

UNIVERSIDAD AUTÓNOMA METROPOLITANA

Unidad Azcapotzalco

División de Ciencias Básicas e Ingeniería

Licenciatura en Ingeniería en Computación

Reporte Final de Proyecto Terminal

“Implementación de una aplicación software para procesamiento de imágenes con wavelets y bancos de filtros paramétricos de reconstrucción perfecta”

Oscar Claudio Vargas

205300960

Contenido

Objetivo General	3
Objetivos Especificos	3
Actividades Realizadas	4
Módulo de Carga de la imagen.....	4
Algoritmo de la Transformada Wavelet Discreta e Inversa	4
Módulo de Despliegue del Espectro de Energía de la Imagen	4
Módulo de Almacenamiento Recuperación y Exportación	5
Problemáticas Resueltas	5
Desplegado de Imágenes	5
Codificación de las Imágenes	5
Dibujar Histograma	6
Almacenamiento en Base de Datos.....	6
Resultados y Conclusiones	6

Reporte Final

Objetivo General

Implementar un programa visual que permita al usuario manipular el espectro de energía y la entropía de una imagen por medio de la transformada *wavelet* discreta con bancos de filtros paramétricos de reconstrucción perfecta.

Objetivos Específicos

- Diseñar una aplicación visual que permita seleccionar el tipo de filtro paramétrico, los valores de los parámetros libres, el límite inferior del umbral de discriminación de coeficientes nulos, y almacenar la información de la imagen procesada en una base de datos relacional con serialización de objetos en tipos de datos BLOB₁.
- Crear el módulo de carga de una imagen en formato PGM₂.
- Implementar el algoritmo de la transformada *wavelet* discreta con bancos de filtros paramétricos de reconstrucción perfecta de longitud cuatro y seis para manipulación de imágenes.
- Implementar el algoritmo de la transformada *wavelet* inversa para reconstruir una imagen con bancos de filtros paramétricos de longitud cuatro y seis.
- Crear el módulo de análisis del espectro de energía de la imagen con detección automática del umbral de eliminación de coeficientes no cero.
- Crear el módulo de análisis de la entropía de la imagen en el dominio tiempo-frecuencia.
- Crear un módulo de almacenamiento, recuperación y exportación de información asociados al procesamiento de una imagen.

Actividades Realizadas

Módulo de Carga de la imagen

La creación de este módulo fue sencilla ya que en lenguaje JAVA ya existe una clase que permite mostrar un explorador de archivos para poder seleccionar el archivo deseado, solo se tiene que implementar el filtro que solo permite que se muestren los tipos de archivos que se requieran, en este caso fueron imágenes PGM. Existen dos tipos de imágenes PGM, P2 y P5, en esta aplicación únicamente se utilizaron imágenes P5 que fueron editadas con el programa GIMP ya que se necesita solo un comentario para que la lectura de la imagen sea correcta.

Algunas partes de este módulo que me dieron ciertos problemas como la codificación de los pixeles de la imagen o al momento de desplegar la imagen. De esto se habla más adelante del documento en la sección de “Problemáticas Resueltas”.

Algoritmo de la Transformada Wavelet Discreta e Inversa

El algoritmo de la transformada wavelet discreta fue implementado para trabajar con vectores ya que los filtros son unidimensionales, ocurre lo mismo para la transformada wavelet inversa. Por esto, además de implementar la transformada y su inversa, se tuvo que implementar una función que separe la matriz de pixeles de la imagen en vectores tanto verticalmente como horizontalmente y aplicar la transformada wavelet discreta y su inversa a estos vectores recursivamente disminuyendo el tamaño de los vectores a los cuales se les aplica la transformación a la mitad mientras que la longitud de los vectores sea la misma longitud de los filtros.

Módulo de Despliegue del Espectro de Energía de la Imagen

Para desplegar el espectro de energía fue necesario meter todos los pixeles de la imagen en un vector y mandarlos a ordenar. El algoritmo que se utilizó para el ordenamiento fue “MergeSort”, ya que, con base en los conocimientos adquiridos en la UEA de Análisis de Algoritmos, este es el algoritmo que en promedio ordena más rápido un arreglo de números aleatorios. Después de mandarlos a ordenar se generó un histograma utilizando la clase Graphics de JAVA.

Módulo de Almacenamiento Recuperación y Exportación

En primer lugar se creó el script que genera la base de datos donde se van a almacenar los datos de la imagen que se desee. Posteriormente se realizó la conexión a la base de datos que permite el almacenamiento y la recuperación de los datos de la imagen, los datos que se guardan son:

- La imagen original
- El nombre de la imagen
- Los parámetros elegidos por el usuario
- El porcentaje de energía

Si se carga una nueva imagen y se desea guardar en la base de datos, se debe elegir el porcentaje de energía y los parámetros de las thetas, una vez que se eligen estas opciones se puede guardar la imagen en la base de datos con el submenú que se encuentra en el menú de archivo.

Para guardar la imagen editada en un lugar específico, se generó un explorador de carpetas donde se debe de elegir el lugar donde se va a almacenar la nueva imagen y el nombre de la imagen.

Problemáticas Resueltas

Despliegado de Imágenes

El primer problema que se me presentó fue el despliegado de la imagen debido a que las imágenes en formato PGM no se son codificadas como imágenes en JAVA si no que son vistas como archivos de texto plano. Después de investigar encontré que la mejor manera seria agregar una API de lenguaje JAVA llamada JAI (Java Advanced Image), esta API reconoce el formato de imagen PGM y permite convertirlas a diversos formatos, de esta forma implemente una función que transforma una imagen en formato PGM a formato JPG.

Codificación de las Imágenes

Existen dos tipos de imágenes PGM las imágenes P2 y las imágenes P5, las primeras contienen los encabezados de las imágenes PGM y los valores de los pixeles están expresados por valores enteros de 0 a 255. Las imágenes P5 traen el valor de los pixeles expresados en bytes, es decir cada pixel es representado por un byte.

El siguiente problema que se me presentó durante el desarrollo del proyecto fue la codificación de los archivos PGM, ya que la aplicación utiliza imágenes P5. Debido a esto tuve que encontrar la manera en que pudiera leer el encabezado de las imágenes PGM como texto y a partir de la quinta línea comenzar a leer en bytes y esto almacenarlo en una matriz y posteriormente transformar los valores de los bytes a datos tipo double ya que para poder aplicar tanto la transformada wavelet como la inversa de la transformada wavelet, se requieren estos tipos de datos.

Reporte Final

Del mismo modo para poder desplegar la imagen que ya fue procesada por la transformada wavelet, es necesario escribir los datos en un archivo con extensión PGM y posteriormente convertirla a una imagen en formato JPG.

Dibujar Histograma

Otro problema a resolver fue la forma de graficar los histogramas de la energía de las imágenes ya que la API JAI contiene una clase Histogram pero esta clase no proporciona la posibilidad de desplegar con facilidad la imagen del histograma generado. Por este motivo decidí dibujar el histograma por medio de líneas en una imagen directamente, con un objeto de la clase Graphics.

Almacenamiento en Base de Datos

Un nuevo problema que resolví fue cuando almacenaba los datos de la imagen en la base de datos, en la propuesta comenté que iba a utilizar datos tipo BLOB pero en el momento de insertar los datos, la información no se guardaba, me di cuenta que los tipos de datos BLOB tienen tamaño de almacenamiento es de 2^{16} y el tamaño máximo de las imágenes que se utilizan es de 512×512 por lo que el tamaño de la imagen es 2^{18} . Por esto el tipo de dato que use para almacenar la imagen es MEDIUMBLOB cuyo tamaño es de 2^{24} .

Resultados y Conclusiones

En el desarrollo del proyecto terminal se obtuvieron los siguientes resultados:

- En la propuesta de proyecto terminal se dijo que la entrada el programa sería una imagen PGM de 512×512 pixeles y que durante el desarrollo del proyecto se decidiría si era posible utilizar imágenes de mayor tamaño, al desarrollar el proyecto me di cuenta que si es posible sin embargo el procesamiento de la imagen es muy lento y si se utilizan imágenes de mayor tamaño, el programa sería demasiado lento como para poder apreciar la manipulación de la imagen.
- Otro punto que se iba a evaluar durante el desarrollo de la aplicación era la posibilidad de utilizar otros tipos de imágenes diferentes a las PGM, la conclusión es que si es posible utilizar otros tipos de imágenes si se agrega un módulo de conversión de imágenes utilizando la API JAI de JAVA.
- La aplicación utiliza imágenes PGM de tamaño 256×256 pixeles e imágenes 512×512 pixeles. En el caso de las imágenes de 256×256 el tiempo de ejecución es muy bueno ya que el despliegue de la imagen editada y la manipulación de los parámetros es prácticamente simultánea.

Reporte Final

En el caso de las imágenes de 512 x 512 el tiempo de ejecución de la aplicación es aceptable ya que en el momento de que el usuario modifica los parámetros libres, la imagen editada tarda un poco en desplegarse, sin embargo es un tiempo razonable tomando en cuenta que se aplica un procesamiento un tanto complicado a la imagen.

- Otro punto que se declaró en la propuesta de proyecto terminal que se iba a evaluar era el tamaño de la imagen de salida. Al comparar el tamaño de la imagen de salida con el tamaño de la imagen de entrada me di cuenta de que son exactamente iguales. Así que puedo concluir que la transformada wavelet no realiza compresión de imagen. Si a la imagen que fue procesada con la transformada wavelet discreta se le aplica un algoritmo de compresión, es posible que la compresión sea mayor ya que aumenta el número de coeficientes cero de la imagen.

UNIVERSIDAD AUTÓNOMA METROPOLITANA

Unidad Azcapotzalco

División de Ciencias Básicas e Ingeniería

Licenciatura en Ingeniería en Computación

Documentación del Proyecto Terminal

“Implementación de una aplicación software para procesamiento de imágenes con wavelets y bancos de filtros paramétricos de reconstrucción perfecta”

Oscar Claudio Vargas

205300960

Índice

VARIABLES	3
CLASES	4
FUNCIONES	5

A continuación se describen las funciones, variables y clases que se utilizaron en el desarrollo de mi proyecto terminal.

VARIABLES

En este apartado del documento se describe el uso que se le dio a cada una de las variables globales que utilice en el proyecto.

- `double h0,h1,h2,h3,h4,h5,g0,g1,g2,g3,g4,g5;`

Este conjunto de variables representan los filtros que se utilizan para implementar la transformada wavelet discreta a un vector de doubles. Son variables globales debido a que sus valores cambian conforme a un JSlider (componente Swing de JAVA) y son utilizados en varias funciones de la aplicación.

- `double lh0,lh1,lh2,lh3,lh4,lh5,lg0,lg1,lg2,lg3,lg4,lg5;`

Este conjunto de variables representan los filtros que se utilizan para implementar la transformada wavelet inversa a un vector de doubles. Son variables globales debido a que sus valores cambian conforme a un JSlider (componente Swing de JAVA) y son utilizados en varias funciones de la aplicación.

- `Imagen original,referencia,editada;`

Estas variables son de tipo Imagen (instancias de una clase que se describe posteriormente). Cada objeto de la clase Imagen representa literalmente a una imagen y cada una de ellas es manipulada en prácticamente toda la aplicación, por eso fueron definidas como variables globales.

- `int filtersize=4;`

Variable que representa el tamaño de los filtros que se utilizan en la aplicación, el valor por defecto es 4 y el usuario tiene la libertad de cambiar su valor a 6 en el momento que él lo decida.

- `double Penergy=0.999999;`

La variable Penergy representa el porcentaje de la energía que se utiliza para la reconstrucción de las imágenes después de haberles aplicado la transformada wavelet discreta. Es una variable global debido a que su valor cambia conforme a un JSlider (componente Swing de JAVA) y es utilizada en varias funciones de la aplicación.

- `String nombre;`

Variable que contiene el nombre completo de una imagen. Fue definida como variable global debido a que se realizan 2 copias de la imagen original y no pueden llamarse de la misma forma, por ende el nombre fue guardado aparte y a cada una de las imágenes se les asignó un nombre distinto.

- `static final String WRITE_OBJECT_SQL = "INSERT INTO InfoWavelets (ID, IMAGEN, NOMBRE, FILTRO, THETA1, THETA2, ENERGIA) "+ "VALUES (null,?,?,?,?,?,?)";`

La variable WRITE_OBJECT_SQL es utilizada para realizar consultas a la base de datos y fue definida como variable global debido a que es de fácil manejo dentro del programa.

- Static String password;

Esta variable representa la contraseña para la conexión a la base de datos

- Static String username;

Esta variable representa el nombre de usuario para la conexión a la base de datos

- Boolean Conex;

Esta variable es la que indica si la conexión fue establecida o no

CLASES

- ```
public class Imagen {
 private String Tipo;
 private String Comentario;
 private int iw;
 private int ih;
 private int nivel;
 private double Matrix [][];
 private char Matriz [][];
}
```

La clase Imagen representa literalmente a una imagen en formato PGM y sus atributos son:

- Tipo: contiene el tipo de imagen que se cargó, en este caso el único tipo de imágenes PGM que se utilizaron fueron P5.
- Comentario: almacena el comentario de la imagen.
- iw: representa el ancho de la imagen
- ih: representa la altura de la imagen
- nivel: representa el nivel máximo de gris que puede tener un pixel
- Matrix: es la matriz de pixeles representada en números doubles
- Matriz: es la matriz de pixeles representada por caracteres

- ```
public class RadioListener implements ActionListener {
    public void actionPerformed(ActionEvent e) {}
}
```

La clase RadioListener es una implementación de la clase ActionListener ya definida en JAVA, esta clase contiene el método actionPerformed que recibe como parámetro un ActionEvent, esto implica que cuando se registre un evento dentro del buttonGroup, dependiendo del RadioButton que se haya seleccionado se realizarán distintas tareas dentro de la aplicación.

FUNCIONES

- `private void AbrirActionPerformer(ed(java.awt.event.ActionEvent e)`

Esta función recibe como parámetro un `ActionEvent` y es llamada cuando se carga una nueva imagen, en esta función hace la llamada a la función para desplegar la imagen original y la llamada a la función de la transformación de la imagen de referencia.

- `private void ImprimeImagenReferencia()`

Esta función es la encargada de transformar la imagen de referencia (imagen a la cual se le aplica la transformada wavelet discreta con los filtros propuestos por Daubechies) en formato PGM en formato JPG para que se pueda desplegar en la pantalla.

- `private void ImprimeImagenEditada()`

Esta función es la encargada de transformar la imagen de editada (imagen a la cual se le aplica la transformada wavelet discreta con los filtros calculados mediante los parámetros elegidos por el usuario) en formato PGM en formato JPG para que se pueda desplegar en la pantalla.

- `private void ReferenciaWavelet4(double im[], int n)`

Esta función es la implementación de la transformada wavelet discreta con los filtros de longitud cuatro propuestos por Daubechies, recibe un vector de doubles y un entero N que indica la longitud del vector.

- `private void TransformadaWavelet4(double im[], int n)`

Esta función es la implementación de la transformada wavelet discreta con los filtros de longitud cuatro que fueron calculados mediante los parámetros elegidos por el usuario, recibe un vector de doubles y un entero N que indica la longitud del vector.

- `private void ReferenciaWavelet6(double im[], int n)`

Esta función es la implementación de la transformada wavelet discreta con los filtros de longitud seis propuestos por Daubechies, recibe un vector de doubles y un entero N que indica la longitud del vector.

- `private void TransformadaWavelet6(double im[], int n)`

Esta función es la implementación de la transformada wavelet discreta con los filtros de longitud seis que fueron calculados mediante los parámetros elegidos por el usuario, recibe un vector de doubles y un entero N que indica la longitud del vector.

- `private void invTransformadaWavelet4(double im[], int n)`

Esta función es la implementación de la transformada wavelet inversa con los filtros de longitud cuatro que fueron calculados mediante los parámetros elegidos por el usuario, recibe un vector de doubles y un entero N que indica la longitud del vector.

- `private void invReferenciaWavelet4(double im[], int n)`

Esta función es la implementación de la transformada wavelet inversa con los filtros de longitud cuatro propuestos por Daubechies, recibe un vector de doubles y un entero N que indica la longitud del vector.

- `private void invReferenciaWavelet6(double im[], int n)`

Esta función es la implementación de la transformada wavelet inversa con los filtros de longitud seis propuestos por Daubechies, recibe un vector de doubles y un entero N que indica la longitud del vector.

- `private void invTransformadaWavelet6(double im[], int n)`

Esta función es la implementación de la transformada wavelet inversa con los filtros de longitud seis que fueron calculados mediante los parámetros elegidos por el usuario, recibe un vector de doubles y un entero N que indica la longitud del vector.

- `private void GuardarImagen(java.awt.event.ActionEvent e)`

Función que es llamada cuando se desea guardar la imagen editada en algún lugar específico. La función recibe como parámetro un `ActionEvent`, cuando se manda llamar a esta función aparecerá un explorador de archivos donde se debe indicar la ubicación donde se desea guardar y el nombre.

- `private void GuardarActionPerformerred(java.awt.event.ActionEvent e)`

Función que es llamada cuando se desea guardar la imagen editada en la base de datos. La función recibe como parámetro un `ActionEvent`, y si la conexión a la base de datos no falla y los datos están completos, se almacenara en la base de datos toda la información necesaria para poder volver a cargar la imagen.

- `private void AbrirBaseActionPerformerred(java.awt.event.ActionEvent e)`

Función que es llamada cuando se desea cargar una imagen desde la base de datos. La función recibe como parámetro un `ActionEvent`, cuando se manda llamar a esta función aparecerá un cuadro de dialogo donde se debe ingresar parte del nombre de la imagen que se desea cargar, la función realizara la consulta a la base de datos y manda llamar a la función que desplegará la información en caso de que sea encontrada.

- `private void Manual(java.awt.event.ActionEvent e)`

Esta función recibe como parámetro un `ActionEvent`, la función es la encargada de mostrar el manual de usuario de la aplicación.

- `private void desplegar(int filter,double th1,double th2,double energy)`

La función desplegar es la encargada de hacer la construcción de la información cuando se mandó a buscar una imagen en la base de datos y muestra los datos encontrado en pantalla.

- `private void histogramaOriginal()`

Función que es la encargada de dibujar el histograma de la imagen original.

- `private void histogramaEditada()`

Función que es la encargada de dibujar el histograma de la imagen editada.

- `private void copiaImagen(Imagen imagen1,Imagen original)`

Debido a que en lenguaje java todos los objetos son representados por referencias esta function es la encargada de clonar la imagen original en una otra imagen existente.

- `private void implementaReferenciaWavelet4()`

Esta función es la encargada de descomponer la matriz de pixeles (doubles) de una imagen en vectores horizontales y verticales, manda llamar a la función donde se encuentra implementada la transformada wavelet discreta con los filtros de longitud 4 propuestos por Daubechies y a su inversa.

- `private void implementaReferenciaWavelet6()`

Esta función es la encargada de descomponer la matriz de pixeles (doubles) de una imagen en vectores horizontales y verticales, manda llamar a la función donde se encuentra implementada la transformada wavelet discreta con los filtros de longitud 6 propuestos por Daubechies y a su inversa.

- `private void implementaWavelet4()`

Esta función es la encargada de descomponer la matriz de pixeles (doubles) de una imagen en vectores horizontales y verticales, manda llamar a la función donde se encuentra implementada la transformada wavelet discreta con los filtros de longitud 4 que fueron calculados mediante los parámetros elegidos por el usuario y a su inversa.

- `private void implementaWavelet6()`

Esta función es la encargada de descomponer la matriz de pixeles (doubles) de una imagen en vectores horizontales y verticales, manda llamar a la función donde se encuentra implementada la transformada wavelet discreta con los filtros de longitud 6 que fueron calculados mediante los parámetros elegidos por el usuario y a su inversa.

- `private void CopiaMatricesRef()`

Función que realiza la conversión de la matriz de pixeles representados por doubles de la imagen de referencia en la matriz de pixeles representados por caracteres de la misma imagen de referencia.

- `private void CopiaMatrices()`

Función que realiza la conversión de la matriz de pixeles representados por doubles de la imagen editada en la matriz de pixeles representados por caracteres de la misma imagen editada.

- `private void siftDown(double numbers[], int root, int bottom)`

Función que implementa el algoritmo de MergeSort, recibe un arreglo de doubles, el límite izquierdo y el límite derecho del arreglo.

- `private void Sort(double numbers[], int array_size)`

Función que recibe un arreglo de doubles y su longitud, esta función es llamada por las funciones que dibujan los histogramas y a su vez esta función manda llamar a la función donde se implementa el algoritmo de MergeSort.

- `private int min(int a, int b)`

Esta función se usa para calcular el mínimo entre dos números, es mandada llamar para recortar un número cuando tiene una longitud muy grande y no es posible desplegarlo.

- `private void SliderTheta1StateChanged(javax.swing.event.ChangeEvent evt)`

Esta función es la encargada de calcular los filtros de longitud 4 para la transformada wavelet y su inversa mediante los parámetros elegidos por el usuario en un JSlider.

- `private void SliderTheta2StateChanged(javax.swing.event.ChangeEvent evt)`

Esta función es la encargada de calcular los filtros de longitud 6 para la transformada wavelet y su inversa mediante los parámetros elegidos por el usuario en un JSlider.

- `private void SliderEnergiaStateChanged(javax.swing.event.ChangeEvent evt)`

Esta función cambia el valor de la variable global Penergy mediante un JSlider manipulado por el usuario.

UNIVERSIDAD AUTÓNOMA METROPOLITANA

Unidad Azcapotzalco

División de Ciencias Básicas e Ingeniería

Licenciatura en Ingeniería en Computación

Manual de usuario para el Proyecto Terminal

“Implementación de una aplicación software para procesamiento de imágenes con wavelets y bancos de filtros paramétricos de reconstrucción perfecta”

Oscar Claudio Vargas

205300960

Índice

Introducción.....	2
Menú Archivo.....	4
Submenú Abrir Nueva Imagen.....	5
Submenú Guardar Como.....	7
Submenú Guardar en Base de Datos.....	8
Submenú Abrir de Base de Datos.....	10

Introducción

Este proyecto fue desarrollado en lenguaje JAVA en el entorno de programación NetBeans IDE 6.9.1 bajo la plataforma de Linux. El manejador de base de datos que se usó es MySQL.

En el proyecto se utilizó un API de programación llamado JAI (Java Advanced Image), por lo tanto es necesario agregar algunas bibliotecas al proyecto, estas bibliotecas se encuentran en la carpeta "Bibliotecas". A continuación se enlistan las bibliotecas necesarias:

- jai_codec.jar
- jai_imageio.jar
- jai_core.jar
- jai_linux-i586.jar
- mlibwrapper_jai.jar
- mysql-connector-java-5.0.8-bin.jar

Una vez que se agregaron los archivos .jar antes mencionados, se procede a compilar y ejecutar el proyecto. La Imagen 1 muestra la pantalla principal.



Imagen 1

En esta pantalla se puede apreciar que existen dos opciones para la transformada wavelet, filtros de longitud 4 y filtros de longitud 6, la opción seleccionada por defecto es la de filtros de longitud 4. En esta opción existen 2 parámetros que pueden ser controlados por el usuario, Theta1 y el porcentaje de energía para la imagen nueva. Los filtros h_0 , h_1 , h_2 y h_3 son calculados por medio de ecuaciones que son funciones de theta1.

Si se selecciona la opción de filtros de longitud 6, aparecerá un nuevo parámetro que puede ser manipulado por el usuario, Theta2. Esto se muestra en la Imagen 2.

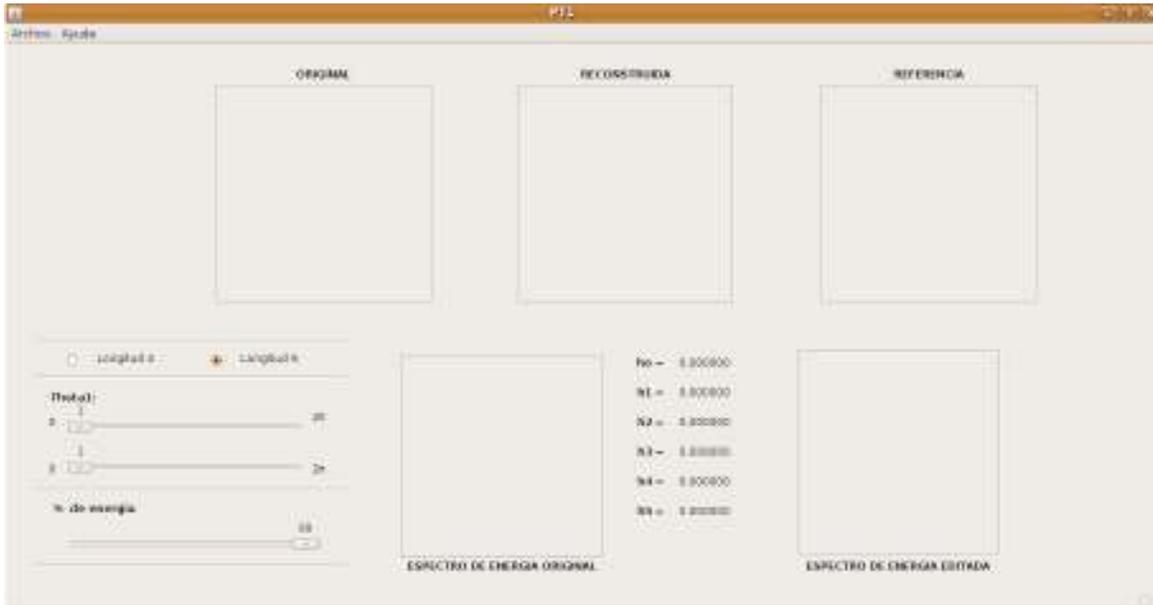


Imagen 2

Al seleccionar esta opción aparecerán también dos filtros más, h4 y h5, estos dos filtros son calculados por medio de ecuaciones que son funciones de theta1 y theta2, los filtros h0, h1, h2 y h3 ya no son calculados a través de las ecuaciones anteriores, ahora son calculados por unas nuevas que también son funciones de theta1 y de theta2.

Menú Archivo

En la parte superior izquierda de la pantalla se encuentra el menú Archivo, dentro de este menú se encuentran las siguientes opciones (Imagen 3):

- Abrir Nueva Imagen
- Abrir de Base de Datos
- Guardar Como
- Guardar en base de datos
- Salir



Imagen 3

Submenú Abrir Nueva Imagen

Para cargar una imagen que se encuentra almacenada en algún lugar en nuestra computadora se debe dar clic sobre el submenú “Abrir Nueva Imagen” o pulsar Ctrl+N. En seguida aparecerá un explorador de archivos como se muestra en la Imagen 4.

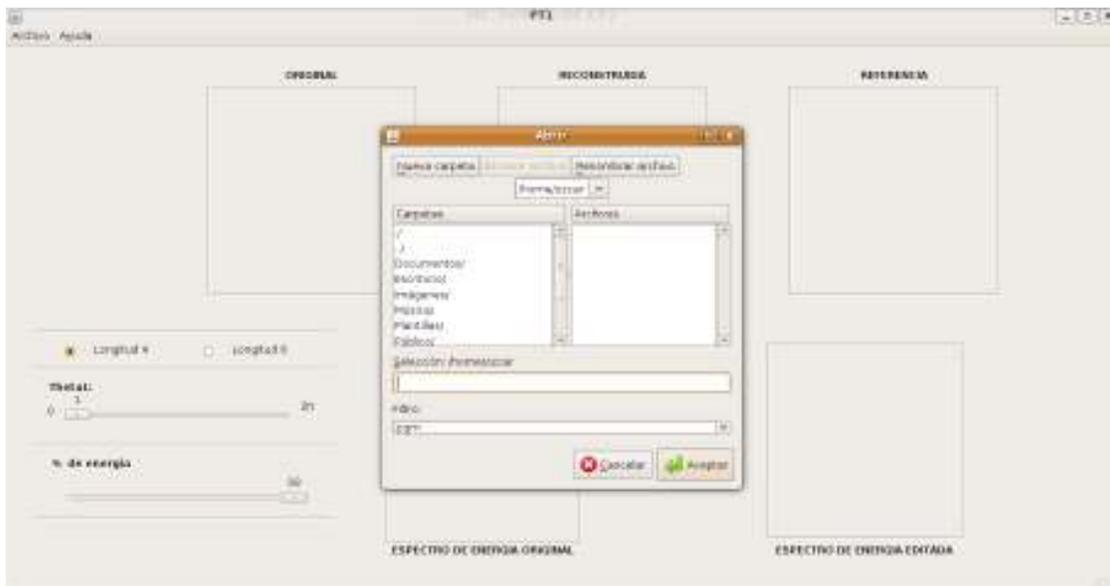


Imagen 4

Al seleccionar la imagen que se desee y dar clic sobre el botón de aceptar, se cargara la imagen original en el recuadro superior izquierdo. En el recuadro superior derecho aparecerá la imagen reconstruida y con los filtros propuestos por Daubechies, de longitud 4 o 6 según sea la opción que se haya seleccionado (Imagen 5).



Imagen 5

El valor por defecto del porcentaje de energía es 99.9999%, para algunas imágenes con los filtros propuestos por Daubechies no se alcanza este nivel de energía por lo tanto se mostrara un mensaje que informa al usuario de este hecho, en este caso se debe bajar un poco el nivel de la energía y se debe cargar de nuevo la imagen. (Imagen 6)

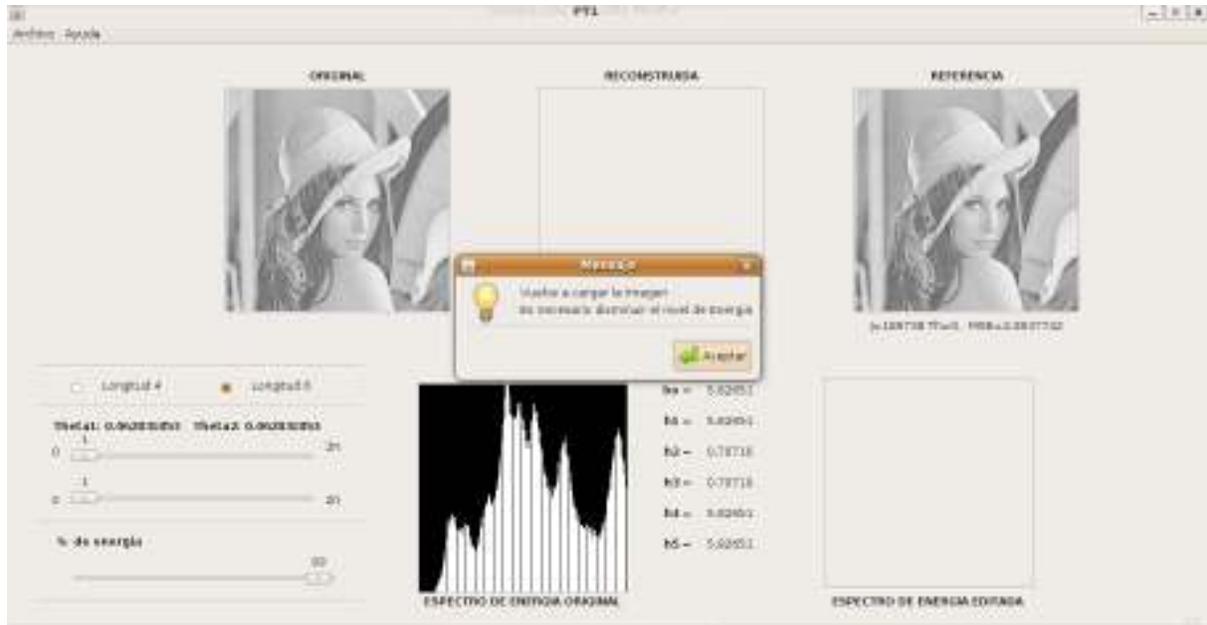


Imagen 6

En el recuadro inferior de la derecha aparece el histograma de la energía de la imagen original, es decir, se grafican los pixeles de la imagen cargada originalmente. En el recuadro superior del centro aparecerá la imagen editada con los parámetros que el usuario haya elegido, del mismo modo aparecerá el histograma de los pixeles, previamente ordenados, en el recuadro inferior derecho al momento de deslizar las barras de los parámetros theta. Lo anterior se muestra en la Imagen 7.

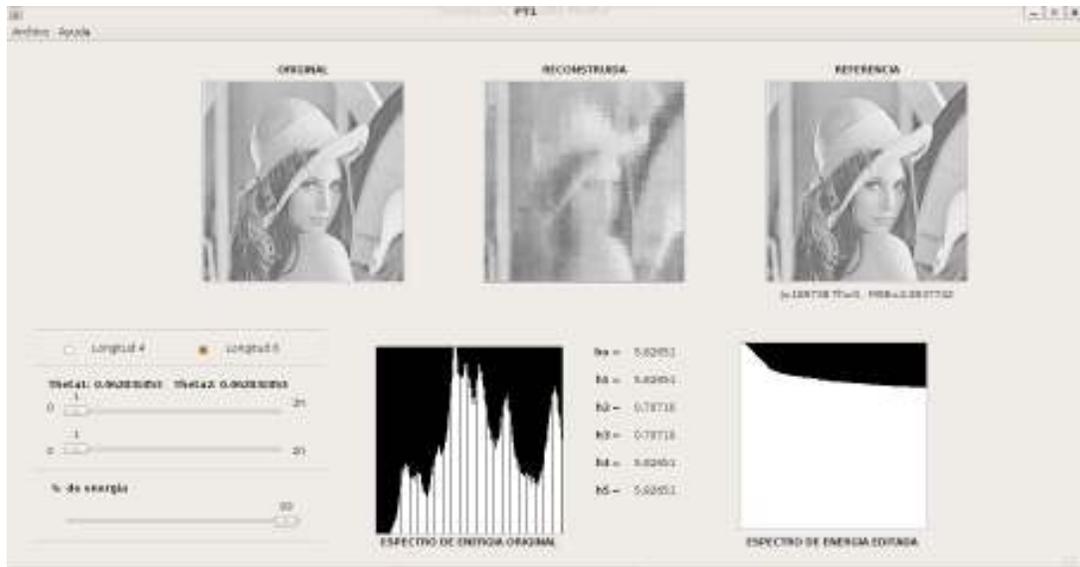


Imagen 7

Submenú Guardar Como

Si se desea guardar la imagen reconstruida con los parámetros que eligió el usuario, se debe dar clic sobre el submenú “Guardar Como” o presionar Ctrl + W, al elegir esta opción aparecerá un explorador de archivos donde se debe elegir la carpeta donde se desea guardar la imagen, teclear el nombre de la nueva imagen, y dar clic en el botón de Guardar. Como se muestra en la Imagen 8.

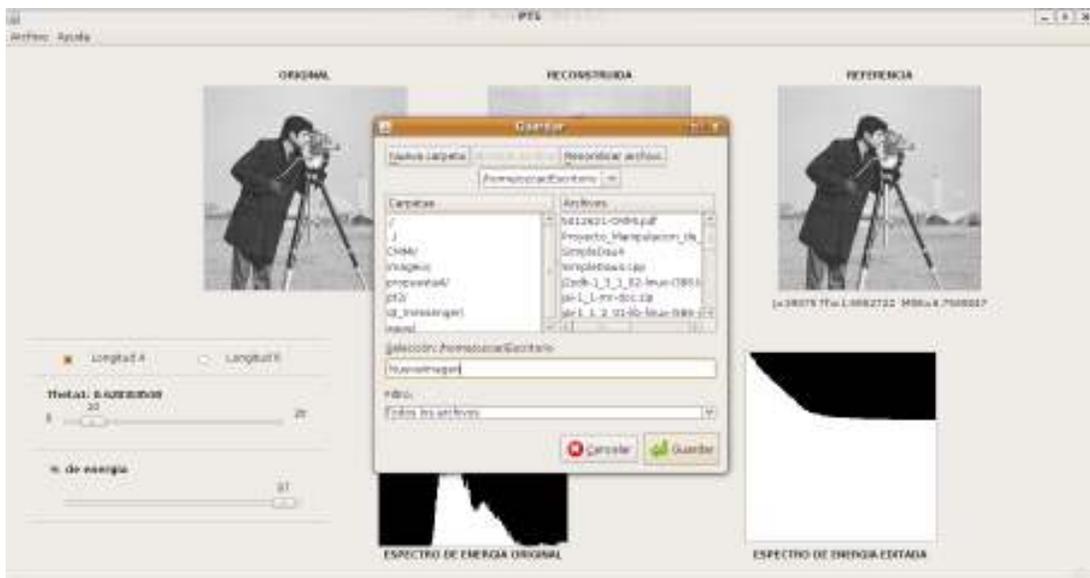


Imagen 8

Submenú Guardar en Base de Datos

Antes de guardar una imagen en la base de datos, debe cerciorarse que la base exista, si no existe se debe ejecutar el script “Database_Proyecto.sql” que es el archivo que genera la base de datos, una vez que se haya ejecutado sin errores el script, se debe configurar la conexión a la base mediante un usuario y una contraseña. Esta configuración aparecerá al seleccionar el submenú “Guardar en Base de Datos” o el submenú “Abrir de Base de Datos” si la conexión no se ha establecido. En primer lugar se mostrara un mensaje como el que aparece en la Imagen 9.



Imagen 9

En seguida aparecerá un cuadro de dialogo (Imagen 10) donde se debe ingresar el usuario y la contraseña para poder establecer la conexión a la base de datos.

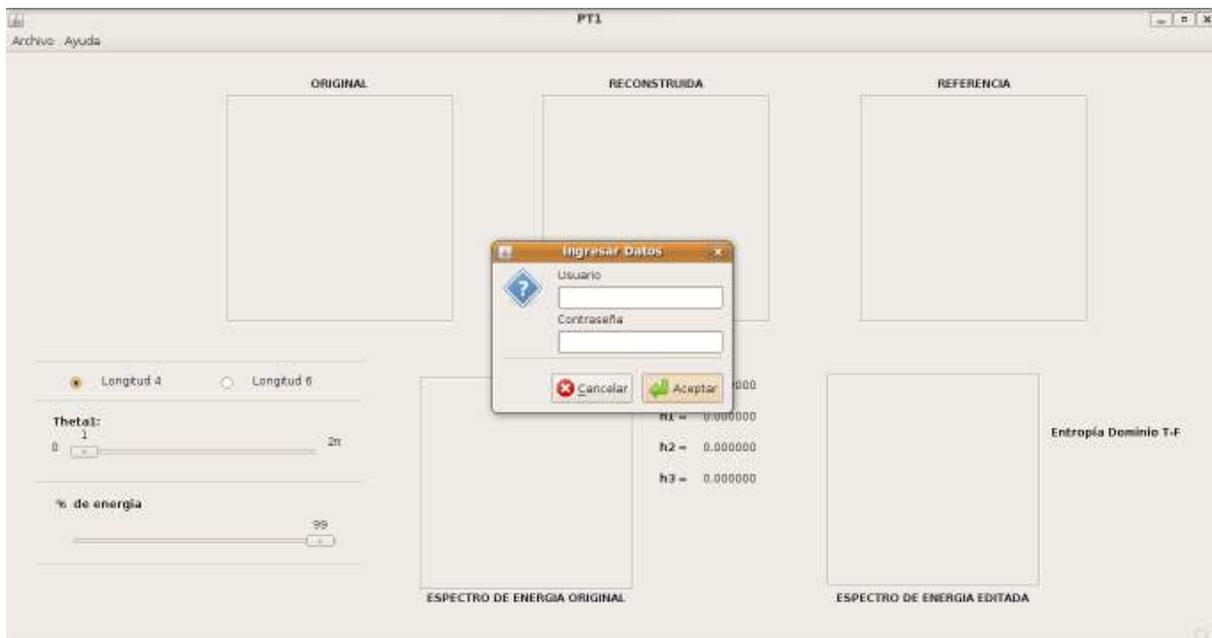


Imagen 10

Una vez que se hayan ingresado los datos, se debe dar clic sobre el botón de aceptar, si los datos son incorrectos aparecerá un mensaje como el de la imagen 11.



Imagen 11

Si los datos ingresados son correctos, aparecerá un mensaje indicando que la conexión a la base de datos fue establecida (Imagen 12).



Imagen 12

Una vez que la conexión se haya configurado ya sea en el submenú de “Guardar en Base de datos” o en el submenú “Abrir de Base de Datos”, será posible utilizar libremente esto submenús.

NOTA: Los datos ingresados no se almacenan, cada vez que se abra la aplicación se deben ingresar los datos para establecer la conexión.

Si se desea guardar la información de una imagen en una base de datos, una vez que se haya seleccionado la imagen, los tipos de filtros, los parámetros deseados y el porcentaje de energía, se debe dar clic en el submenú de “Guardar en Base de Datos” o presionar Ctrl+S, si no existe ningún problema con la inserción en la base de datos se mostrara el mensaje siguiente (Imagen 13):

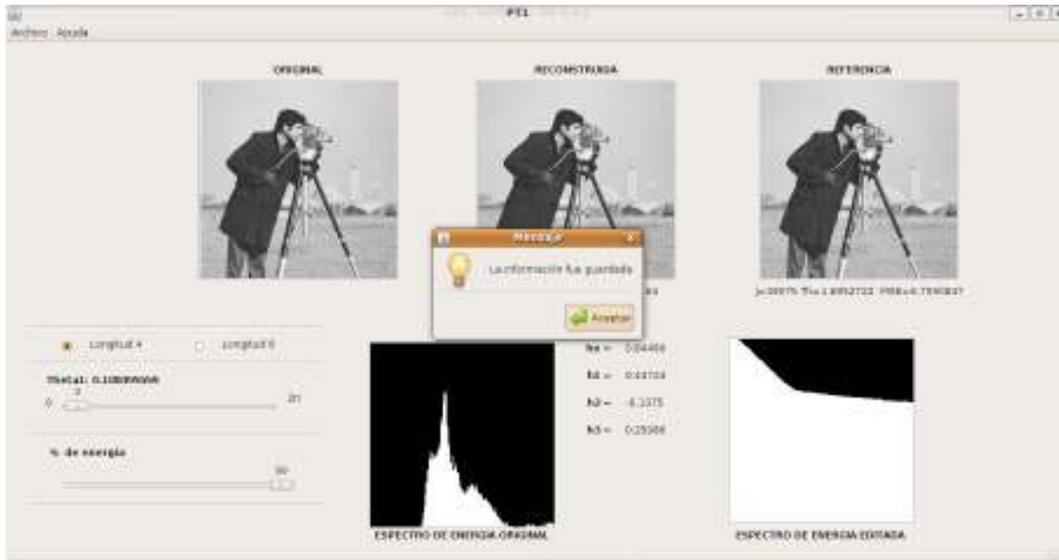


Imagen 13

Sin embargo si existe algún problema con la base de datos se mostrara el mensaje de la Imagen 14.

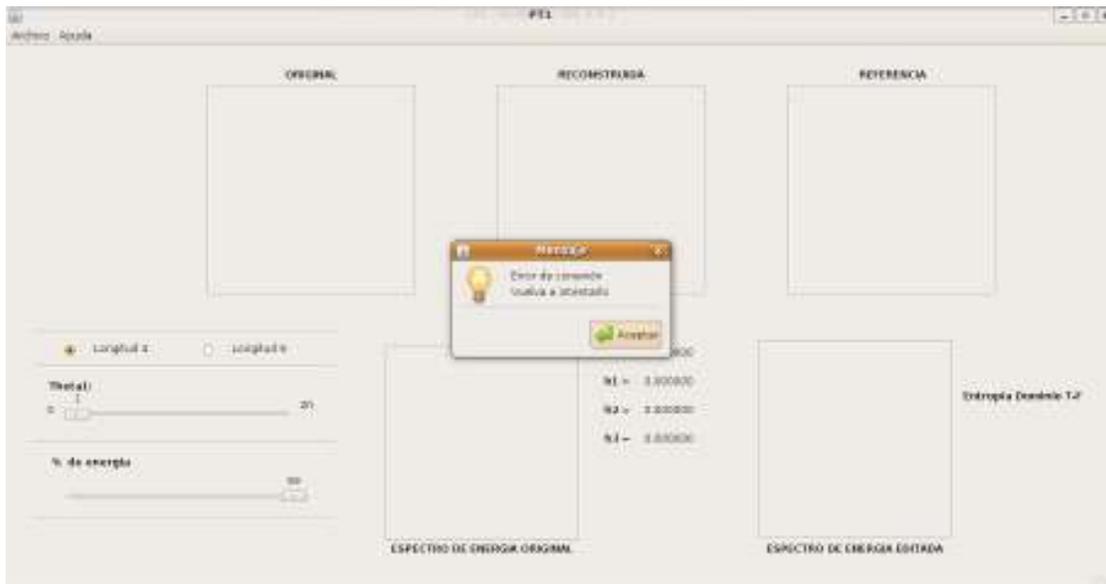


Imagen 14

Si se muestra el mensaje anterior, se debe verificar:

- Que la base de datos exista y se llame “Proyecto”
- Que dentro de esta base de datos exista una tabla llamada “InfoWavelets” y que contenga las siguientes columnas:
 - ID (Tipo int)
 - IMAGEN (Tipo medumblob)
 - NOMBRE (Tipo texto)
 - FILTRO (Tipo int)
 - THETA1 (Tipo double)
 - THETA2 (Tipo double)
 - ENERGIA (Tipo double)
- Que la base de datos se encuentre en el manejador MYSQL

Submenú Abrir de Base de Datos

Si la conexión a la base de datos ya se ha establecido, se podrá abrir una imagen que se encuentre guardada en la base de datos, de lo contrario, aparecerá un recuadro como el de la Imagen 9.

La opción para poder recuperar la información de una imagen desde la base de datos se encuentra en el menú de archivo, seleccionando el submenú “Abrir de Base de Datos”, al dar clic sobre este menú o presionando Ctrl+A aparecerá un pequeño recuadro (Imagen 15) donde se debe ingresar el nombre de la imagen que se busca.



Imagen 15

Si los datos de la imagen existen en la base de datos, todos los datos guardados serán desplegados en pantalla tal como se mostró en la Imagen 7.

Si la imagen no se encontró en la base de datos se mostrara un mensaje como el que se muestra en la Imagen 16.



Imagen 16