ESCUELA SUPERIOR POLITECNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

“Análisis e Implementación de algoritmos para el mejoramiento de Imágenes de Radiografías en busca de Detalles”

Informe de Materia de Graduación

Previa a la obtención del título de:

Ingeniero Electrónico y Telecomunicaciones

Presentado por:

Carlos Julio Guzmán Real

Erick Ermory González Linch

Guayaquil-Ecuador

2009

DEDICATORIA

A Dios, a mis padres, abuelos, a mi abuelita Rosario una de las principales figura en mi crianza, a mis hermanos, tíos y a mis compañeros ya que con su dedicación y apoyo me han guiado por el camino correcto de la vida, ayudándome a lograr mis objetivos propuestos.

Carlos Julio Guzmán

El siguiente trabajo es dedicado a Dios y a mis padres por el apoyo incondicional recibido durante mi formación profesional y humana.

Erick González

DECLARACION EXPRESA

"La responsabilidad del contenido de este Proyecto nos corresponde exclusivamente; y el patrimonio intelectual de la misma a la ESCUELA SUPERIOR POLITECNICA DEL LITORAL"

(Reglamento de Exámenes y títulos profesionales de la ESPOL)

Carlos Julio Guzmán

Erick González Linch

Resumen

El tema de la calidad de las radiografías es un asunto de gran importancia para la medicina encargada de estudiar los traumas con los que se ve afectado el cuerpo humano, su aguda visualización es de vital importancia para poder realizar un correcto diagnostico y por consiguiente tomar las medidas correctivas necesarias, ya que de lo contrario se podría derivar en serios problemas ocasionados simplemente por un diagnostico erróneo.

Este informe analiza y prueba brevemente algunos algoritmos con el fin de acentuar y esclarecer rasgos imperceptibles de ciertas radiografías para ayudar en el diagnostico de las mismas, utilizando el procesamiento digital de imágenes.

Se desarrollará una aplicación en Matlab, principalmente basada en la aplicación de mascaras y filtros para mejorar la detección de bordes, además de otras herramientas como ajuste de curva las cuales ayudaran a mejorar la imágenes de radiografía, se espera que esta aplicación sea de fácil uso para cualquier usuario, y este sea capaz de obtener los resultados deseados.

INDICE GENERAL

INDICE DE FIGURAS

INTRODUCCIÒN

El procesamiento digital de imágenes es el conjunto de técnicas que se aplican a las imágenes digitales con el objetivo de mejorar la calidad o facilitar la búsqueda de información en dichas imágenes.

De manera específica nuestro objetivo es realizar un procesamiento que permita identificar detalles, prácticamente imperceptibles a simple vista, lo que sin duda será de gran ayuda en el área de la medicina.

Los filtros y ajustes son las principales herramientas para lograr nuestro objetivo, ya que mediante su uso podremos detectar bordes, modificar es aspecto de las imágenes o sencillamente eliminar ruido de las mismas, nuestro proyecto los utilizará para mejorar los detalles en los bordes de los huesos tomados en radiografías, con fin de proporcionar al doctor una mejor imagen que lo ayude en su diagnostico.

Para el desarrollo del proyecto utilizaremos MATLAB V7.7.0.471 (R2008b) ya que es un programa interactivo para computación numérica y visualización de datos. Es ampliamente usado por Ingenieros de Control en el análisis y diseño, además tiene una extraordinaria versatilidad y capacidad para resolver problemas en matemática aplicada, física, química, ingeniería, finanzas y muchas otras aplicaciones. Está basado en un sofisticado software de matrices para el análisis de sistemas de ecuaciones. Permite resolver complicados problemas numéricos sin necesidad de escribir un programa.

CAPITULO 1

1. FILTRADO DE IMAGENES

Los filtros son operaciones que se aplican a los píxeles de una imagen digital para mejorarla, resaltar cierta información o conseguir un efecto especial.

El filtrado digital de imágenes es una operación de convolución de la imagen original con la función del filtro, la operación de convolución entre las funciones $f_1(t)$ y $f_2(t)$ se define de la siguiente manera:

$$f(t) = f_1(t) * f_2(t) = \int_{-\infty}^{\infty} f_1(x) f_2(t-x) dx$$

1.1 Dominio espacial y de la frecuencia

Las técnicas de procesamiento de punto o de convolución que operan sobre la imagen en el dominio espacial calculan los valores sobre los propios valores de los píxeles, a diferencia de los del dominio de la frecuencia que la traducen primero a un mapa de frecuencias mediante una transformada de tipo Fourier.

Nuestra acción será sobre el filtrado espacial, el mismo que se realiza definiendo un entorno con los vecinos de un píxel central. Este entorno se denomina ventana, máscara o matriz de convolución, y suele ser cuadrado o rectangular, aunque es posible definir otras formas rómbicas, circulares, etc.

A cada posición en la ventana se le asigna un peso o participación en el cálculo que dará el nuevo valor para el píxel central. Entonces, se va desplazando la máscara, centrándola en cada uno de los píxeles de la imagen. Al recorrido completo es a lo que denominamos filtrado, y los sucesivos resultados, siempre a partir de los valores originales, forman la nueva imagen. Es obvio que en los bordes y esquinas la ecuación varía, ya que parte de la máscara se pierde ([1]).

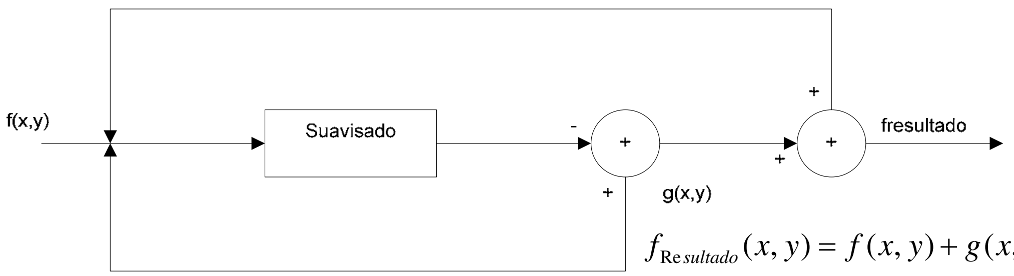
Es importante saber que los ejes o bordes de una imagen se encuentran en zonas donde el nivel de intensidad cambia bruscamente, cuanto más rápido se produce el cambio de intensidad, el borde es más fuerte. Un buen proceso de detección de bordes facilita la elaboración de las fronteras de objetos con lo que, el reconocimiento de objetos se simplifica.

[1] www.quesabesde.com . Paulo Porta, Técnicas de Filtrado
http://www.quesabesde.com/camdig/articulos.asp?articulo=137

1.2 Filtros Utilizados

1.2.1 Unsharp

El filtro Unsharp es un simple operador que busca nitidez, y es una de las funciones dentro de 'fspecial' de MATLAB, su nombre se infiere debido al hecho que aumenta los bordes además de algunos componentes de altas frecuencias de la imagen. Mediante un procedimiento resta una versión alisada de la imagen original, lo que aumenta considerablemente la nitidez del resultado ([2]).



$$f_{\mathrm{Re}\,sultado}(x, y) = f(x, y) + g(x, y)$$

Fig. 1 Filtro Unsharp.

-0.1667	-0.6667	-0.1667
-0.6667	4.3333	-0.6667
-0.1667	-0.6667	-0.1667

Máscara del Filtro Unsharp

[2] http://homepages.inf.ed.ac.uk/rbf/HIPR2/unsharp.htm

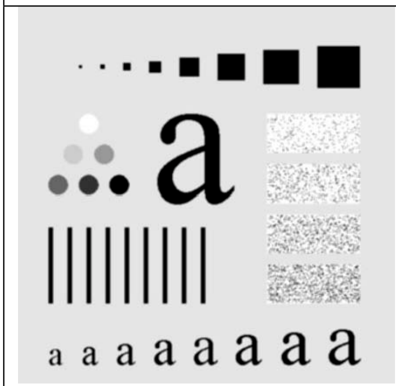
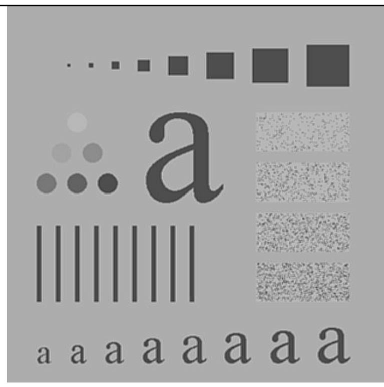
Figura Original	Figura Unsharp
	

Fig. 2 Comparación Unsharp.

1.2.2 Filtro Emboss

El filtro Emboss, conocido como de “Repujado” hace que una imagen aparezca estampada debido a la supresión del color en la imagen y marcando sus bordes con negro ([3])

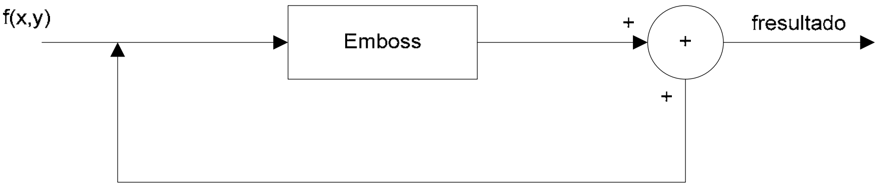


Fig. 3 Filtro Emboss

$$f_{\mathrm{Re}\,sultado}(x, y) = f(x, y) + f(x, y) * H(x, y)$$

[3] http://docs.gimp.org/en/plug-in-emboss.html

-2	-1	0
-1	1	1
0	1	2

Máscara del Filtro Emboss

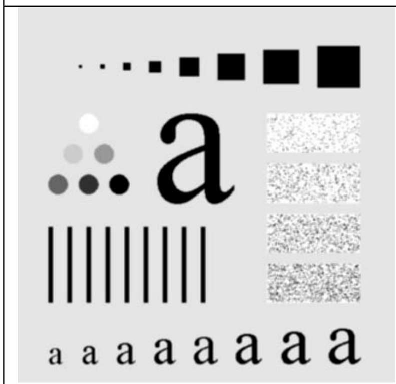
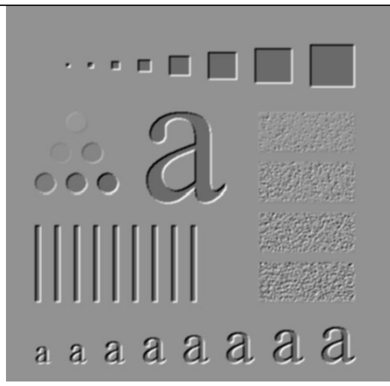
Figura Original	Figura Emboss
	

Fig. 4 Comparación Emboss.

1.2.3 Filtro de Realce 1 y 2

Dos matrices son utilizadas para filtrar la imagen original, y de esta manera realizar el realce de la misma. Se puede ver cómo estas máscaras tienen un efecto de afilado de los bordes y mejora de los detalles borrosos sin que cambie la apariencia global de la imagen ([4])

[4] http://www.des.udc.es/~adriana/TercerCiclo/CursoImagen/curso/web/Filtrado_Espacial_Real.html

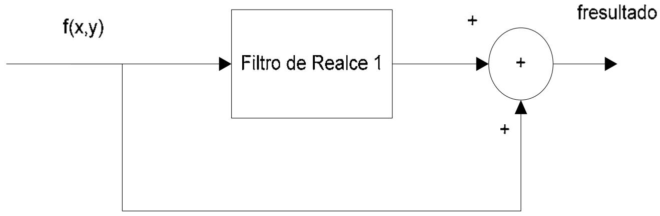


Fig. 5 Filtro de Realce 1.

-1	-1	-1
-1	9	-1
-1	-1	-1

Máscara del Filtro de Realce 1

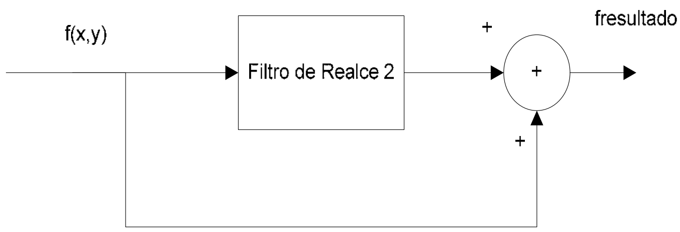


Fig. 6 Filtro de Realce 2.

-0.1429	-0.1176	-0.1429
-0.1176	2.7143	-0.1176
-0.1429	-0.1176	-0.1429

Máscara del Filtro de Realce 2

$$f_{\mathrm{Re}\,sultado}(x, y) = f(x, y) + f(x, y) * H(x, y)$$

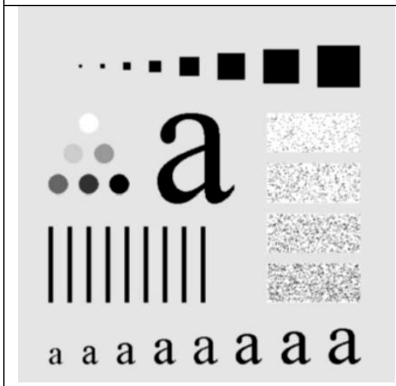
Figura Original	Figura Realce 1
	

Fig. 7 Comparación Filtro de Realce 1.

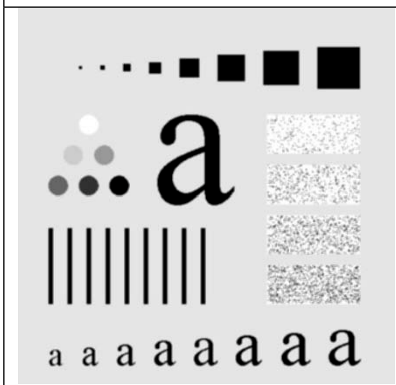
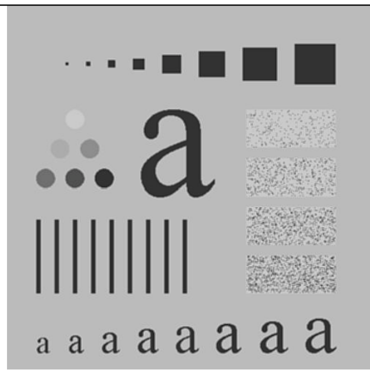
Figura Original	Figura Realce 2
	

Fig. 8 Comparación Filtro de Realce 2.

1.2.4 Roberts, Prewitt y Sobel

Podemos realizar una clasificación general de los bordes según sea su dirección en:

- *Bordes verticales*, cuando píxeles conectados verticalmente tienen valores diferentes respecto de los anteriores o posteriores.

- *Bordes horizontales*, cuando tenemos píxeles conectados horizontalmente, y estos tienen distintos valores respecto de los anteriores o posteriores.

- *Bordes oblicuos,* cuando tenemos una combinación de las componentes horizontales y verticales [5].

Los operadores de **Prewitt**, **Sobel** y **Roberts** son filtros diferenciales de primer orden y son considerados operadores dobles o de dos etapas. La detección de bordes se realiza en dos pasos, en el primero se aplica una máscara para buscar bordes horizontales, y en el segundo paso buscamos los verticales, el resultado final es la suma de ambos. Se muestran algunas máscaras de convolución comunes a continuación. Los

5 http://www.des.udc.es/~adriana/TercerCiclo/CursoImagen/curso/web/Filtrado_Espacial_Borde.html

detectores de fila (horizontales) son ***H_h*** y los detectores de columna (verticales) son ***H_v*** ([6]).

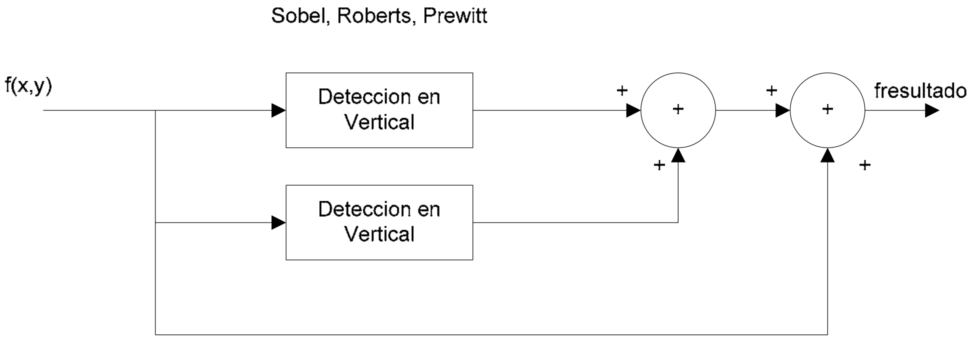


Fig. 9 Filtros de Roberts, Prewitt y Sobel.

$$f_{\mathrm{Re}\,sultado}(x,y) = f(x,y) + ABS[f(x,y) * H_X(x,y)] + ABS[f(x,y) * H_Y(x,y)]$$

	Roberts	Prewitt	Sobel
H_h	$\begin{matrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{matrix}$	$\begin{matrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{matrix}$	$\begin{matrix} 1 & 0 & -1 \\ 2 & 1 & -2 \\ 1 & 0 & -1 \end{matrix}$
H_v	$\begin{matrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{matrix}$	$\begin{matrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{matrix}$	$\begin{matrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{matrix}$

[6] http://www.des.udc.es/~adriana/TercerCiclo/CursoImagen/curso/web/Filtrado_Espacial_Borde.html

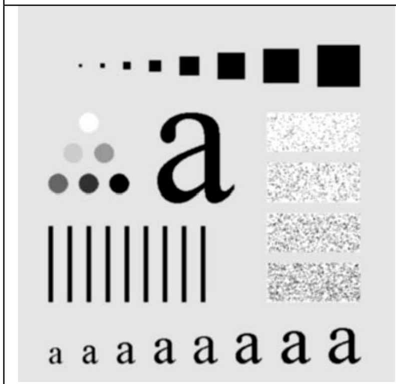
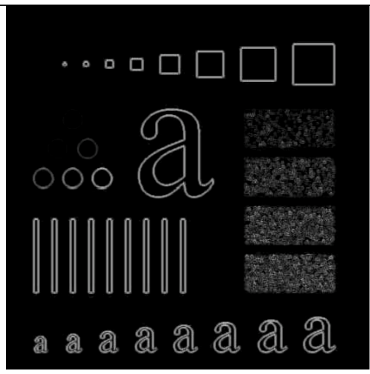
Figura Original	**Figura Roberts**
	

Fig. 10 Comparación Roberts.

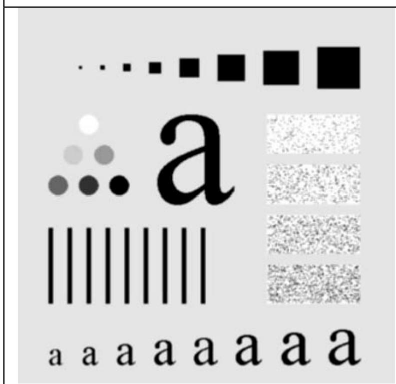
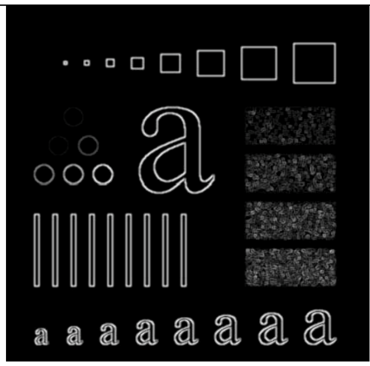
Figura Original	**Figura Prewitt**
	

Fig. 11 Comparación Prewitt.

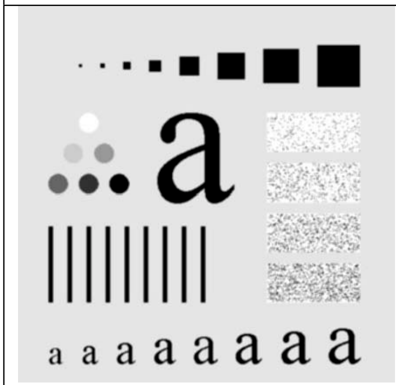
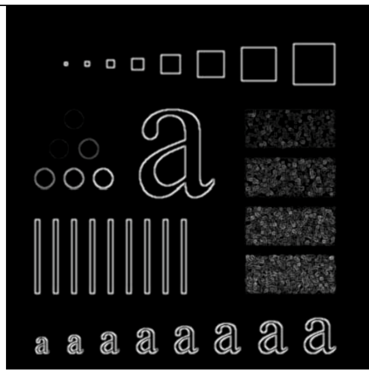
Figura Original	Figura Sobel
	

Fig. 12 Comparación Sobel.

1.2.5 Enfatizador de Altas Frecuencias

El realce de la nitidez con este método consiste en sustraer una versión de la imagen original sin bordes, por ejemplo, el resultado de un filtro pasa bajos, de un arreglo de la misma imagen multiplicada por un factor escalar K, dicha sustracción debe resultar en la imagen original con los bordes ligeramente acentuados.

Fig. 13 Enfatizador de Altas Frecuencias

$$f_{\mathrm{Re}\,sultado}(x,y) = k[f(x,y)] - f(x,y) * H(x,y)$$

0.1111	0.1111	0.1111
0.1111	0.1111	0.1111
0.1111	0.1111	0.1111

H - High Boost Filter

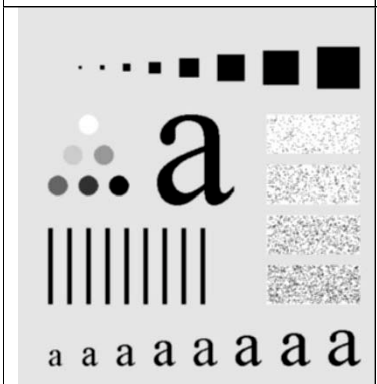
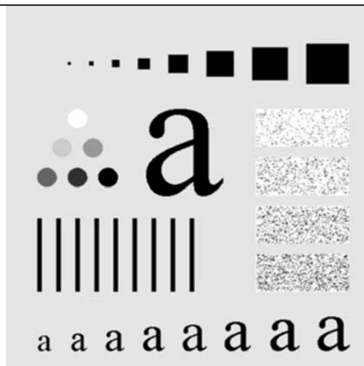
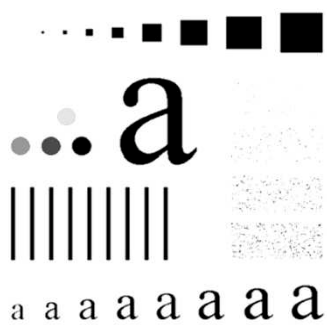
Figura Original	**Figura Enfatizador de Altas Frecuencias K=2**	**Figura Enfatizador de Altas Frecuencias K=3**
		

Fig. 14 Comparación Enfatizador de Altas Frecuencias

1.2.6 Laplaciano

El operador Laplaciano se define como un filtro diferencial de segundo orden, en general estos proporcionan una localización mejor del borde y los contornos detectados son curvas cerradas. Esto es muy importante en la segmentación de imagen. También, no hay respuesta a las áreas de variaciones lineales lisas en intensidad.

No obstante presenta una sensibilidad más grande frente al ruido, una ligera incapacidad para determinar la dirección de los bordes y puede generar bordes dobles.

El Laplaciano es omnidireccional, es decir, destacará los bordes en todas las direcciones, entonces producirá bordes más agudos que la mayoría de las otras técnicas.

El Laplaciano detecta los bordes, es decir, las altas frecuencias de la imagen, sin considerar la orientación, por lo que además de utilizarse para la detección de bordes sirve también para el filtrado paso alto de imágenes([7]).

[7] http://www.des.udc.es/~adriana/TercerCiclo/CursoImagen/curso/web/Filtrado_Espacial_Borde.html

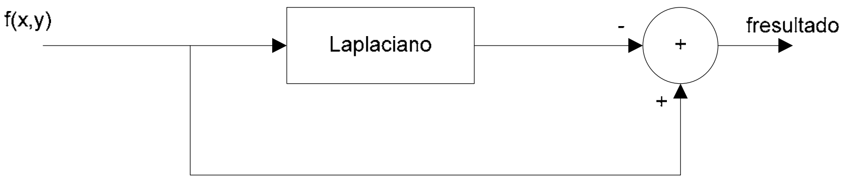


Fig. 15 Filtro Laplaciano.

$$f_{\text{Re}\,sultado}(x, y) = f(x, y) - f(x, y) * H(x, y)$$

$$\begin{matrix} 0.1667 & 0.6667 & 0.1667 \\ 0.6667 & -3.3333 & 0.6667 \\ 0.1667 & 0.6667 & 0.1667 \end{matrix}$$

H - Laplaciano

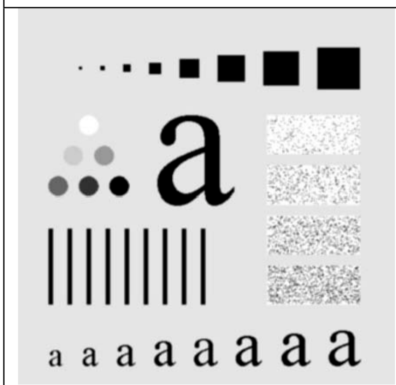
Figura Original	**Figura Laplaciano**
	

Fig. 16 Comparación Laplaciano.

CAPITULO 2

2. AJUSTE DE CURVA

Es una ecuación dada en un grafico, dependiendo del grado de correlación que mas se ajuste al conjunto de datos.

2.1 Ajuste Logarítmico

Se busca ajustar la información de la imagen hacia una curva logarítmica, el cual se logra aplicando la función descrita en la figura 17. Donde k es una constante que da en valor de la curva

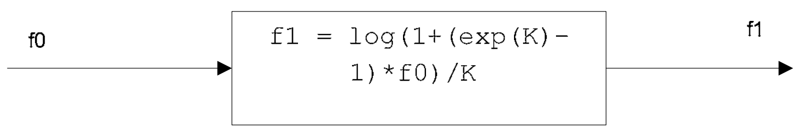


Fig. 17 Ajuste Logarítmico.

Figura Original	Figura Ajuste Log K=2	Figura Ajuste Log K=3
		

Fig. 18 Comparación Ajuste Logarítmico ([8]).

2.2 Ecualización de Histograma

La ecualización del histograma de una imagen es una transformación que pretende obtener para una imagen un histograma (*representación gráfica de una variable en forma de barras, donde el valor de cada barra es proporcional a la frecuencia de los valores representados*) con una distribución uniforme. Es decir, que exista el mismo número de pixeles para cada nivel de gris del histograma de una imagen monocroma.

[8] FOTO: http://www3.uji.es/~lopeza/pdidoc//E75/figuras/figura-3-10.html

El resultado de la ecualización maximiza el contraste de una imagen sin perder información de tipo estructural, esto es, conservando su información ([9]).

$$v(i,j)=\frac{(Fu(u(i,j))-Fu(a))}{1-Fu(a)}(L-1)+0.5$$

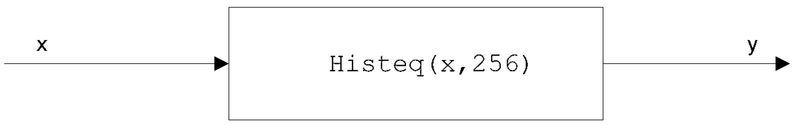


Fig. 19 Ecualización de Histograma.

Donde HISTEG es una función especial de MATLAB que hace la ecualización de histograma.

[9] http://es.wikipedia.org/wiki/Ecualizaci%C3%B3n_del_histograma

Figura Original	**Figura Ecualización de Histograma**
	

Fig. 20 Comparación Ecualización de Histograma.

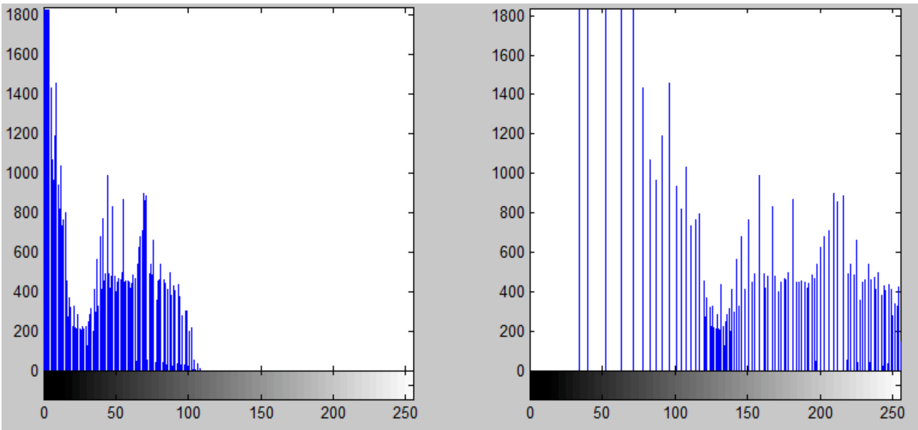


Fig. 21 Histogramas Resultantes.

Se observa en el histograma de la derecha como los valores de la imagen se distribuyen mejor en la escala de grises, mejorando el contraste.

CAPITULO 3

3. IMPLEMENTACION Y RESULTADOS EXPERIMENTALES

Para el desarrollo del proyecto implementaremos los filtros y ajustes provistos en las secciones III y IV en una interfaz grafica de usuario, de esta manera obtendremos una aplicación, realizada en MATLAB, y la que puede convertirse en una herramienta de gran utilidad en el diagnostico de traumas en el cuerpo humano.

Véase el manual de usuario en el ***Apéndice A*** y véase el código de la aplicación en el ***Apéndice B.***

La aplicación mejora imágenes radiográficas y podría ayudar a detectar fracturas y fisuras que son imperceptibles a simple vista.

Al cargar una imagen radiográfica en el programa esta es llevada a escala de grises, debido a que el tratamiento de este tipo de imágenes es más simple, y además de que se puede apreciar de mejor manera los perfiles de los huesos, que son el principal objetivo de nuestra propuesta.

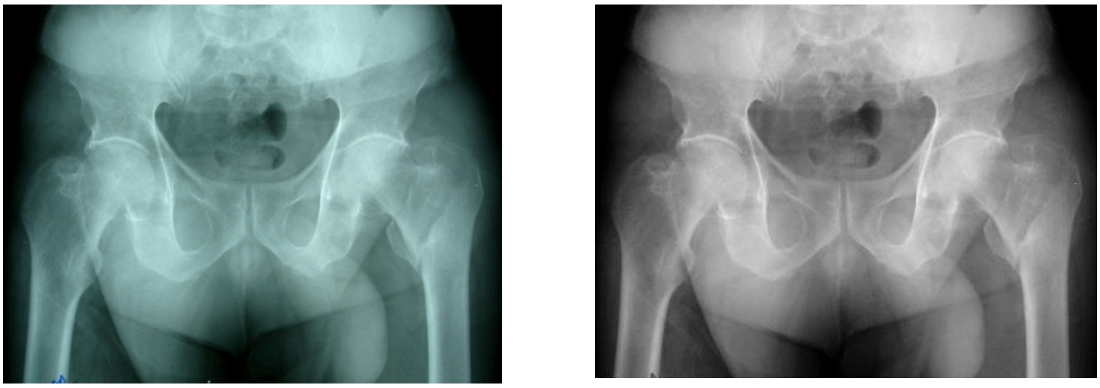


Fig. 22 Comparación radiografías a color y en escala de grises.

A continuación se presentan algunos resultados obtenidos al trabajar con la aplicación:

Original	**Filtrada 1era UnSharp**
	

Filtrada 2da UnSharp	Filtrada 3era UnSharp
	

Fig. 23 Ejemplo 1.

Original	Ajuste Logarítmico K = 2.08	Enfatizador de Altas Frecuencias A = 2.3
		

Fig. 24 Ejemplo 2.

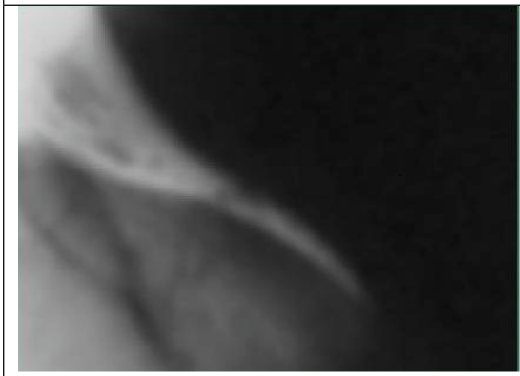
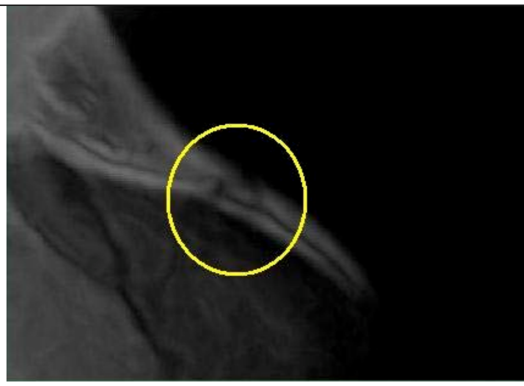
Original	Filtrada Por Sobel
	

Fig. 25 Ejemplo 3.

Original	Original con Zoom	Prewitt y Enfatizador de Altas Frecuencias con A = 2
		

Fig. 26 Ejemplo 4.

Según los resultados obtenidos en las figuras 23, 24, 25 y 26, se muestra que de ser necesario se podrían re-utilizar los resultados previos obtenidos, con el fin de alcanzar los resultados deseados, o bien se podría aceptar el primer filtrado, según lo requiera el usuario.

CONCLUSIONES

1 El programa desarrollado compila numerosas herramientas aprendidas en el curso dando una aplicación didáctica como la mejora de imágenes radiográficas.

2 El tratamiento de las imágenes con estas herramientas ayudaran o confirmar o descartar cualquier duda que el doctor pudiera tener acera sobre una determinada fisura o fractura, cabe destacar que los problemas comunes radican en meseta tibial de la rodilla, el escafoides de la muñeca y fractura imperceptibles en el cuello del fémur especialmente en ancianos.

3 El usuario debe familiarizarse con el programa ya que la aplicación de un determinado filtro no necesariamente sería suficiente para encontrar una mejora significativa de la imagen, todo depende de la imagen fuente

4 Con la utilización de este software para imágenes de alta resoluciones (ejemplo: 3264 x 2448), Matlab comienza a inhibirse y

deja de tener un funcionamiento esperado, los cálculos para esa cantidad de pixeles exigen de una cantidad grande de memoria

5 Los filtros utilizados son de tipo pasa altos, debido a que las altas frecuencias de una imagen corresponden a los filos bien definidos de las mismas, de hecho en el dominio del espacio es equivalente al rápido cambio del brillo de 2 o más pixeles adyacentes.

6 Para imágenes de pobre resolución, el filtrado por cualquiera de los tipos mostrados aquí, aumentan el ruido de la misma, provocando distorsión y carece de todo propósito el funcionamiento de este programa para ese tipo de imágenes.

7 Los ajustes de curva ya sea exponencial o logarítmica (como en este caso) llevan los pixeles a una función, la cual determinara su nueva intensidad según sea la grafica a la que se esté evaluando. El ajuste logarítmico aplicado es ideal para radiografías opacas o con poca luz porque ayuda a obtener mejores detalles ocultos, debido a los problemas de ojo humano al solo poder diferencia 80 escala de grises, mientras más alto sea el valor de k más brillo tendrá la imagen.

REFERENCIAS

[1] Fotografías. Dr. Miguel Mite Vivar, Medico Traumatólogo, Clínica San Francisco, Ciudad de Guayaquil.

[2] Erik Valdemar Cuevas Jimenez ,"Visión por Computador utilizando MatLAB y el Toolbox de Procesamiento Digital de Imágenes", 2007

[3] Hector Zarate, "Aplicación de las Matrices en el Procesamiento de Imágenes", Universidad Nacional Autónoma de México

[4] Alexander Vera Tasamá, Andrés F. Ramírez Sánchez, Daniel Moreno Martínez, Milton J. Alzate Silva, "Aplicación Didáctica para el Procesamiento de Imágenes Digitales Usando Interfaz Gráfica de Usuario en MATLAB",

[5] Departamento de Ingeniería electrónica, Telecomunicación y Automática. "Detección de bordes en una imagen", Universidad de Jaen, 2005

[6] Rafael C. Gonzalez, Richard E. Woods, Steven L. Eddins, "Digital Image Processing using Matlab", Prentice Hall, 2004

[7] Vinay K. Ingle, John G. Proakis, Northeastern University, “Digital Signal Processing using Matlab V.4”, PWS, 1997

[8] Steven W. Smith, “A Practical Guide for Enginners and Scientist” , Newnes 2003

[9] Saeed V. Vaseghi, “Advance Digital Sinal Procesing and Noise Reduction”, second edition, John Wiley, 2000

[10] Ing. Sergio Casillas de la Torre, Tutorial de Matlab

[11]http://www.fas.hcmut.edu.vn/webhn10/Baocao/PDF/TDLinh-MIRegistration.pdf

[12]http://amath.colorado.edu/courses/4720/2000Spr/Labs/Worksheets/Matlab_tutorial/matlabimpr.html

[13]http://www.cogs.susx.ac.uk/users/davidy/compvis/matlab_demos/intro_demo.html

[14]http://www.eng.auburn.edu/~sjreeves/Classes/IP/IP.html

[15]http://www.ece.umn.edu/users/chand132/

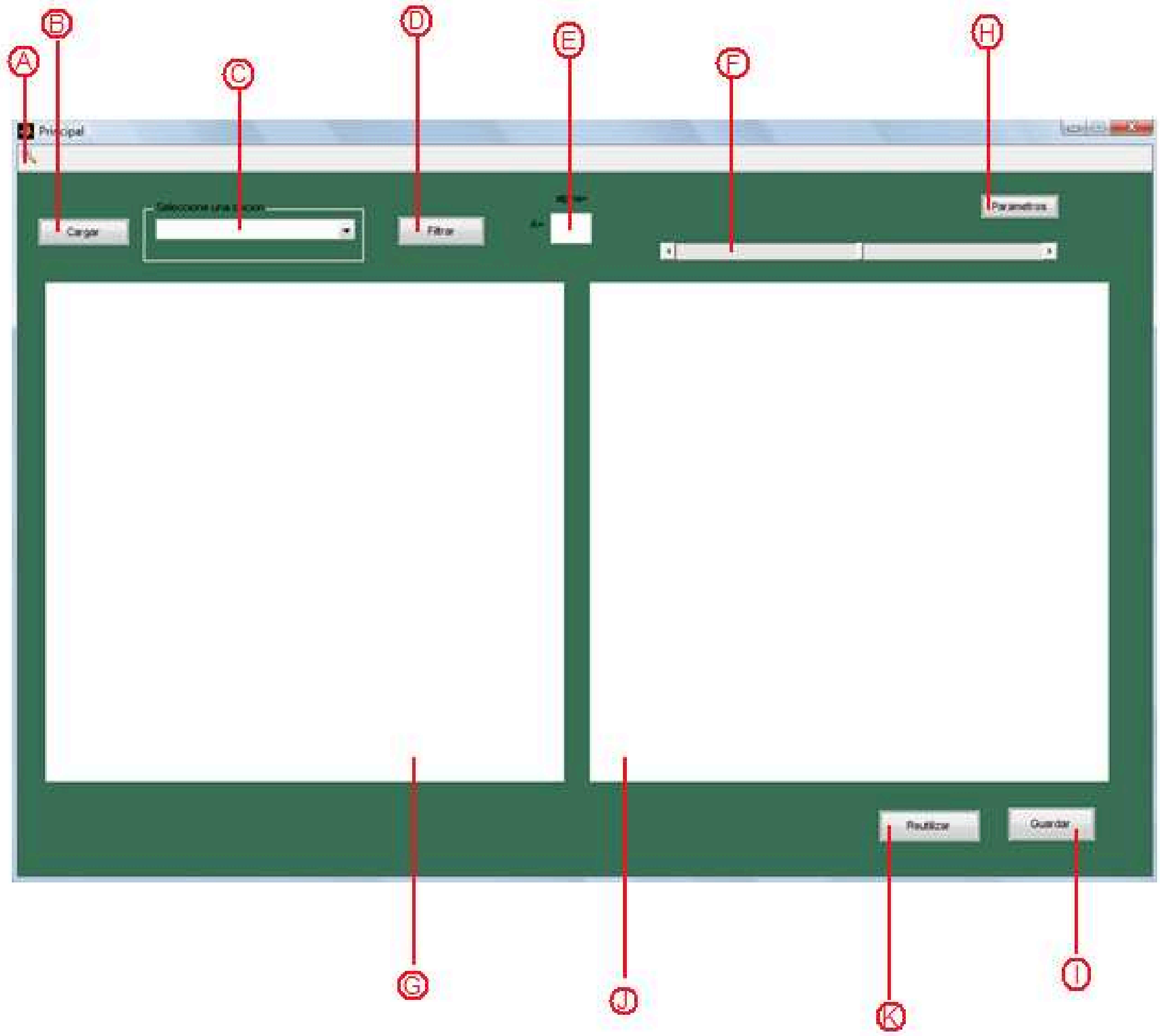
[16]http://homepages.inf.ed.ac.uk/rbf/HIPR2/log.htm

ANEXOS

ANEXO A

MANUAL DE USUARIO

Pagina Principal:



A: Herramienta para hacer zoom a las imágenes

B: Botón para cargar imagen

C: Menú desplegable para seleccionar una de las siguientes opciones:

1. Unsharp
2. Emboss
3. Realce 1
4. Realce 2
5. Roberts
6. Prewitt
7. Sobel
8. Enfatizador de Altas Frecuencias
9. Laplaciano
10. Ajuste Logarítmico
11. Ecualización de Histograma

D: Botón para filtrar o ecualizar una imagen luego de ser cargada

E: Caja de texto para ingresar valores correspondientes al valor de A para Enfatizador de Altas Frecuencias o el valor de alpha para filtro Laplaciano

F: Barra de desplazamiento utilizada en el ajuste logarítmico de imágenes

G: Muestra la imagen a ser procesada

H: Llama a un submenú para cargar los valores máximos y mínimos de la barra de desplazamiento

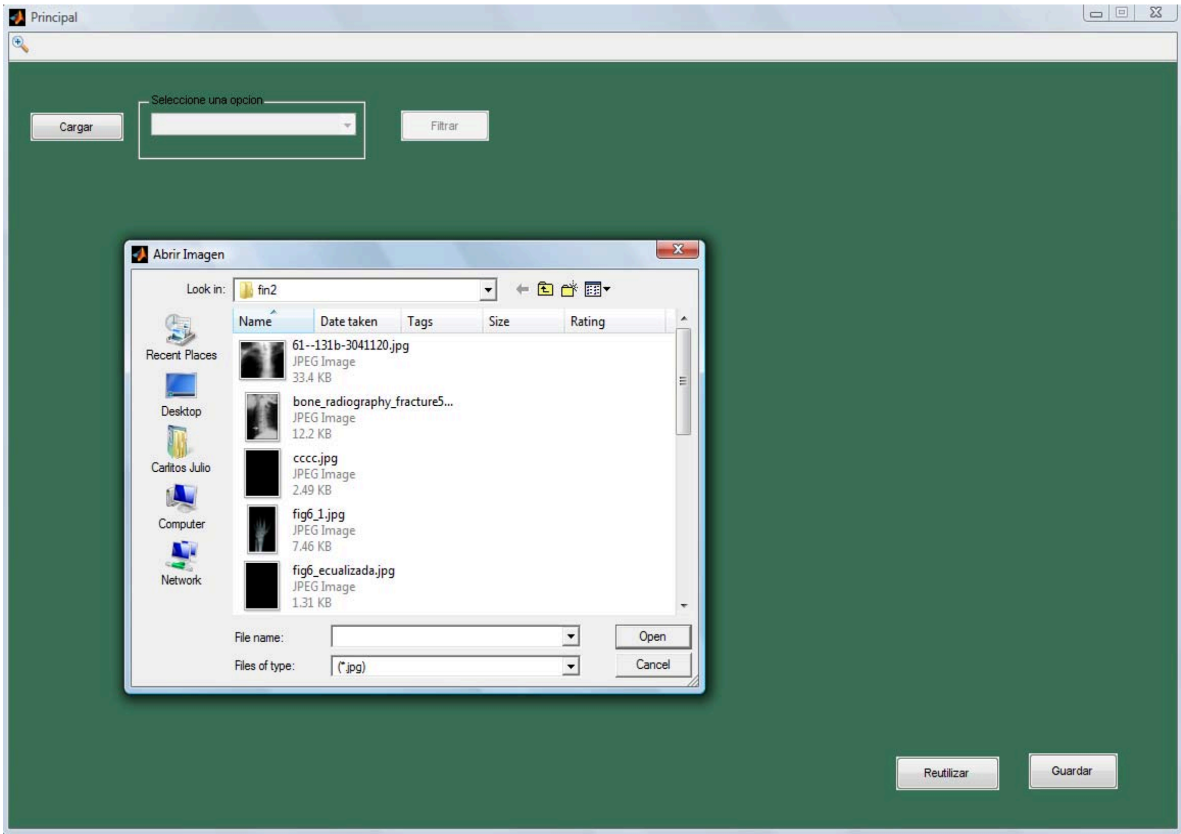
I: Guarda la imagen procesada

J: Muestra la imagen procesada

K: Botón para reprocesar una imagen ya utilizada, la imagen es cargada en G y se limpia la imagen mostrada J

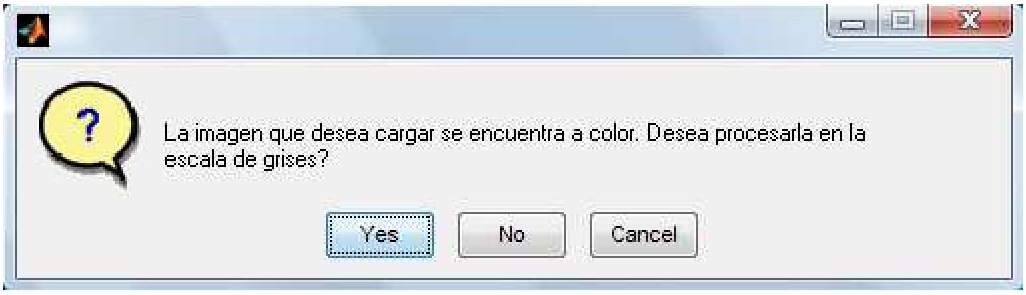
------------------------------CARGANDO UNA IMAGEN---------------------------------

1. Para cargar una imagen hacer click en <u>cargar</u>



2. Luego seleccione la imagen a ser procesada en formato *.jpg
3. Presionar <u>Open</u>

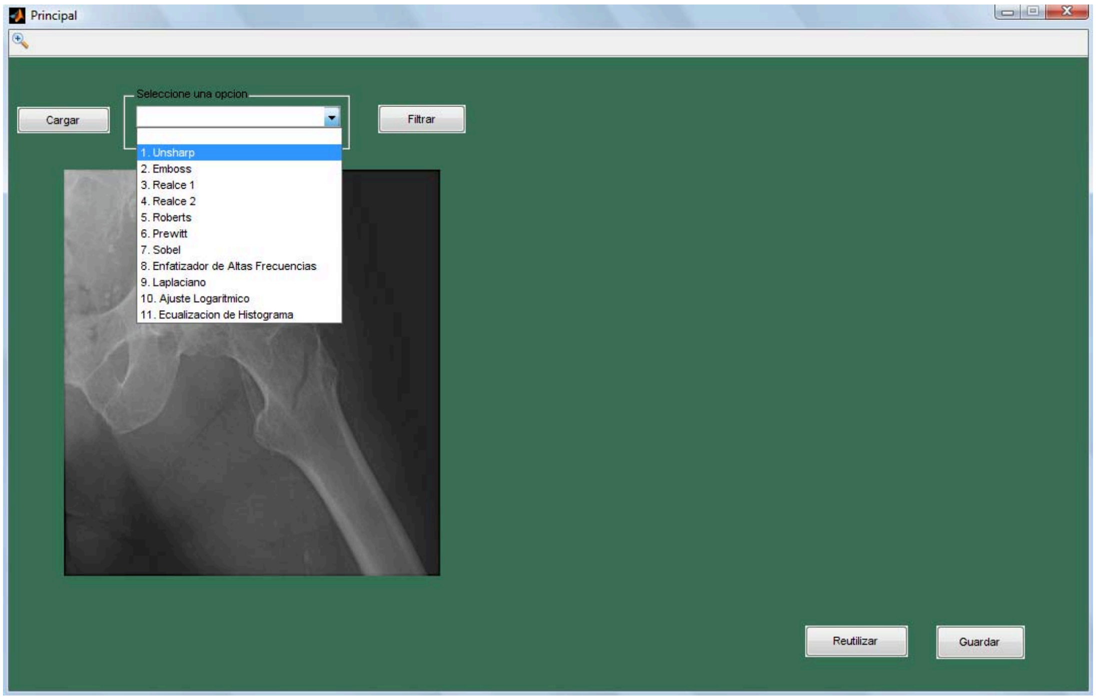
En caso de ser una imagen a color se mostrara el siguiente dialogo



El programa trabaja solo con imágenes en blanco y negro, no se cargara ninguna imagen en caso de seleccionar <u>No</u> o <u>Cancel</u>

-------------------SELECCIÓN DE OPCIONES (Mejorando Detalles) -----------------

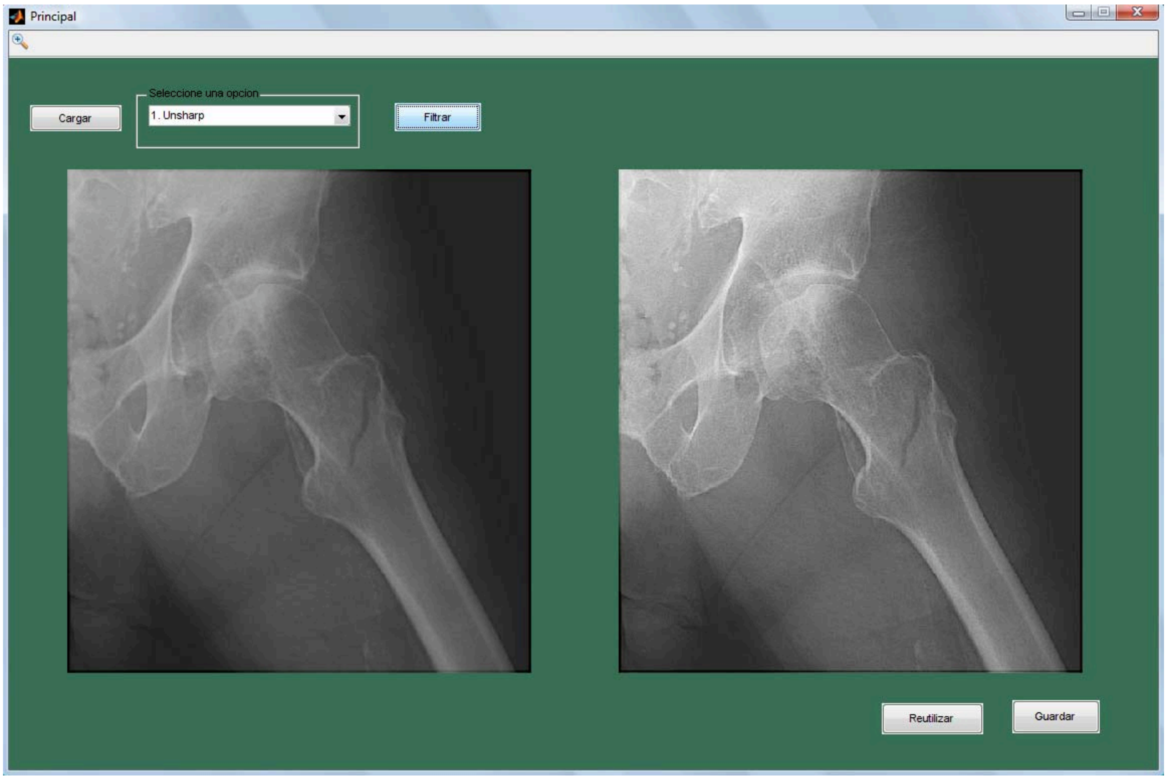
UNSHARP, EMBOSS, REALCE 1, REALCE 2, ROBERTS, PREWITT, SOBEL



1. Cargar imagen

2. Seleccionar una opción entre: Unsharp, Emboss, Realce 1, Realce 2, Roberts, Prewitt, Sobel

3. Presionar Filtrar



Principal
Cargar
Seleccione una opcion
1. Unsharp
Filtrar
Reutilizar
Guardar

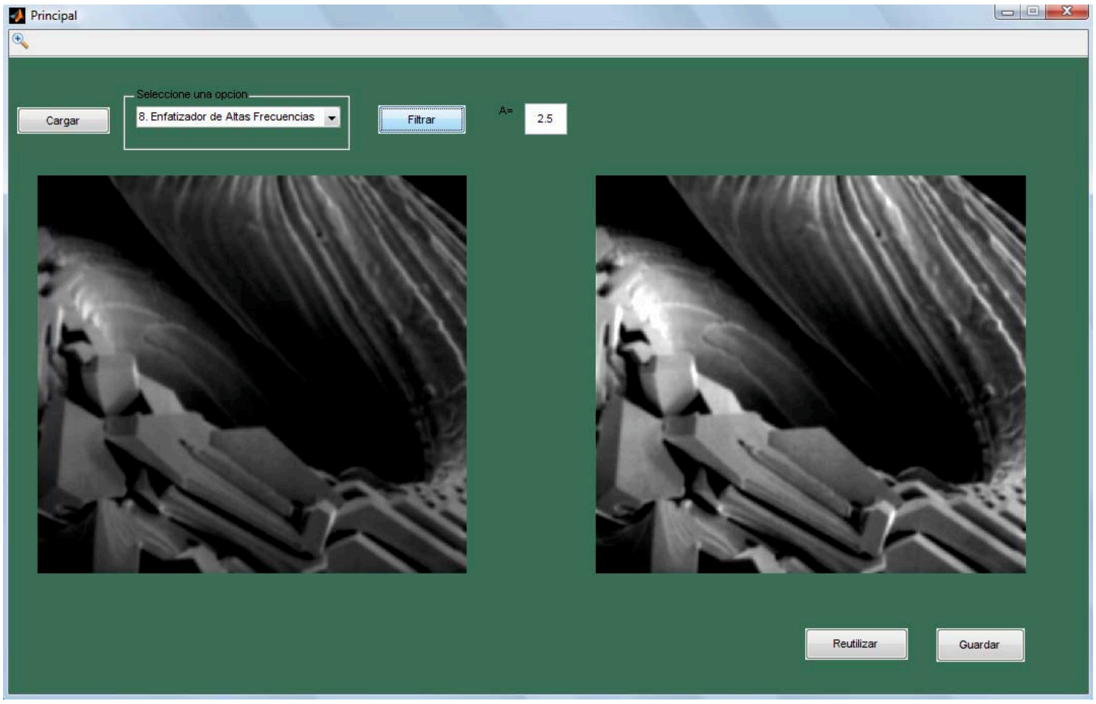
ENFATIZADOR DE ALTAS FRECUENCIAS

1. Cargar Imagen

2. Seleccionar “Enfatizador de Altas Frecuencias”

3. Ingresar el valor de A. (Entre más alto, más claro; por default se encuentra en 2)

4. Presionar Filtrar



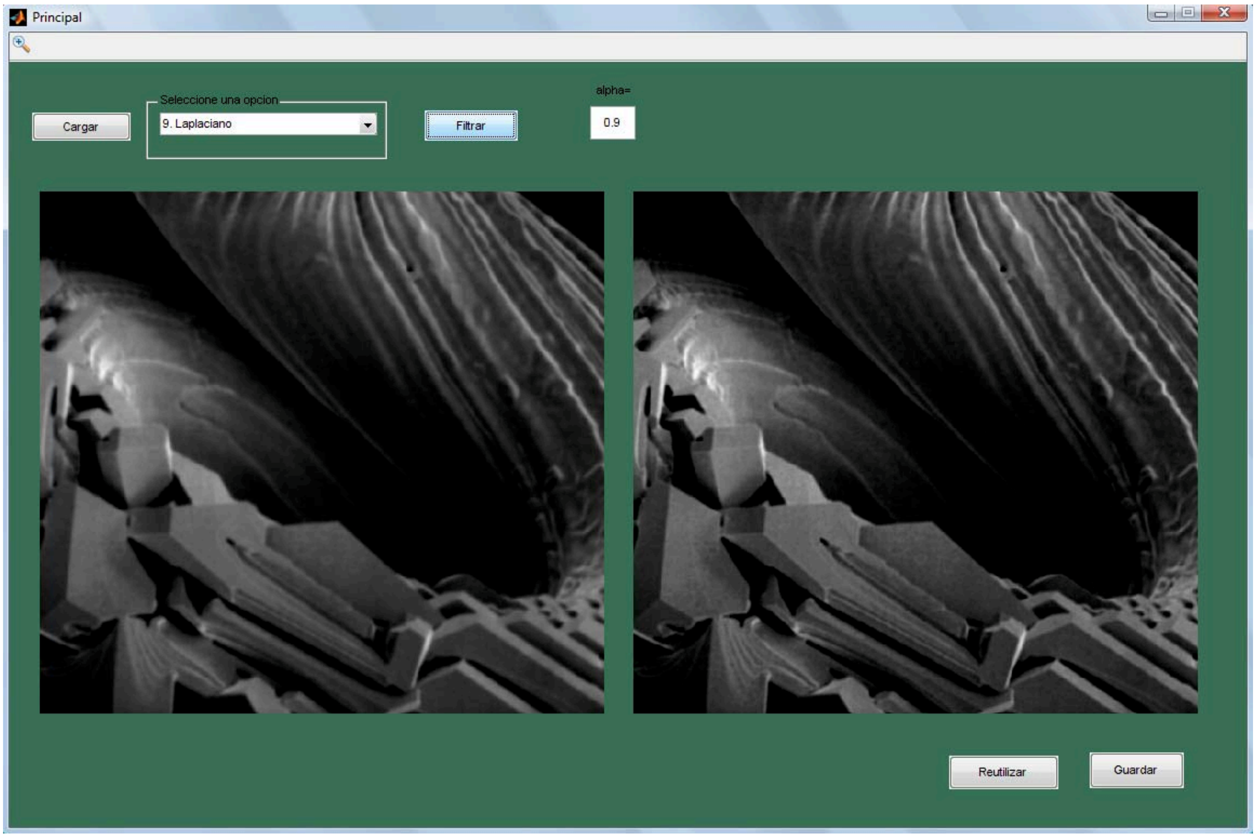
<u>LAPLACIANO</u>

1. Cargar Imagen

2. Seleccionar “Laplaciano”

3. Ingresar el valor de <u>alpha</u>. (Entre más alto, más claro; por default se encuentra en 0.2) ***Advertencia:*** *alpha es un que se encuentra entre 0 y 1*

4. Presionar <u>Filtrar</u>



AJUSTE LOGARITMICO

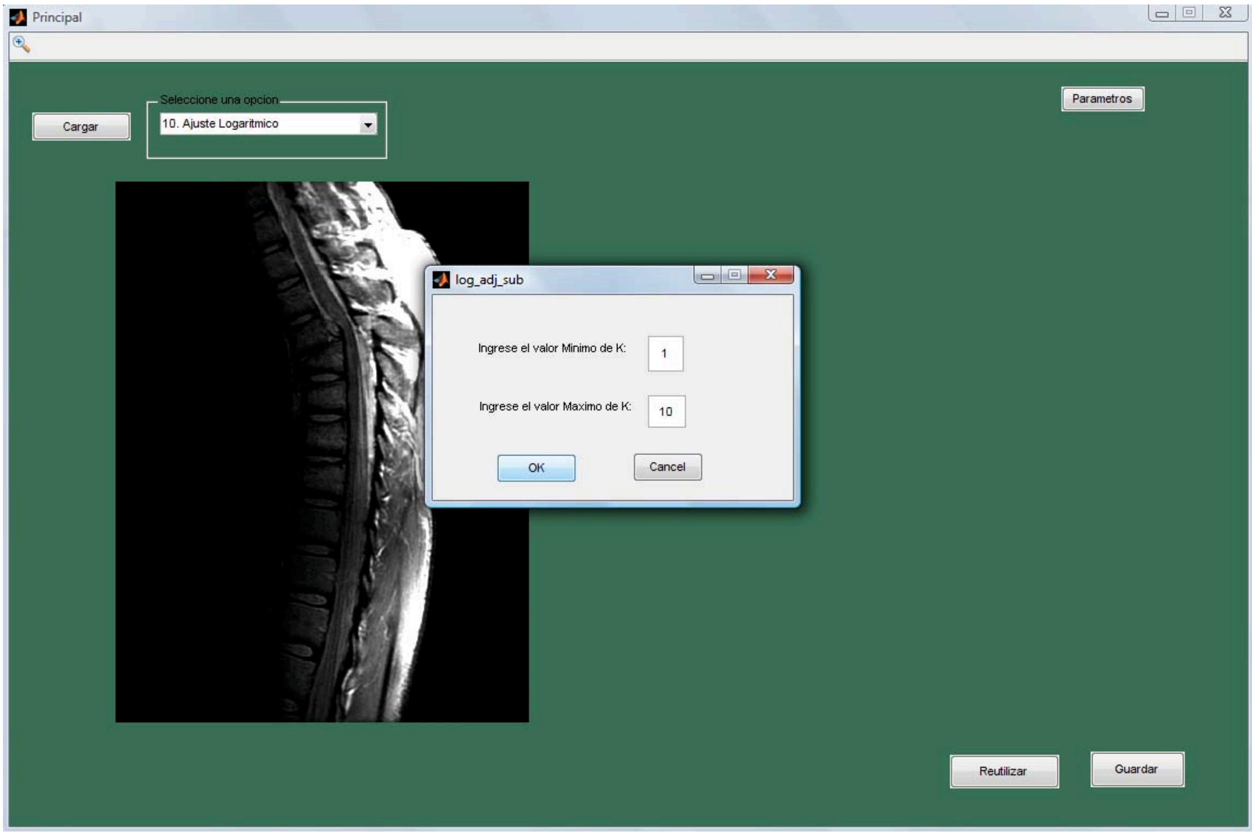
Recomendado para radiografías e imágenes oscuras.

Para ajustar la intensidad de los pixeles logarítmicamente debe hacer lo siguiente:

1. Cargar imagen

2. Seleccionar "Ajuste Logarítmico"

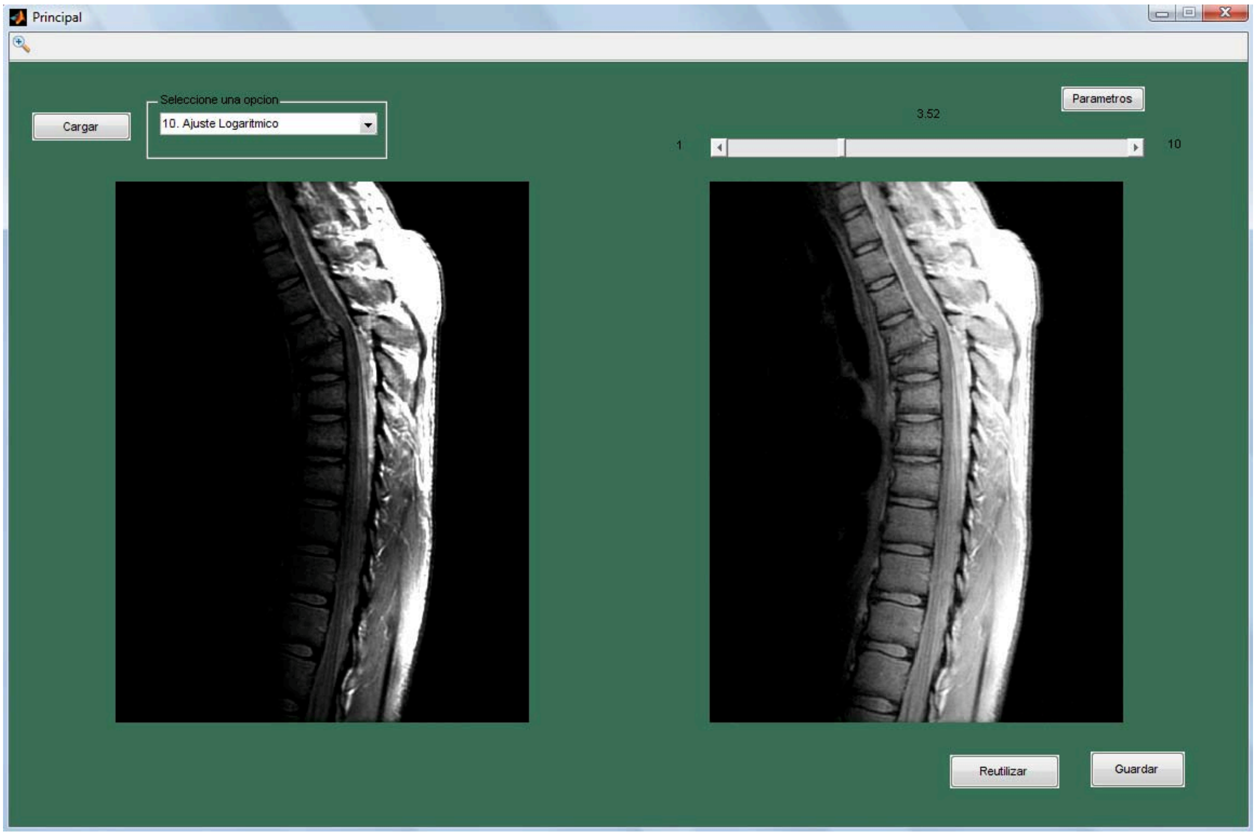
3. Presionar Parámetros



Parámetros llama a un submenú para cargar los valores máximos y mínimos para el ajuste, Se recomienda dejar en los valores por defecto.

4. Presionar OK

5. Mover la barra de desplazamiento hasta obtener la imagen deseada



A los extremos de la barra se encuentran los valores máximos y mínimos cargados, en medio se muestra el valor de K exacto para la imagen mostrada

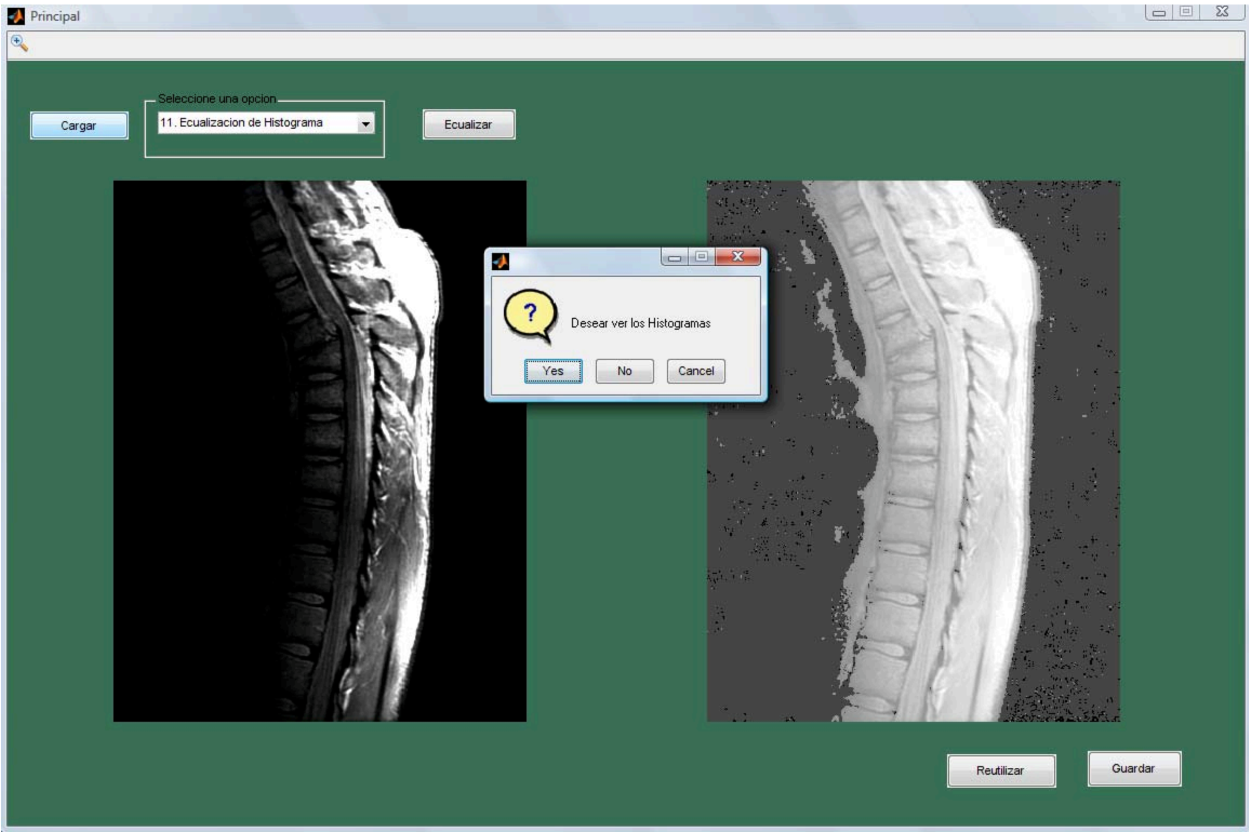
ECUALIZACION POR HISTOGRAMA

Recomendado para imágenes muy opacas

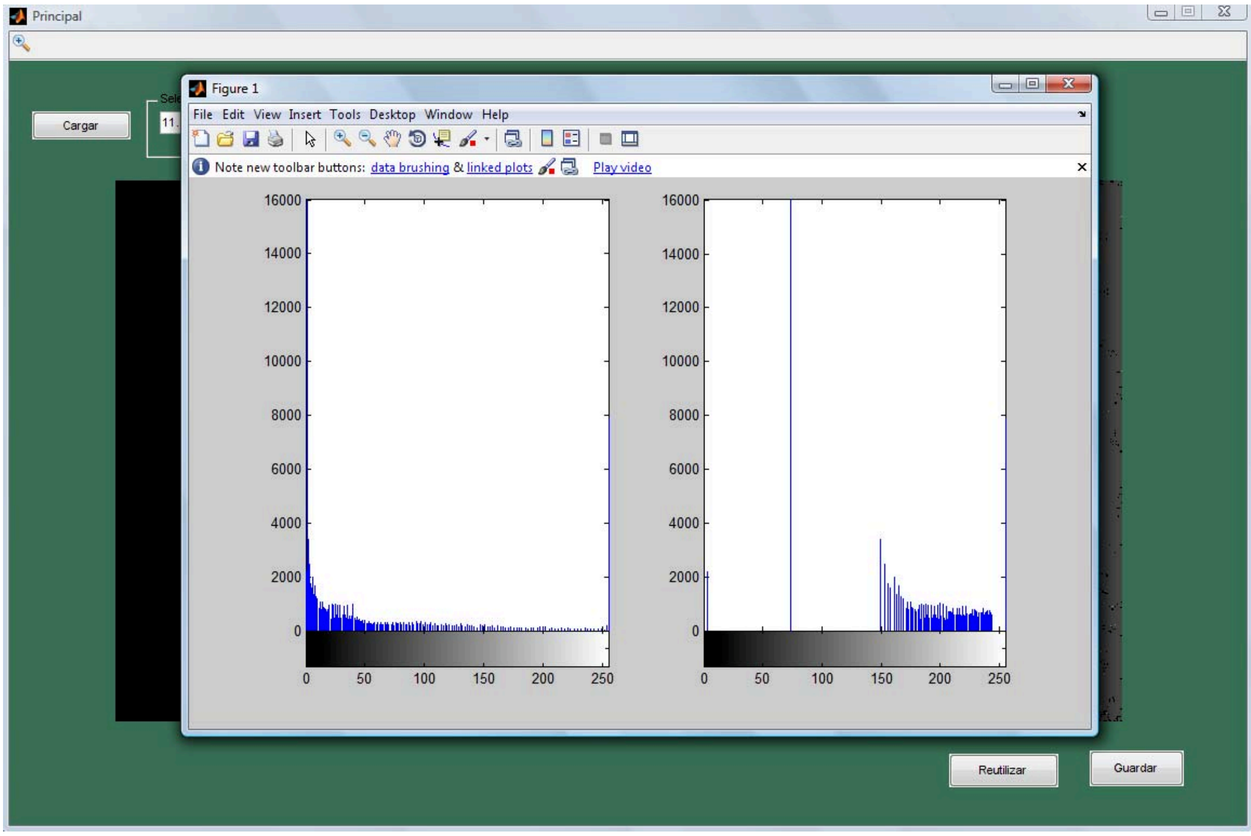
1. Cargar Imagen

2. Seleccionar "Ecualización por Histograma"

3. Presionar Ecualizar



Si selecciona Yes, observara el histograma de las imágenes

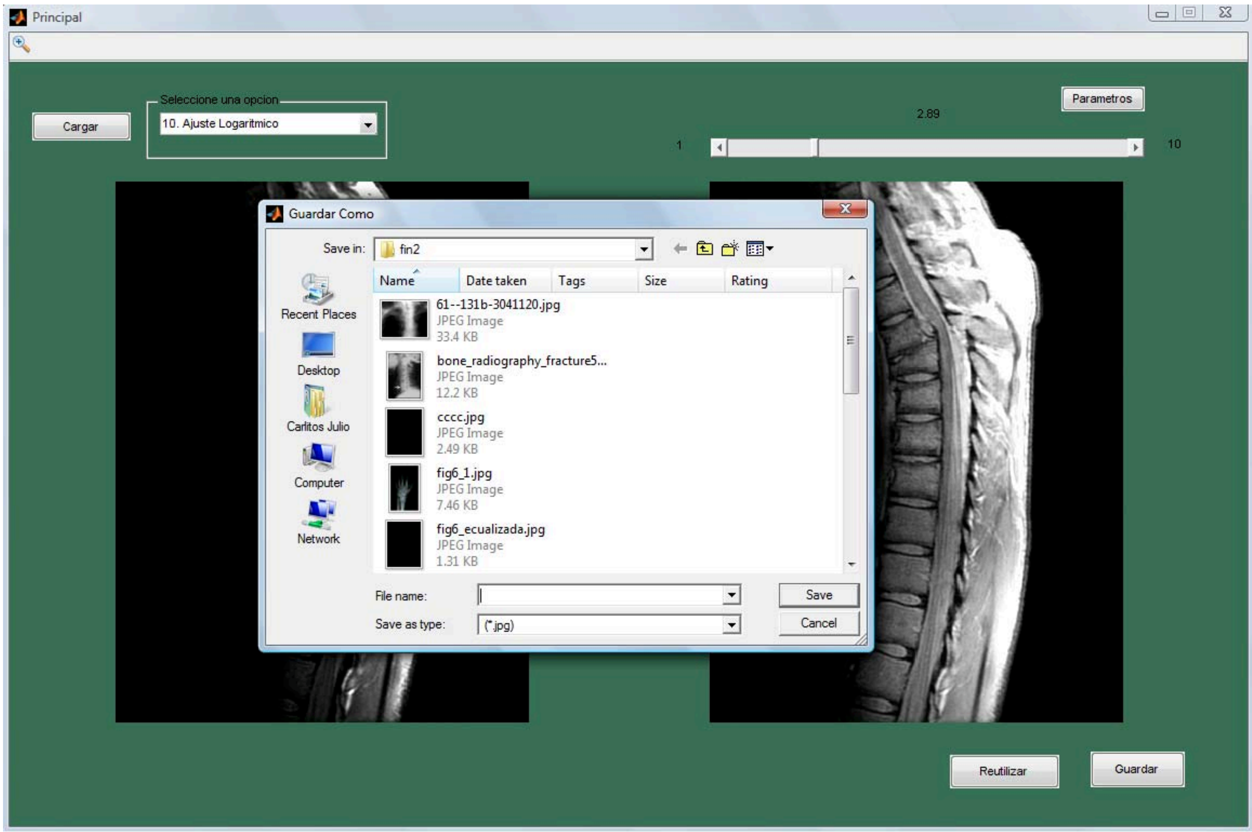


------------------------------PARA GUARDAR UNA IMAGEN------------------------

----

1. Una vez procesada la imagen presionar <u>Guardar</u>

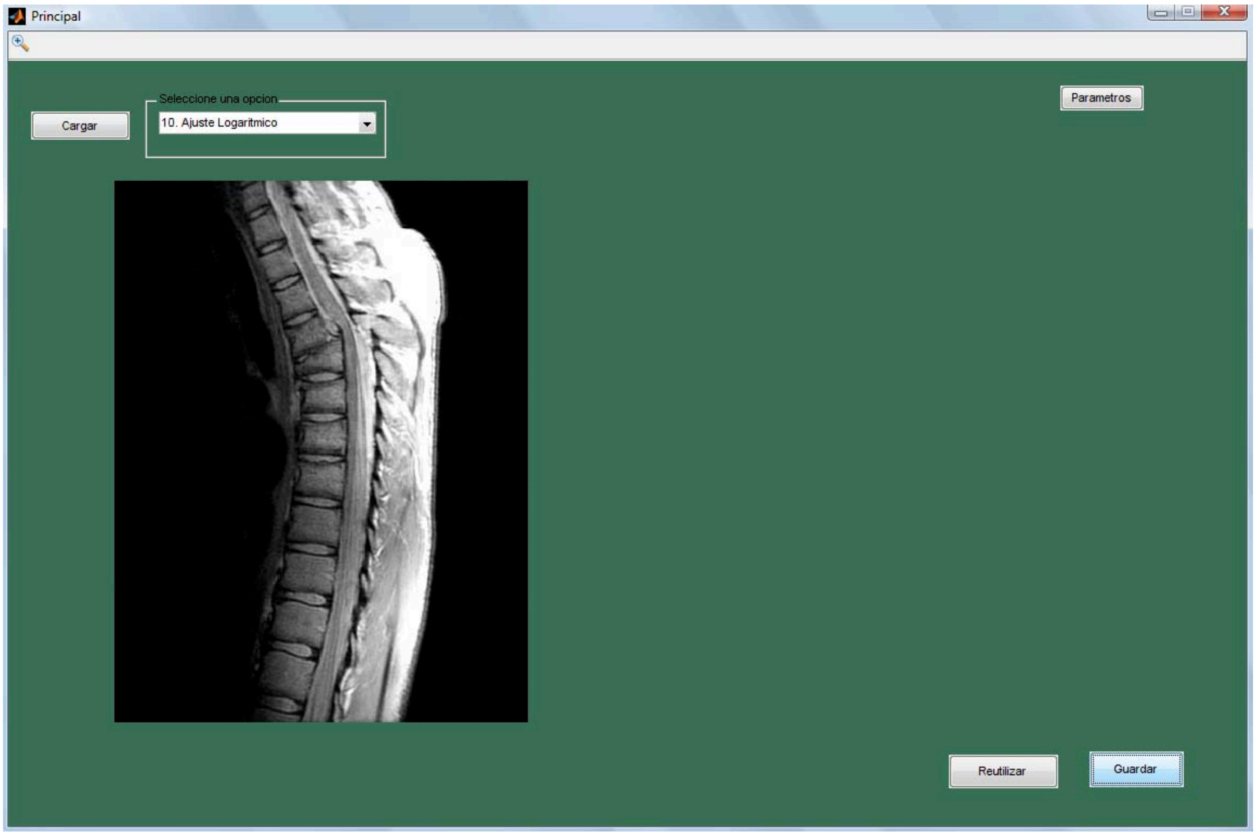
2. Escribir el nombre del archivo y presionar <u>Save</u>



---------------------------------REUTILIZAR IMAGEN-------------------------------------

--

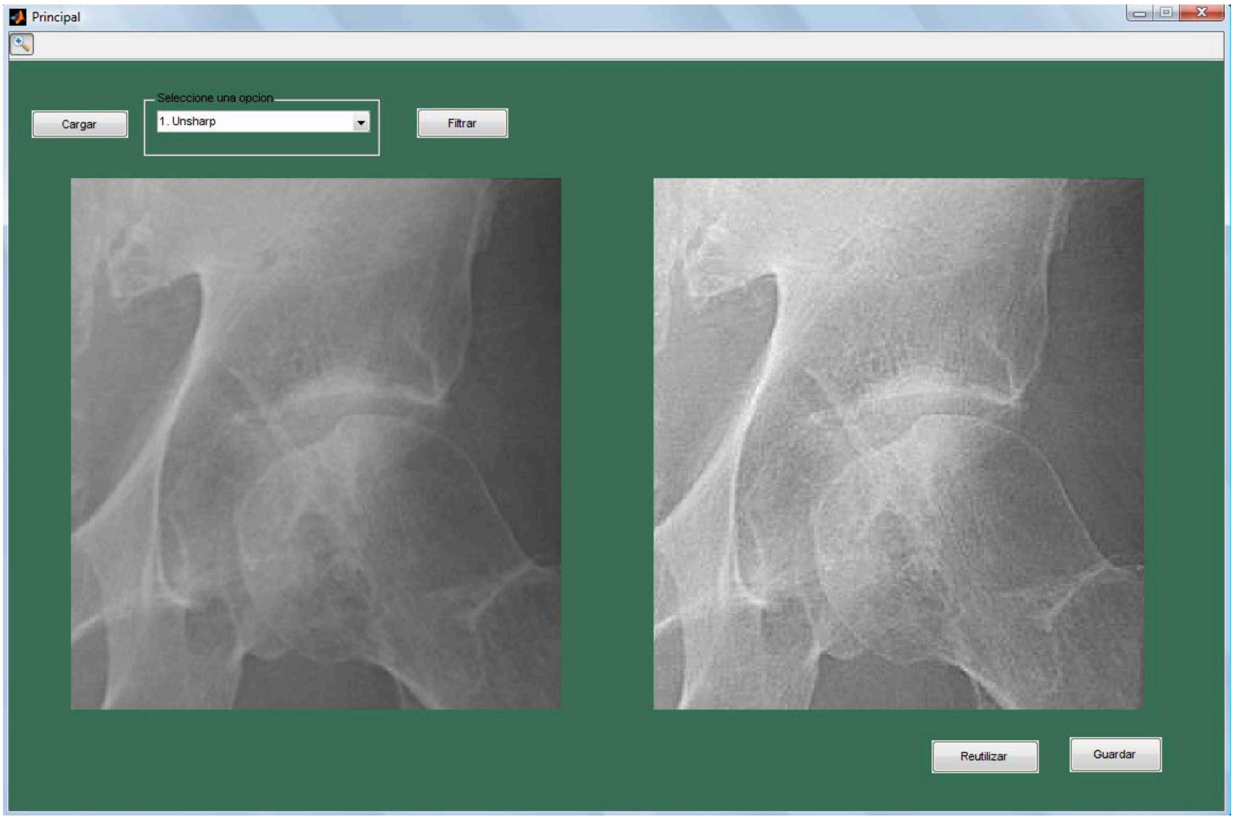
Se utiliza si desea aplicar más de un cambio a una imagen por ejemplo Filtrar y después ecualizarla. Esto evita que el usuario previamente guarde la imagen para proceder hacer otro cambio sobre la imagen ya procesada.



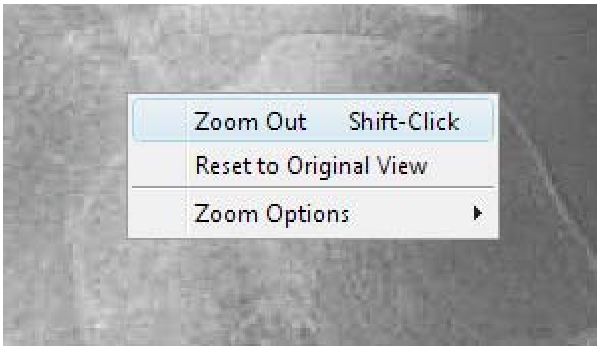
--------------------------------ZOOM DE UNA IMAGEN--------------------------------

----

Presionar  para agrandar una imagen



Para volver a la imagen original o hacer un zoom out, presionar click derecho y una de las opciones



ANEXO B

CODIGO DE MATLAB

```
function varargout = Principal(varargin)
% PRINCIPAL M-file for Principal.fig
%      PRINCIPAL, by itself, creates a new PRINCIPAL or raises the existing
%      singleton*.
%
%      H = PRINCIPAL returns the handle to a new PRINCIPAL or the handle to
%      the existing singleton*.
%
%      PRINCIPAL('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in PRINCIPAL.M with the given input arguments.
%
%      PRINCIPAL('Property','Value',...) creates a new PRINCIPAL or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before Principal_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to Principal_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools file.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Principal

% Last Modified by GUIDE v2.5 21-Feb-2009 14:29:28

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Principal_OpeningFcn, ...
                   'gui_OutputFcn',  @Principal_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
% End initialization code - DO NOT EDIT


% --- Executes just before Principal is made visible.
function Principal_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Principal (see VARARGIN)
```

```
% Choose default command line output for Principal
%Valores Iniciales al cargar
handles.output = hObject;
clc
axes(handles.graf1)
cla reset
axis off
axes(handles.graf2)
axis off
cla reset
axis off
set(handles.Filtrar,'Enable','Off')
set(handles.tfiltro,'Enable','Off')
set(handles.edit1,'Visible','Off')
set(handles.text1,'Visible','Off')
set(handles.textalpha,'Visible','Off')
set(handles.text5,'Visible','Off')
set(handles.text6,'Visible','Off')
set(handles.text7,'Visible','Off')
set(handles.text5,'String',' ')
set(handles.text6,'String',' ')
set(handles.text7,'String',' ')
set(handles.slider2,'Visible','Off')
set(handles.parametros,'Visible','Off')
handles.im2=0;
handles.haydata=0;
handles.d=0;
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Principal wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = Principal_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in Cargar.
function Cargar_Callback(hObject, eventdata, handles)
% hObject    handle to Cargar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Funciones del boton 'Cargar' para procesar una imagen
[FileName Path]=uigetfile({'*.jpg'},'Abrir Imagen');
if isequal(FileName,0)
   return
else
a=imread(strcat(Path,FileName));
tam = size(a);
```

```
end

%Verifico si la imagen es a color o en blanco y negro
if length(tam)==3
   rpta=questdlg('La imagen que desea cargar se encuentra a color. Desea procesarla en la escala de
grises?');

   switch rpta

   case 'Yes'
      a=rgb2gray(a);
      handles.imload=1    %bandera para determinar si existe imagen cargada

   case 'No'
      handles.imload=0

   case 'Cancel'
      handles.imload=0
   end

else
   handles.imload=1;
end

if handles.imload==1

   axes(handles.graf1);
   axis off;
   imshow(a);
   handles.im=a;
   set(handles.Filtrar,'Enable','On')

   axes(handles.graf2)
   axis off
   cla reset
   axis off

   set(handles.tfiltro,'Enable','On')
   set(handles.text5,'String',' ')
   set(handles.text6,'String',' ')
   set(handles.text7,'String',' ')
   set(handles.slider2,'Visible','Off')
end



guidata(hObject,handles);


% --- Executes on button press in Filtrar.
function Filtrar_Callback(hObject, eventdata, handles)
% hObject    handle to Filtrar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%Accion del boton Filtrar dependiende del valor seleccionado en el
%el menu deplegable
```

```
h=get(handles.tfiltro,'Value'); %Lectura del valor del menu desplegable

switch h

   case 1
     errordlg('Por favor seleccione un tipo de filtro');

   case 2      % Unsharp
      p=handles.im;
      f=mat2gray(p);
      H=fspecial('unsharp');
      y=imfilter(f,H,'replicate');
      final=f+y;
      final=mat2gray(final,[0,255]);
      axes(handles.graf2);
      axis off;
      imshow(final,[]);
      handles.im2=final;
      handles.op67=0;% bandera para determinar si la imagen esta uint8 o double
      guidata(hObject, handles);

   case 3      %Emboss
      p=handles.im;
      f=mat2gray(p);
      H=[-2 -1 0; -1 1 1; 0 1 2];
      y=imfilter(f,H,'replicate');
      final=f+y;
      final=mat2gray(final,[0,255]);
      axes(handles.graf2);
      axis off;
      imshow(final,[]);
      handles.im2=final;
      handles.op67=0;
      guidata(hObject,handles);

   case 4      %Realce 1
      p=handles.im;
      f=mat2gray(p);
      H=[-1 -1 -1; -1 9 -1; -1 -1 -1];
      y=imfilter(f,H,'replicate');
      final=f+y;
      final=mat2gray(final,[0,255]);
      axes(handles.graf2);
      axis off;
      imshow(final,[]);
      handles.im2=final;
      handles.op67=0;
      guidata(hObject,handles);

   case 5      %Realce 2
      p=handles.im;
      f=mat2gray(p);
      H=(1/7)*[-1 -2 -1; -2 19 -2; -1 -2 -1];
      y=imfilter(f,H,'replicate');
      final=f+y;
      final=mat2gray(final,[0,255]);
      axes(handles.graf2);
      axis off;
      imshow(final,[]);
```

```
    handles.im2=final;
    handles.op67=0;
    guidata(hObject,handles);

case 6      %Roberts
    p=handles.im;
    f=mat2gray(p);

    Hx=[0 0 -1; 0 1 0; 0 0 0];
    yx=imfilter(f,Hx,'replicate');
    Hy=[-1 0 0; 0 1 0; 0 0 0];
    yy=imfilter(f,Hy,'replicate');
    y=abs(yx)+abs(yy)
    final=f+y;
    final=mat2gray(final,[0,255]);
    axes(handles.graf2);
    axis off;
    imshow(final,[]);
    handles.im2=final
    handles.op67=0;
    guidata(hObject,handles);

case 7      %Prewitt
    p=handles.im;
    f=mat2gray(p);

    Hx=fspecial('prewitt');
    Hy=Hx';
    yx=imfilter(f,Hx,'replicate');
    yy=imfilter(f,Hy,'replicate');
    y=abs(yx)+abs(yy);
    final=f+y;
    final=mat2gray(final,[0,255]);
    axes(handles.graf2);
    axis off;
    imshow(final,[]);
    handles.im2=final
    handles.op67=0;
    guidata(hObject,handles);

case 8      %Sobel
    p=handles.im;
    f=mat2gray(p);

    Hx=fspecial('sobel');
    Hy=Hx'
    yx=imfilter(f,Hx,'replicate');
    yy=imfilter(f,Hy,'replicate');
    y=abs(yx)+ abs(yy);
    final=f+y;
    final=mat2gray(final,[0 255]);
    axes(handles.graf2);
    axis off;
    imshow(final,[]);
    handles.im2=final
    handles.op67=0;
    guidata(hObject,handles);

case 9      %Enfatizador de Altas Frecuecias
```

```
        x=handles.im;
        x = mat2gray(double(x),[0,255]);
        xlp=imfilter(x,fspecial('average',[3 3]));
        A=str2num(get(handles.edit1,'String'));
           if A>=1
               y = A*x-xlp;
               y = uint8(y*(255));
               axes(handles.graf2);
               axis off;
               imshow(y);
           else
               errordlg('Imposible Filtrar: Seleccione un valor de A >= 1');
           end
          handles.im2=y;
          handles.op67=1;
          guidata(hObject,handles);

    case 10     %Laplaciano

        p=handles.im;
        f=mat2gray(p);

        b=str2num(get(handles.edit1,'String'));

        if (b>=0)&&(b<=1)

           H=fspecial('laplacian',b);
           y=imfilter(f,H,'replicate');
           final=f-y;
           final=mat2gray(final,[0,255]);
           axes(handles.graf2);
           axis off;
           imshow(final,[]);
           handles.im2=final;
           handles.op67=0;
        else
           errordlg('Imposible Filtrar: Seleccione un valor de alpha valido entre 0 y 1');
        end
        guidata(hObject,handles);

    case 12        %Ecualizacion por Histograma
        x=handles.im;
        y=histeq(x,256);
        axes(handles.graf2);
        axis off;
        imshow(y,[]);
        rp=questdlg('Desear ver los Histogramas')
        switch rp
           case 'Yes'
               figure, subplot (1,2,1),imhist(x),subplot (1,2,2),imhist(y);
        end
        handles.im2=y;
        handles.op67=1;

        guidata(hObject,handles);
end

guidata(hObject,handles);
```

```
% --- Executes on selection change in tfiltro.
function tfiltro_Callback(hObject, eventdata, handles)
% hObject    handle to tfiltro (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns tfiltro contents as cell array
%        contents{get(hObject,'Value')} returns selected item from tfiltro

%Habilitacion/Desabilitacion de los cuadros segun la funcion Seleccionada
%en la lista desplegable
if(get(hObject,'Value'))==0
   set(handles.Filtrar,'Visible','On')
   set(handles.edit1,'Visible','Off')
   set(handles.text1,'Visible','Off')
   set(handles.parametros,'Visible','Off')
   set(handles.textalpha,'Visible','Off')
   set(handles.text5,'Visible','Off')
   set(handles.text6,'Visible','Off')
   set(handles.text7,'Visible','Off')
   set(handles.slider2,'Visible','Off')
   set(handles.Filtrar,'String','Filtrar')
end
if(get(hObject,'Value'))==1
    set(handles.Filtrar,'Visible','On')
    set(handles.edit1,'Visible','Off')
    set(handles.text1,'Visible','Off')
    set(handles.parametros,'Visible','Off')
    set(handles.textalpha,'Visible','Off')
    set(handles.text5,'Visible','Off')
    set(handles.text6,'Visible','Off')
    set(handles.text7,'Visible','Off')
    set(handles.slider2,'Visible','Off')
    set(handles.Filtrar,'String','Filtrar')
end
if(get(hObject,'Value'))==2
   set(handles.Filtrar,'Visible','On')
   set(handles.edit1,'Visible','Off')
   set(handles.text1,'Visible','Off')
   set(handles.parametros,'Visible','Off')
   set(handles.textalpha,'Visible','Off')
   set(handles.text5,'Visible','Off')
   set(handles.text6,'Visible','Off')
   set(handles.text7,'Visible','Off')
   set(handles.slider2,'Visible','Off')
   set(handles.Filtrar,'String','Filtrar')
end
if(get(hObject,'Value'))==3
   set(handles.Filtrar,'Visible','On')
   set(handles.edit1,'Visible','Off')
   set(handles.text1,'Visible','Off')
   set(handles.parametros,'Visible','Off')
   set(handles.textalpha,'Visible','Off')
   set(handles.text5,'Visible','Off')
   set(handles.text6,'Visible','Off')
   set(handles.text7,'Visible','Off')
   set(handles.slider2,'Visible','Off')
   set(handles.Filtrar,'String','Filtrar')
```

```
end
if(get(hObject,'Value'))==4
   set(handles.Filtrar,'Visible','On')
   set(handles.edit1,'Visible','Off')
   set(handles.text1,'Visible','Off')
   set(handles.parametros,'Visible','Off')
   set(handles.textalpha,'Visible','Off')
   set(handles.text5,'Visible','Off')
   set(handles.text6,'Visible','Off')
   set(handles.text7,'Visible','Off')
   set(handles.slider2,'Visible','Off')
   set(handles.Filtrar,'String','Filtrar')
end
if(get(hObject,'Value'))==5
   set(handles.Filtrar,'Visible','On')
   set(handles.edit1,'Visible','Off')
   set(handles.text1,'Visible','Off')
   set(handles.parametros,'Visible','Off')
   set(handles.textalpha,'Visible','Off')
   set(handles.text5,'Visible','Off')
   set(handles.text6,'Visible','Off')
   set(handles.text7,'Visible','Off')
   set(handles.slider2,'Visible','Off')
   set(handles.Filtrar,'String','Filtrar')
end



if(get(hObject,'Value'))==6
   set(handles.Filtrar,'Visible','On')
   set(handles.edit1,'Visible','Off')
   set(handles.text1,'Visible','Off')
   set(handles.parametros,'Visible','Off')
   set(handles.textalpha,'Visible','Off')
   set(handles.text5,'Visible','Off')
   set(handles.text6,'Visible','Off')
   set(handles.text7,'Visible','Off')
   set(handles.slider2,'Visible','Off')
   set(handles.Filtrar,'String','Filtrar')
end
if(get(hObject,'Value'))==7
   set(handles.Filtrar,'Visible','On')
   set(handles.edit1,'Visible','Off')
   set(handles.text1,'Visible','Off')
   set(handles.parametros,'Visible','Off')
   set(handles.textalpha,'Visible','Off')
   set(handles.text5,'Visible','Off')
   set(handles.text6,'Visible','Off')
   set(handles.text7,'Visible','Off')
   set(handles.slider2,'Visible','Off')
   set(handles.Filtrar,'String','Filtrar')
end


if(get(hObject,'Value'))==8
   set(handles.Filtrar,'Visible','On')
   set(handles.edit1,'Visible','Off')
   set(handles.text1,'Visible','Off')
   set(handles.parametros,'Visible','Off')
```

```
    set(handles.textalpha,'Visible','Off')
    set(handles.text5,'Visible','Off')
    set(handles.text6,'Visible','Off')
    set(handles.text7,'Visible','Off')
    set(handles.slider2,'Visible','Off')
    set(handles.Filtrar,'String','Filtrar')
end
if(get(hObject,'Value'))==9
    set(handles.Filtrar,'Visible','On')
    set(handles.edit1,'Visible','On')
    set(handles.edit1,'String','2')
    set(handles.text1,'Visible','On')
    set(handles.textalpha,'Visible','Off')
    set(handles.parametros,'Visible','Off')
    set(handles.text5,'Visible','Off')
    set(handles.text6,'Visible','Off')
    set(handles.text7,'Visible','Off')
    set(handles.slider2,'Visible','Off')
    set(handles.Filtrar,'String','Filtrar')
end
if(get(hObject,'Value'))==10
    set(handles.Filtrar,'Visible','On')
    set(handles.edit1,'Visible','On')
    set(handles.edit1,'String','0.2')
    set(handles.text1,'Visible','Off')
    set(handles.textalpha,'Visible','On')
    set(handles.parametros,'Visible','Off')
    set(handles.text5,'Visible','Off')
    set(handles.text6,'Visible','Off')
    set(handles.text7,'Visible','Off')
    set(handles.slider2,'Visible','Off')
    set(handles.Filtrar,'String','Filtrar')
end
if(get(hObject,'Value'))==11
     set(handles.Filtrar,'Visible','Off')
     set(handles.edit1,'Visible','Off')
     set(handles.text1,'Visible','Off')
     set(handles.textalpha,'Visible','Off')
     set(handles.text5,'Visible','Off')
     set(handles.text6,'Visible','Off')
      set(handles.text7,'Visible','Off')
     set(handles.slider2,'Visible','Off')
     set(handles.parametros,'Visible','On')
end

if(get(hObject,'Value'))==12
     set(handles.Filtrar,'Visible','On')
     set(handles.Filtrar,'String','Ecualizar')
     set(handles.edit1,'Visible','Off')
     set(handles.text1,'Visible','Off')
     set(handles.textalpha,'Visible','Off')
     set(handles.text5,'Visible','Off')
     set(handles.text6,'Visible','Off')
      set(handles.text7,'Visible','Off')
     set(handles.slider2,'Visible','Off')
     set(handles.parametros,'Visible','Off')
end
% --- Executes during object creation, after setting all properties.
function tfiltro_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to tfiltro (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as a double



% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in Guardar.
function Guardar_Callback(hObject, eventdata, handles)
% hObject    handle to Guardar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%Boton 'Guardar'
if handles.im2==0
    errordlg('No hay imagen procesada')
else
     if handles.op67==0
        if min(min(handles.im2))<0; %Normalizo la imagen para grabarla
            handles.im2 = handles.im2 - min(min(handles.im2));
        end
        handles.im2 = handles.im2 / max(max(handles.im2));
     end

guidata(hObject,handles)
[newfile,newpath] = uiputfile('*.jpg','Guardar Como');
if isequal(newfile,0)
    return
else
filename=strcat(newpath,newfile);
imwrite(handles.im2,filename);
```

```
end
end



% --- Executes on slider movement.
 function slider2_Callback(hObject, eventdata, handles)
% hObject    handle to slider2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider

% Funcion de la barra de desplazamiento para Ajuste Logaritmo
t=0;
if handles.d>=1

    t=get(hObject,'Value');
    set(handles.text7,'String',t);
    p=handles.im;
    yp=log_adj(p,t,256);
    axes(handles.graf2);
    axis off;
    imshow(yp,[]);
    handles.im2=yp;
    handles.op67=1;
    guidata(hObject,handles)
end
% --- Executes during object creation, after setting all properties.
function slider2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end



function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3 as a double


% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
```

```
%      See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end





% --- Executes on button press in logval.
function logval_Callback(hObject, eventdata, handles)
% hObject    handle to logval (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
guidata(hObject,handles);



function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%        str2double(get(hObject,'String')) returns contents of edit5 as a double


% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%      See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in parametros.
function parametros_Callback(hObject, eventdata, handles)
% hObject    handle to parametros (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% Funciones del Boton 'parametros' para Ajuste Logaritmico
log_adj_sub   % Cargo la otra subinterface
waitfor(log_adj_sub) % Espero por OK, Cancel o X

        load haypar handles1      %Hay Parametros?
        haypar1=handles1.haypar
        handles.haypar1=haypar1

        if handles.haypar1==1     %Si hay parametros

            load kmin handles1    %Cargo valores min y max del slider
```

```
        smin=handles1.kmin
        load kmax handles1
        smax=handles1.kmax
        handles.top=smax
        handles.bot=smin
        set(handles.slider2,'Min',handles.bot);
        set(handles.slider2,'Max',handles.top);
        set(handles.text5,'String',smin);
        set(handles.text6,'String',smax);
        pos=(smax+smin)/2      %Seteo la posicion inicial de la barra en la mitad
        set(handles.slider2,'Value',pos);
        set(handles.text7,'String',pos);
        clear handles1        %Limpio los datos
        delete kmax.mat
        delete kmin.mat
        delete haypar.mat
        handles.haydata=1;
        set(handles.slider2,'Visible','On')  %Solo si hay dato visualizo la barra (slider2)
        set(handles.text5,'Visible','On')
        set(handles.text6,'Visible','On')
        set(handles.text7,'Visible','On')
        p=handles.im;
        yp=log_adj(p,pos,256);
        axes(handles.graf2);
        axis off;
        imshow(yp,[]);
        handles.d=handles.d+1  % acumulador, sirve para verificacion si hubo datos
        guidata(hObject,handles)

      end



% --- If Enable == 'on', executes on mouse press in 5 pixel border.
% --- Otherwise, executes on mouse press in 5 pixel border or over slider2.
function slider2_ButtonDownFcn(hObject, eventdata, handles)
% hObject    handle to slider2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in Reutilizar.
function Reutilizar_Callback(hObject, eventdata, handles)
% hObject    handle to Reutilizar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%Funciones del boton 'Reutilizar'
if handles.im2==0
   errordlg('No hay imagen procesada')
else
    if handles.op67==0
       if min(min(handles.im2))<0; %Normalizo la imagen para grabarla
           handles.im2 = handles.im2 - min(min(handles.im2));
       end
       handles.im2 = handles.im2 / max(max(handles.im2));
    end

guidata(hObject,handles)
```

```
imwrite(handles.im2,'temp.jpg');
temp=imread('temp.jpg');
delete temp.jpg
handles.im=temp;
axes(handles.graf1);
axis off;
imshow(temp);
axes(handles.graf2)
axis off
cla reset
axis off
set(handles.slider2,'Visible','Off')  %Solo si hay dato visualizo la barra (slider2)
set(handles.text5,'Visible','Off')
set(handles.text6,'Visible','Off')
set(handles.text7,'Visible','Off')
guidata(hObject,handles)
end
```