal para calculos de 2 armonicos
 fich_temp2=fullfile(carpeta_actual,'temp_2.txt');
 fid=fopen(fich_temp2,'wt');
 fclose(fid);
 try,mkdir('intervalos'); catch,end;
 fid3=fopen(fich_criterio,'r');
 u=0;
 [datos]=fgetl(fid3);
  while (ischar(datos))
      u=u+1;
        [senal_analisis(u).senal,senal_analisis(u).canales] = strread(datos,'%s %s');
        [datos]=fgetl(fid3);
  end
  fclose(fid3);
 for i=3:cont
        for j=1:u
            % analizo si la señal corresponde al criterio %
             if strcmp(senal_analisis(j).senal, senales(i).name)
                 % analizo si el canal corresponde a los canales %

nom_dst2=strcat(parametro.nom_direc_dst,'\',parametro.num_desc,'\',senales(i).name)

                    canales=dir(nom_dst2);

                    % con esto voy a leer los canales que hay en cada señal
                    cont2=length(canales);
                    %close all;
                    for t=3:cont2
                            aa=findstr(char(senal_analisis(u).canales),canales(t).name(1:1))
                            if ~isempty(aa)

[positivas,negativas,positivo3,negativo3]=prb_regiones(nom_dst2,canales(t).name,parametro);
                                clear prb_regiones;
                                % close all;
                            end
                    end
                    pause(1);
             end
        end %del segundo for
 end
 %aqui no hemos tenido exito y no hemos detectado nada
 depura;

 fid=fopen(fullfile(pwd,'temp_2.txt'),'r');
 fichero_anot=strcat(ruta_logs,'\itv',parametro.num_desc,'.txt');
fid1=fopen(fichero_anot,'a');
[datos]=fgetl(fid);

while ischar(datos)
         [descarga1,senal1,canal1,frec1,ppio1,final1] = strread(datos,'%s %s %s %f %f %f');
        fprintf(fid1,'%8s %5s %3s %3.4f %3.4f
%3.4f\r\n',char(descarga1),char(senal1),char(canal1),...
        double(frec1),double(ppio1),double(final1));
        [datos]=fgetl(fid);

end;
fclose(fid);
fclose(fid1);


 if negativas
        file=strcat(ruta_logs,'\negativas2.txt')
        fid=fopen(file,'a');
        fprintf(fid,'%8s\r\n',num2str(parametro.num_desc));
        fclose(fid);
 end;
 if positivas
        file=strcat(ruta_logs,'\positivas2.txt')
        fid=fopen(file,'a');
        fprintf(fid,'%8s\r\n',num2str(parametro.num_desc));
        fclose(fid);
```

```
 end;
 if positivo3
        file=strcat(ruta_logs,'\positivas3.txt')
        fid=fopen(file,'a');
        fprintf(fid,'%8s\r\n',num2str(parametro.num_desc));
        fclose(fid);
 end;
 if negativo3
        file=strcat(ruta_logs,'\negativas3.txt')
        fid=fopen(file,'a');
        fprintf(fid,'%8s\r\n',num2str(parametro.num_desc));
        fclose(fid);
 end;
 pause(1);
 return;
 pause(1);
 catch,
           disp('Desde.........: search_arm2');

            [msgstr, msgid] = lasterr;
            err=lasterror;
            st1 = err.stack(1,1)
            %st2=err.stack(2,1)
            texto=strcat('Error  search_arm2.........:',msgstr,msgid);
            disp(msgstr);
           disp(msgid);
           disp(['fichero..:',
st1.file]);disp(['Nombre..:',st1.name]);disp(['sentencia..:',num2str(st1.line)]);
           %disp(['fichero..:',
st2.file]);disp(['Nombre..:',st2.name]);disp(['sentencia..:',num2str(st2.line)]);
           return;
end
```

Procedimiento *depura.m*

```
% este programa realiza la depuracion de los intervalos de armonicos
% detectados. Al margen del canal o señal donde se haya encontrado el
% armonico es llamado por el programa pgm901
fich_temp2=fullfile(pwd,'temp_2.txt');
fid=fopen(fullfile(pwd,'temp_2.txt'),'r');
[datos]=fgetl(fid);
matriz=struct([]);
fundamental=[];
%final=[];
i=1;
while ischar(datos)

[matriz(i).descarga,matriz(i).senal,matriz(i).canal,matriz(i).frec,matriz(i).ppio,matriz(i).fi
nal] = strread(datos,'%s %s %s %f %f %f');
        fundamental(i,1)=matriz(i).ppio;
        fundamental(i,2)=matriz(i).final;
        [datos]=fgetl(fid);
        i=i+1;
end;
fclose(fid);
[fundamental2,ind]=sort(fundamental,1,'ascend');
fundamental2(:,3)=ind(:,1);
clear('fundamental');
fundamental=[];
[fi,col]=size(fundamental2);
fundamental(1,1)=fundamental2(1,1);
fundamental(1,2)=fundamental2(1,2);
fundamental(1,3)=fundamental2(1,3);
g=2;
%eliminamo dupliidadesde intervalos
for t=2:fi
    if fundamental(g-1,1)~=fundamental2(t,1)
        fundamental(g,1)=fundamental2(t,1);
        fundamental(g,2)=fundamental2(t,2);
```

```
        fundamental(g,3)=fundamental2(t,3);
        g=g+1;
    end
end
% ahora vamos a hacer las uniones entre intervalos
  for t=2:fi
    if fundamental(g-1,1)~=fundamental2(t,1)
        fundamental(g,1)=fundamental2(t,1);
        fundamental(g,2)=fundamental2(t,2);
        fundamental(g,3)=fundamental2(t,3);
        g=g+1;
    end
  end
  clear('fundamental2');
fundamental2=[];
fundamental2=fundamental;
 clear('fundamental');
fundamental=[];

fundamental(1,1)=fundamental2(1,1);
fundamental(1,2)=fundamental2(1,2);
fundamental(1,3)=fundamental2(1,3);
k=2;
t=2;
bucle=true
while bucle
    indice=find(fundamental2(:,1)>fundamental2(t-1,2))
    if ~isempty(indice)
    t=indice(1)

        fundamental(k,1)=fundamental2(t,1)
         fundamental(k,2)=fundamental2(t,2)
         fundamental(k,3)=fundamental2(t,3);
        k=k+1;
        t=t+1;
    else
        bucle=false;
    end
end
%Ahora ya vamos a ornadr la lista pues ya hemos quitado duplicidades y hemos hecho uniones
[fi,col]=size(fundamental);
fid=fopen(fullfile(pwd,'temp_2.txt'),'w');
for i=1:fi
    ind=fundamental(i,3);
    fprintf(fid,'%8s %5s %3s %3.4f %3.4f
%3.4f\r\n',char(matriz(ind).descarga),char(matriz(ind).senal),char(matriz(ind).canal),...
           double(matriz(ind).frec),double(matriz(ind).ppio),double(matriz(ind).final));
end
fclose(fid);
```

Procedimiento numFich.*m*

```
% fucnion que ayuda a calcular el numero de ficheros de la carpeta
% es llamad por la funcion calculaNumFich

function numero = numFich(dirName,senal,fichero,num_descargas)
numero=0;
fileList = getAllFiles(dirName);
[cont,x]=size(fileList);
[x,cont_desc]=size(num_descargas);
for i=1:cont;
    nombre=char(fileList(i));
    cant=strfind(nombre,'.mat'); %only study .mat file becuase exists other files type
    if ~isempty(cant)
        cant=strfind(nombre,senal);% name signal  it is?
        if ~isempty(cant)
                cant=strfind(nombre,fichero); % discharge channel it is?
                if ~isempty(cant)
                    for k=1:cont_desc
```

```
                        cant=strfind(nombre, num2str(num_descargas(k))); % discharge number it
is?
                        if ~isempty(cant)
                           numero=numero+1;
                        end
                    end
                end
        end
    end
end
```

Procedimiento prb_regiones2.*m*

```
% programa que aplica el algoritmo seleccionado. Se basa en la búsqeda de
 % regiones de interés. Este programa analiza la existencia de TRES armonicos
 % pasos del algoritmo:
 % 1.   Cálculo del espectrograma de la descarga.
 % 2.   Extracción de la componente con mayor dispersión.
 % 3.   Binarización de la imagen anterior.
 % 4.   Aplicación operación XOR.
 % 5.   Eliminación de grandes áreas sin interés
 % 6.   Asignación de etiquetas a cada área
 % 7.   Eliminación mediante estudio de propiedades.
 % 8.   Cálculo de multiplicidades.
function [positivo3,negativo3]=prb_regiones_2(imagen,fndtal_2,fichero_log2);
positivo3=false;
negativo3=false;
try,
pack;
[xxx,long]=size(fndtal_2);
% ahora por cada fila de la variable fndtal debemos calcular si tres
% armonicos en la columnas de la imagen definida por 5 columnas a la isq de
% donde comienza el fundamental y 5 col por donde finaliza el fndtal.
for h=1:long
    if (fndtal_2(h).col1 -5)>0      % esto es para control de lateralidad y que no se salga de
la imagen
        inic=fndtal_2(h).col1 -5;
    else
        inic=1;
    end
    if (fndtal_2(h).col2 + 5)<931
        fin=fndtal_2(h).col2 + 5;
    else
        fin=931;
    end
    ancho=fin-inic;

imagen2 = imcrop(imagen,[inic 1 ancho 247]);
 %  parte  III     estudio de regiones
Verde = imagen2(:,:,2);
clear('dispersion','c','indice');
a=im2bw(Verde);
[fi,col]=size(a);
aa=ones(fi,col);
aa=xor(aa,a);
BWb2=bwareaopen(aa,250,4);
aaa=aa-BWb2;
[Etiquetas, N]=bwlabel(aaa,8);    % figure,imshow(Etiquetas==13); Con esta inst mostramos la
region numero 13
Prop
=regionprops(Etiquetas,'FilledArea','Centroid','MinorAxisLength','Orientation','BoundingBox','
SubarrayIdx','Extrema');
% selecciono las areas que tienen mas de 6 pixel de area
% las otras dos selecciones es para evitar las rayas verticales que
% aparecen en los espectrogramas.
idx = find(  ([Prop.FilledArea] > 6) & ([Prop.MinorAxisLength] > 1.9)...
            & (           (  ([Prop.Orientation]< 88)      &          ([Prop.Orientation]>-88)
)...
                |          (  ([Prop.Orientation]>92)       &          ([Prop.Orientation]< -
92)  )   ));
```

```
BW2 = ismember(Etiquetas,idx); % en idx tenemos las regiones limpias y dispuestas para
analizar
cont=length(idx);
% creamos una tabla donde metemos todas los centroides de las regiones para ordenarles. Es
% decir debemso ordenar las filas donde estan los armonicos de mayor a
% menor para mas tarde calcular los multiplos en filas dode estan las regiones que a su
% seran los armonicos
%
regiones_sort=[];
for i=1:cont
    regiones_sort(i,1)=idx(i);  % meto el numero de region
    regiones_sort(i,2)=fix(Prop(idx(i)).Centroid(2));  % meto la fila del centroide  de la
region
    regiones_sort(i,3)=fix(Prop(idx(i)).Centroid(1));  % meto la columna del centroide  de la
region
end
% verify if variable  regiones_sort is tridimensional vector
if isempty(regiones_sort)
    fclose('all');
    %clear;
    % close(gcf);
    % fclose('all');
    return;
end;
if isvector(regiones_sort(:,1))& isvector(regiones_sort(:,2)) & isvector(regiones_sort(:,1))
      if isempty(regiones_sort)
            fclose('all');
            return;
     end;
      regiones_sort=sortrows(regiones_sort,-2);  %ordeno en descendente por la segunda columna
else
    fclose('all');
   % clear;
    % close(gcf);
       %  fclose('all');
    return; % we haven't 3 vectors
end;
% ahora voy a calcular las regiones que pueden llegar a ser multimplos
% entre ellas en funcion de las filas.
regiones_filas_ok=[]; %aqui voy a meter en la primera columna el numero del arm fdtal y en la
segunda col el primer arm.Sol vamos a
lon_reg=0;                % buscar un fundamental y su primer armonico
for i=1:cont-2
    distancia=fi-regiones_sort(i,2);  % miro a cuantas filas está del 0 es decir del origen
     %para calcular el armonico 1 tendremos en cuanta un tolerancia de +/-
     %  +/- 10  filas
     dist1=distancia-10;
    dist2=distancia+10;
    %lateral1=regiones_sort(i,3)-regiones_sort(i,3)*0.08;  %tolerancia del 8%
    %lateral2=regiones_sort(i,3)+regiones_sort(i,3)*0.08;  %tolerancia del 8%
    lateral1=regiones_sort(i,3)-5;  %tolerancia de -5 columnas
    lateral2=regiones_sort(i,3)+5;  %tolerancia del +5 columnas
    for jj=i+1:cont-1
        if
(regiones_sort(i,2)>=(regiones_sort(jj,2)+dist1))&(regiones_sort(i,2)<=(regiones_sort(jj,2)+di
st2))
           if (regiones_sort(jj,3)>=lateral1)&(regiones_sort(jj,3)<=lateral2)
               for jjj=jj+1:cont
                   if
(regiones_sort(jj,2)>=(regiones_sort(jjj,2)+dist1))&(regiones_sort(jj,2)<=(regiones_sort(jjj,2
)+dist2))
                               lon_reg=lon_reg+1;
                               regiones_filas_ok(lon_reg,1)=regiones_sort(i,1);  % meto el
numero de region del fdtal
                               regiones_filas_ok(lon_reg,2)=regiones_sort(jj,1); % meto el
numero de region del primer armonico
                               regiones_filas_ok(lon_reg,3)=regiones_sort(jjj,1); % meto el
numero de region del segundo armonico
                   end
               end
           end
```

```
        end
    end
end


cont2=lon_reg;
%
%    intervalo=(fi-5:fi);  %esta variable sirve para comparar el inetrvalo que debe de haber
entre el origen y el fundamental
%     lo que hacemos es no tener en cuenta aquellos fundamentales que estan
%      practicamente pegando al eje 0.
%      Se elimina esta modificación por dar mas problemas.
%      if lon_reg>0
  %           for rr=1:lon_reg
    %              if
ismember(1,ismember(intervalo,fix(Prop(regiones_filas_ok(rr,1),1).Extrema(:,2))))
      %            cont2=cont2-1;
    %                        end;
   %               end;
%                   end;

fid=fopen(fichero_log2,'a');
%
% Extraigo datos relatvos a la descarga
%
if cont2~=0
            % calculo una regla de tres para aber a donde corresponde la columna del
            % arm fdtal. La regla de tres se hace con la long del vctor de tiempo,
            % la lon en columnas de la imagen y la columna de la region del arm
            % fdtal
            % calculo mediante una media aritmetica el tiempo que le corresponde a cada
armonico
            % esta info la extraigo de X
            %lon_X=length(X);
            %periodo=round(lon_X/(col-1));
            %tpo_ppio2=round(1+((tpo-eje)-1)*periodo);     % Media para encontrar el tiempo en
el vector de tiempos
            %tpo_fin2=round(1+((tpo+eje)-1)*periodo);
            texto=strcat('  Descarga..  ',fndtal_2(h).desc,'  senal..:  ',fndtal_2(h).senal,'
canal..:  ',fndtal_2(h).canal,...
              '  frec.: ',fndtal_2(h).frec,'  Inic.: ',fndtal_2(h).ppio,'   Fin.:
',fndtal_2(h).final );
            disp(texto);
             fprintf(fid,'%8s %5s %3s %3.4f %3.4f
%3.4f\r\n',num2str(fndtal_2(h).desc),fndtal_2(h).senal,...

num2str(fndtal_2(h).canal),str2num(fndtal_2(h).frec),str2num(fndtal_2(h).ppio),str2num(fndtal_
2(h).final));
            positivo3=true;
            fclose(fid);
            % hasta aqui hemos buscado armonicos que tengan el fundamental

    else
        negativo3=true;
    end %del if anterior
    end  %del for h=1:long
    fclose('all');
   % clear;
    % close(gcf);
        % fclose('all');
return
catch,
            fclose('all');
            disp('Desde.........: prb_regiones_2');
            fclose('all');
            [msgstr, msgid] = lasterr;
            err=lasterror;
            st1 = err.stack(1,1)
            %st2=err.stack(2,1)
            texto=strcat('Error prb_regiones_2.........:',msgstr,msgid);
            disp(msgstr);
```

```
            disp(msgid);
            disp(['fichero..:',
st1.file]);disp(['Nombre..:',st1.name]);disp(['sentencia..:',num2str(st1.line)]);
            fclose('all');
           % clear;
     %close(gcf);
        % fclose('all');
            return;
end
```

Procedimiento search_arm3.*m*

```
% Esta rutina se encarga de anotar LOGS para el caso de resultados con dos armonicos.
%
function search_arm3(fichero_log3,fichero_log,fichero_log2);
  try,
 fid=fopen(fichero_log2,'r');

 matriz_frec=[];
 %matriz_senal=[];
 matriz_canal=[];
 matriz_frec(1,:)=0;
 matriz_ppio=[];
  matriz_ppio(1,:)=0;
 cont=1;
 cont2=1;
 [datos]=fgetl(fid);
 matriz_senal=struct([]);
  if ischar(datos)
            [descarga,senal,canal,frec,ppio,final] = strread(datos,'%d %s %d %f %f %f');
            % [datos]=fgetl(fid);
            while  datos>0
                        nuevasenal=true;
                       %   analizo estadisticas de frecuencia
                       frecuencia=round(frec);
                       [indice,xx]=find(matriz_frec(:,1)==frecuencia);
                       if isempty(indice)
                                   % no ha encontrado coincidencia e
                                   % introduce un nueva frecuencia en
                                   % la matriz
                                   matriz_frec(cont,1)=frecuencia;
                                   matriz_frec(cont,2)=1;
                                   %matriz_senal(cont).senal=senal;
                                  % matriz_senal(cont).cuenta=1;
                                   cont=cont+1;

                       else
                                   % encuentra coincidencia y lo que
                                   % hace es sumar a la frec ya registrada
                                    matriz_frec(indice,2)=matriz_frec(indice,2)+1;


                       end
                       %   analizo la aparicion de señal
                       if isempty(matriz_senal)
                                       matriz_senal(1).senal=upper(senal);
                                           matriz_senal(1).cuenta=1;
                       else
                               [jj,xx]=size(matriz_senal);
                               for j=1:xx
                                         if  isequal(matriz_senal(j).senal,upper(senal))

matriz_senal(j).cuenta=matriz_senal(j).cuenta+1;
                                             nuevasenal=false;
                                             break;
                                         end

                                   end
                       end
                       if nuevasenal
```

```
                                [jj,xx]=size(matriz_senal);
                                matriz_senal(xx+1).cuenta=1;
                                matriz_senal(xx+1).senal=senal;
                            end
                            %   analizo intervalos de armonicos
                             ppio=fix(ppio);
                            [indice,xx]=find(matriz_ppio(:,1)==ppio);
                            if isempty(indice)
                                        matriz_ppio(cont2,1)=ppio;
                                        matriz_ppio(cont2,2)=1;
                                        cont2=cont2+1;
                            else
                                        matriz_ppio(indice,2)=matriz_ppio(indice,2)+1;
                            end


                          [datos]=fgetl(fid);
                          if ~ischar(datos),   break,   end
                          [descarga,senal,canal,frec,ppio,final] = strread(datos,'%d %s %d %f %f
%f');

                end;  %del while
  end; %del if
  try, close(fid); catch,end;
  fid3=fopen(fichero_log3,'a');
   [jj,xx]=size(matriz_ppio);
  total=0;
  for j=1:jj
      total=matriz_ppio(j,2)+total;
  end
  for j=1:jj
       texto=strcat('Armonicos_con_comienzo_en..: ',num2str(  matriz_ppio(j,1)
),'..segundos..',...
                          num2str(  (matriz_ppio(j,2)*100)/total   ), '%' );
        fprintf(fid3,'%s\r\n',texto);disp(texto);
  end
 %calculamos los procentajes de las frecuencias
   [jj,xx]=size(matriz_frec);
  total=0;
  for j=1:jj
      total=matriz_frec(j,2)+total;
  end
  for j=1:jj
       texto=strcat('Frecuencia_del_fundamental.:',num2str(matriz_frec(j,1)), '_Khz.......',
num2str(  (matriz_frec(j,2)*100)/total   ), '%' );
        fprintf(fid3,'%s\r\n',texto);disp(texto);
  end

 pause(1);
 total=0;
 [jj,xx]=size(matriz_senal);
 for j=1:xx
        total=matriz_senal(j).cuenta+total;
 end
 %calculamos los procentajes de las señales
 disp('Porcentajes de exito en señales');
 for j=1:xx

        texto=strcat('Positivos_de_la_senal..:',
cell2mat(matriz_senal(j).senal),'....',num2str(    (matriz_senal(j).cuenta*100)/total)
,'%' );
         % disp([ matriz_senal(j).senal,'..:', num2str(    (matriz_senal(j).cuenta*100)/total)
,'%']);
        fprintf(fid3,'%s\r\n',texto);disp(texto);
 end
   fclose(fid3);
   %fclose('all');
   return;
 catch,

            [msgstr, msgid] = lasterr;
```

```
            err=lasterror;
            st1 = err.stack(1,1)
            %st2=err.stack(2,1)
            texto=strcat('Error  search_arm3.........:',msgstr,msgid);
            disp(msgstr);
           disp(msgid);
           disp(['fichero..:',
st1.file]);disp(['Nombre..:',st1.name]);disp(['sentencia..:',num2str(st1.line)]);
           %disp(['fichero..:',
st2.file]);disp(['Nombre..:',st2.name]);disp(['sentencia..:',num2str(st2.line)]);
           %fclose('all');
           return;
end
```

Procedimiento search_arm4.*m*

```
% Esta rutina se encarga de anotar LOGS para el caso de resultados con tres armonicos.
%
function search_arm4(fichero_log3,fichero_log,fichero_log2);
  % este programa analiza la existencia de dos armonicos. El fundamental +
  % 1 armonico
 try,
 fid=fopen(fichero_log,'r');

 matriz_frec=[];
 %matriz_senal=[];
 matriz_canal=[];
 matriz_frec(1,:)=0;
 matriz_ppio=[];
  matriz_ppio(1,:)=0;
 cont=1;
 cont2=1;
 [datos]=fgetl(fid);
 matriz_senal=struct([]);
  if ischar(datos)
           [descarga,senal,canal,frec,ppio,final] = strread(datos,'%d %s %d %f %f %f');
            [datos]=fgetl(fid);
           while  datos>0
                        nuevasenal=true;
                       %   analizo estadisticas de frecuencia
                       frecuencia=round(frec);
                       [indice,xx]=find(matriz_frec(:,1)==frecuencia);
                       if isempty(indice)
                                    % no ha encontrado coincidencia e
                                    % introduce un nueva frecuencia en
                                    % la matriz
                                    matriz_frec(cont,1)=frecuencia;
                                    matriz_frec(cont,2)=1;
                                    %matriz_senal(cont).senal=senal;
                                   % matriz_senal(cont).cuenta=1;
                                    cont=cont+1;

                       else
                                    % encuentra coincidencia y lo que
                                    % hace es sumar a la frec ya registrada
                                     matriz_frec(indice,2)=matriz_frec(indice,2)+1;


                       end
                       %   analizo la aparicion de señal
                       if isempty(matriz_senal)
                                        matriz_senal(1).senal=upper(senal);
                                            matriz_senal(1).cuenta=1;
                       else
                               [jj,xx]=size(matriz_senal);
                               for j=1:xx
                                         if  isequal(matriz_senal(j).senal,upper(senal))

matriz_senal(j).cuenta=matriz_senal(j).cuenta+1;
                                            nuevasenal=false;
```

```
                                        break;
                                    end

                                end
                            end
                            if nuevasenal
                                   [jj,xx]=size(matriz_senal);
                                   matriz_senal(xx+1).cuenta=1;
                                   matriz_senal(xx+1).senal=senal;
                            end
                            %   analizo intervalos de armonicos
                             ppio=fix(ppio);
                            [indice,xx]=find(matriz_ppio(:,1)==ppio);
                            if isempty(indice)
                                         matriz_ppio(cont2,1)=ppio;
                                         matriz_ppio(cont2,2)=1;
                                         cont2=cont2+1;
                            else
                                         matriz_ppio(indice,2)=matriz_ppio(indice,2)+1;
                            end


                           [datos]=fgetl(fid);
                           if ~ischar(datos),   break,   end
                           [descarga,senal,canal,frec,ppio,final] = strread(datos,'%d %s %d %f %f
%f');

                end;  %del while
  end; %del if
  try, close(fid); catch,end;
  fid3=fopen(fichero_log3,'a');
   [jj,xx]=size(matriz_ppio);
  total=0;
  for j=1:jj
      total=matriz_ppio(j,2)+total;
  end
  for j=1:jj
       texto=strcat('Armonicos_con_comienzo_en..: ',num2str(  matriz_ppio(j,1)
),'..segundos..',...
                        num2str(  (matriz_ppio(j,2)*100)/total   ), '%' );
        fprintf(fid3,'%s\r\n',texto);disp(texto);
  end
 %calculamos los procentajes de las frecuencias
   [jj,xx]=size(matriz_frec);
  total=0;
  for j=1:jj
      total=matriz_frec(j,2)+total;
  end
  for j=1:jj
       texto=strcat('Frecuencia_del_fundamental.:',num2str(matriz_frec(j,1)), '_Khz.......',
num2str(  (matriz_frec(j,2)*100)/total   ), '%' );
        fprintf(fid3,'%s\r\n',texto);disp(texto);
  end

 pause(1);
 total=0;
 [jj,xx]=size(matriz_senal);
 for j=1:xx
        total=matriz_senal(j).cuenta+total;
 end
 %calculamos los procentajes de las señales
 disp('Porcentajes de exito en señales');
 for j=1:xx

        texto=strcat('Positivos_de_la_senal..:',
cell2mat(matriz_senal(j).senal),'....',num2str(    (matriz_senal(j).cuenta*100)/total)
,'%' );
        % disp([ matriz_senal(j).senal,'..:', num2str(    (matriz_senal(j).cuenta*100)/total)
,'%']);
        fprintf(fid3,'%s\r\n',texto);disp(texto);
 end
```

```
   fclose(fid3);
  % fclose('all');
   return;
 catch,

            [msgstr, msgid] = lasterr;
            err=lasterror;
            st1 = err.stack(1,1)
            %st2=err.stack(2,1)
            texto=strcat('Error  search_arm4.........:',msgstr,msgid);
            disp(msgstr);
           disp(msgid);
           disp(['fichero..:',
st1.file]);disp(['Nombre..:',st1.name]);disp(['sentencia..:',num2str(st1.line)]);
           %disp(['fichero..:',
st2.file]);disp(['Nombre..:',st2.name]);disp(['sentencia..:',num2str(st2.line)]);
          % fclose('all');
           return;
end
```

8. BIBLIOGRAFÍA.

P.V.C. Hough, "Methods and Means for Recognizing Complex Patterns", U.S. Patent 3,069,654, 1962.

J.M. Martinez-Otzeta, Combinación de Clasificadores. En Aprendizaje Automático: conceptos básicos y avanzados (Serra, Ed.), Prentice-Hall, 2006.

T.S. Newman and Jain, A.K. (1995). A survey of automated visual inspection. Computer Visión and Image Understanding. 61(2), 231-262.

G. Pajares y J.M. de la Cruz, "Visión por Computador: Imágenes Digitales y Aplicaciones", RAMA, 2001.

G. Pajares and J. M. Cruz, Clasificación de Texturas Naturales mediante K-Means, Revista Electrónica de Visión por Computador http://revc.uab.es/revista/06/, no. 6, pp. 1-18, Ed. Centre de Visio per Computador, Universidad Autónoma Barcelona, 2002.

R. Rud, M. Shoshany, V. Alchanatis and Y. Cohen, "Application of spectral features' ratios for improving classification in partially calibrated hyperspectral imagery: a case study of separating Mediterranean vegetation species", Journal Real-Time Image Processing, vol. 1, pp. 143-152, 2006.

Richard O. Duda and Peter E. Hart. Use of the hough transformation to detect lines and curves in pictures. Commun. ACM, 15(1):11{15, 1972. ISSN 0001-0782. doi: http://doi.acm.org/10.1145/361237.361242.

Rafael C. Gonzalez and Richard E. Woods. Digital Image Processing. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001. ISBN 0201180758.

Rafael C. Gonzalez, Richard E. Woods, and Steven L. Eddins. Digital Image Processing Using MATLAB. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2003. ISBN 0130085197.

G. Pajares, J. M. de la Cruz y V. Moreno, Clasificación de texturas naturales mediante agrupamiento borroso, Ingeniería Civil. Centro de Estudios y Experimentación de Obras Públicas (CEDEX).- Ministerio de Fomento, no 127, pp. 83-89, 2002.

Alan V. Oppenheim, Ronald W. Shafer. Tratamiento de señales en tiempo discreto. Prentice Hall.