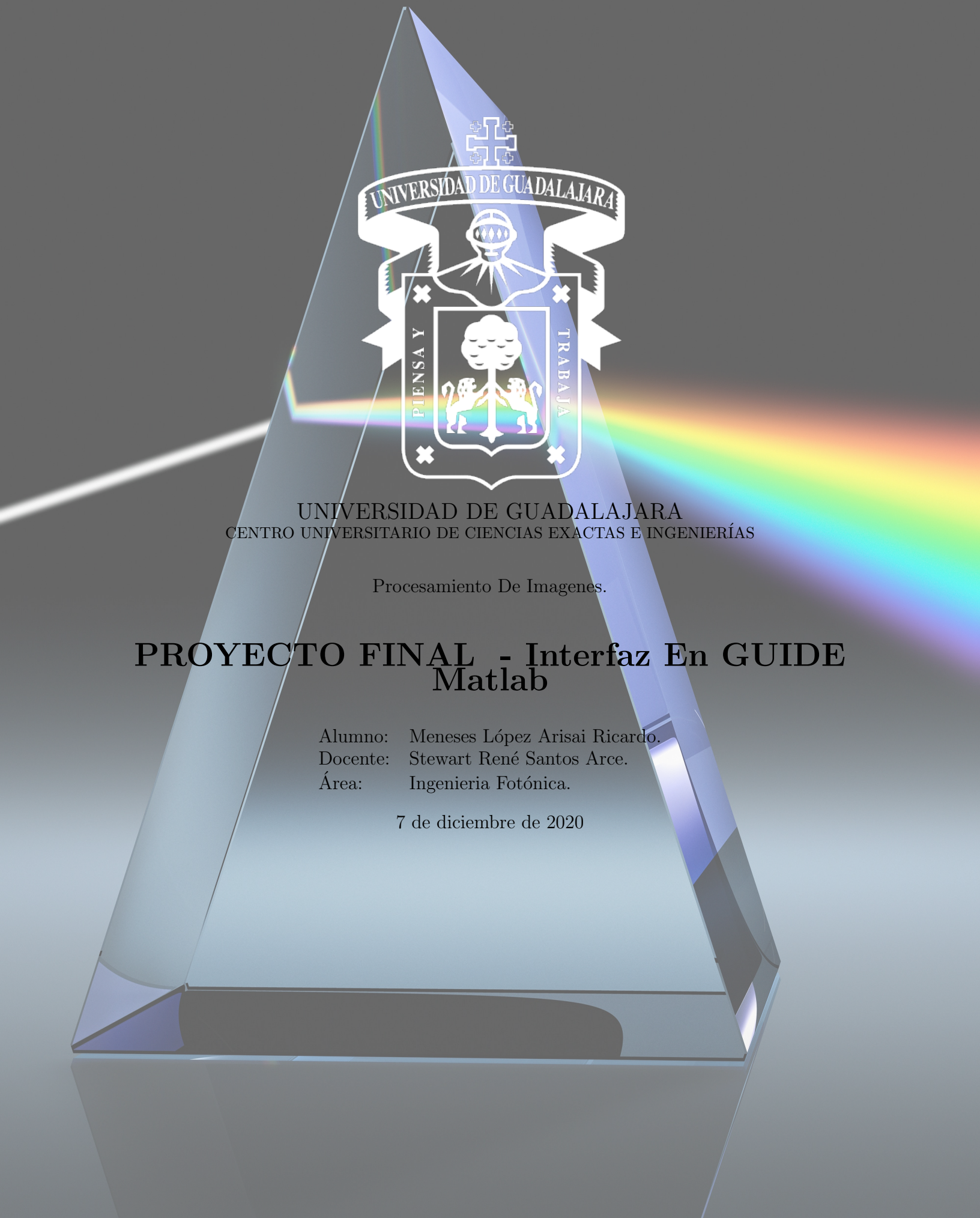


UNIVERSIDAD DE GUADALAJARA
CENTRO UNIVERSITARIO DE CIENCIAS EXACTAS E INGENIERÍAS

Procesamiento De Imagenes.

PROYECTO FINAL - Interfaz En GUIDE Matlab

Alumno: Meneses López Arisai Ricardo.
Docente: Stewart René Santos Arce.
Área: Ingenieria Fotónica.

7 de diciembre de 2020

Índice

1. Introducción

El procesamiento digital de imágenes es el conjunto de técnicas englobadas cuyo objetivo fundamental es obtener, a partir de una imagen origen, una nueva imagen cuyo resultado sea más adecuado para una aplicación específica mejorando ciertas características de la misma que posibilite efectuar operaciones del procesado sobre ella.

Una imagen digital se compone de:

- **Resolución Espectral**, Se define como el número y ancho de bandas espectrales que un sensor puede registrar. Más numero de bandas espectrales significa menor anchura entre las bandas y se consigue una mejor resolución espectral para los distintos canales que presenta una imagen en un distinto formato.
- **Resolución Espacial**, Designa al objeto más pequéño que se puede distingir en la imagen y está determinado por el tamaño del pixel.
- **Resolución Radiométrica**, Se define como el número de bits almacenados por pixel el cual hace referencia al muestreo.
- **Resolución temporal**, Se define como el periodo de tiempo entre imágenes consecutivas detectadas por un sensor.

Algunas funciones y propiedades utilizadas en el entorno para diseñar la interfaz son:

- **handles.** - Sirve como identificador o puntero para referirnos al nombre (TAG) de los objetos gráficos que tiene la iterfaz.
- **get()** - La función se usa para obtener valores y su sintaxis es: get(Identificador.NombreDelObjeto,'AtributoDelObjeto').
- **set()** - La función se utiliza para modificar algún atributo de los objetos, su sintaxis es: set(Identificador.NombreObjeto,'AtributoObjeto',NuevoValorObjeto).
- **assignin()** - Se usa para 'almacenar variables' en el workspace, su sintaxis es: assignin('base','VariableLocal','N
- **evalin()** - La función se utiliza para comunicar variables entre funciones, previamente las variables deben estar almacenadas en el workspace para poder ser llamadas, su sintaxis es: evalin('base','NombreDeLaVariableAlmacenada').
- **num2str()** - Función que convierte una variable de tipo númerico a string, su sintaxis es: num2string(VariableAConvertir).
- **char()** - Función que convierte las variables a tipo char, su sintaxis es: char(VariableAConvertir).
- **Gray()** - Función que convierte la imagen a escala de grises, en caso de que la imagen sea gris, entonces no hace dicha conversión para no generar errores, su sintaxis es: Gray(Imagen).
- **Kernel()** - Función que contiene núcleos de transformación que sirve para realizar filtros en el el dominio espacial. Su estructura depende del núcleo que se quiera usar y normalmente sus parámetros de entrar son de tipo string.

- **Transformation()** - Función que realiza el movimiento requerido a la imagen, su sintaxis es: Transformation(Imagen,MatrizDeTransformación).
- **Callback** - Función que toma parte de la creación de cada objeto que se añade a la interfaz, solamente pertenece al objeto creado. Su trabajo es realizar todo aquello que contenga dentro de la misma. Puede ser llamada cada vez que se utiliza un objeto (Como respuesta) o bien internamente en el código se hace mención a la función a través de la siguiente manera: $NombreDelObjeto\_Callback(hObject, eventdata, handles)$.

2. Objetivos

- Diseñar una interfaz gráfica en GUIDE Matlab donde se incluyan los algorítmos presentados a lo largo del curso de procesamiento digital de imágenes 2020 B.
- El diseño debe ser simple y sin saturación de objetos o colores con propósito de facilitar el manejo y entendimiento de la misma.
- Es necesario reutilizar cada objeto presente en el diseño interno de la interfaz con la finalidad de tener menos objetos.
- Se debe tener la opción de guardar los cambios aplicados, así como el almacenamiento directo de la imagen modificada.
- Se debe mostrar la imagen en todo momento de ejecución de la interfaz.

3. Desarrollo

Teniendo en cuenta los objetivos presentados anteriormente, se tiene como comienzo la interfaz en su punto inicial donde se muestra una imagen en color blanco y además un menú horizontal estático donde se habilita una única opción la cual es 'Cargar Imagen' y se encuentra en la barra superior de la interfaz.

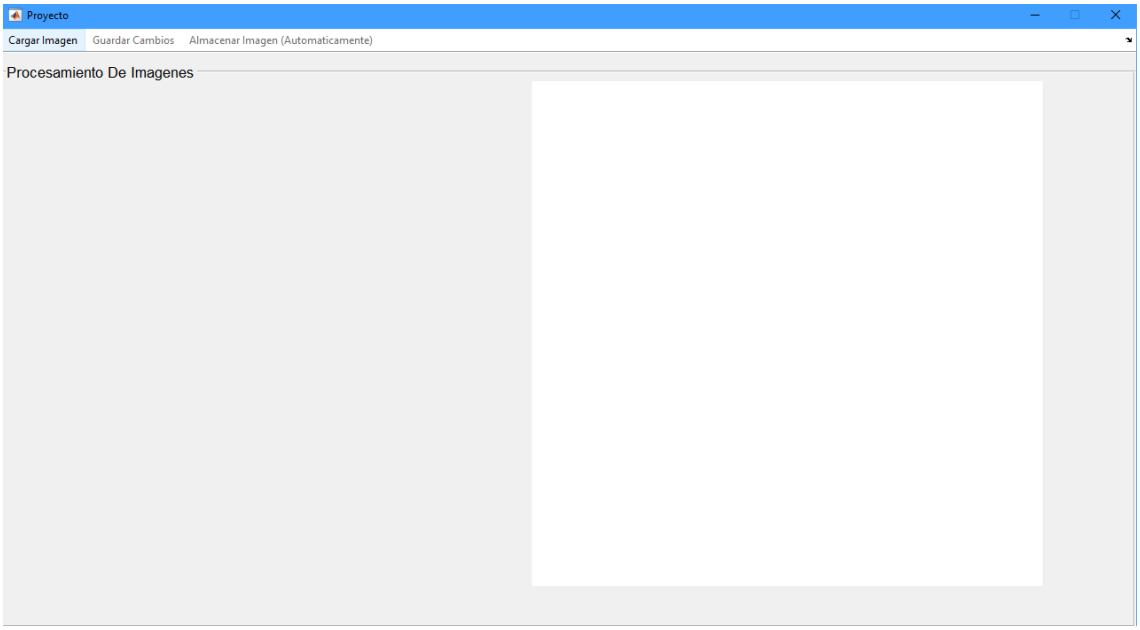


Figura 1: Interfaz - Inicio

Las opciones mostradas son:

- **Cargar imagen** - Habilitada en todo momento y su función principal es obtener una imagen desde el computador, si el proceso es cancelado se mostrará la imagen en blanco y una ventana emergente, además obtiene el 'pathname' y el 'filename' de la imagen.
- **Guardar cambios** - Se habilita hasta que se realice un cambio o modificación en la imagen actual y trabajar en base al cambio hecho. Mientras el cambio no sea guardado se tendrá la imagen actual para su manejo.
- **Almacenar Imagen Automáticamente** - Se habilita cuando se guarda un cambio y se deshabilita cuando el cambio es cancelado. Su función principal es almacenar la imagen en el mismo folio y formato, pero con nombre similar, ejemplo: $NombreImagen.jpg$ pasa a $NombreImagenN.jpg$.

A continuación el código de la función 'Cargar Imagen':

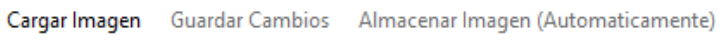


Figura 2: Interfaz - Menú Superior

```
function CargarImagen_Callback(hObject, eventdata, handles) %Abrir Imagen
hold off; clc;
[filename, pathname] = uigetfile( ...
       '*.jpg', ...
        'Pick a file', ...
        'MultiSelect', 'on');
if isequal(filename,0)||isequal(pathname,0)
    disp('User Pressed "Cancel"');   warndlg('Elige una Imagen', 'Advertencia');  set(handles.uibuttongroupl,'Visible','Off');
    set(handles.PanelMR,'Visible','Off');   set(handles.PanelF,'Visible','Off');  set(handles.Dimentions,'Visible','Off');
    x=ones(2000,2000,'double'); axes(handles.axes1);  imshow(x);  set(handles.AlmacenarImagen,'Enable','Off');
    set(handles.AplicarF,'Visible','Off'); set(handles.AplicarMR,'Visible','Off'); set(handles.GuardarCambios,'Enable','Off');
else
    x=imread([pathname, filename]); x=im2double(x); axes(handles.axes1); imshow(x);
    assignin('base','pathname',pathname);    assignin('base','filename',filename);
    assignin('base','x',x);                  xi=x; assignin('base','xi',xi);
    set(handles.GuardarCambios, 'Enable', 'on'); set(handles.uibuttongroupl, 'Visible', 'on');
    set(handles.PanelMR,'Visible','Off');   set(handles.PanelF,'Visible','Off');
    set(handles.checkboxR, 'Enable', 'on'); [M,N,O]=size(x);  str1 = [num2str(N)];str2 = [num2str(M)];
    str3=[num2str(O)];str1=strcat(str1,'x'); str2=strcat(str2,'x'); set(handles.Dimentions,'Visible','On');
    set(handles.AplicarF,'Visible','Off'); set(handles.AplicarMR,'Visible','Off'); set(handles.GuardarCambios,'Enable','Off');
    str=strcat(str1,str2); str=strcat(str,str3); set(handles.Dimentions,'String',str);  set(handles.AlmacenarImagen,'Enable','Off');
end
```

Figura 3: Código - Cargar Imagen

La estructura del proyecto se divide en cuatro secciones donde cada una crea un menú distinto reutilizando el mismo objeto gráfico (popupmenu). Dentro del menú existen submenús que contienen funciones o procesos donde se pueden realizar modificaciones a la imagen, dichos cambios se limitan a la sección o subsección donde se encuentran.
A continuación se muestran las secciones y su 'checkbox' correspondiente agrupados en un 'button group'. También se presenta el código para cada función que se asignó a su respectiva seccion donde se muestran los distintos menús que se crean.

La logica del programa es que primero debes seleccionar alguna opción de las que se van mostrando para así poder continuar y llegar a realizar modificaciones a la imagen. Las opciones van apareciendo conforme a lo que se vaya eligiendo.

Un ejemplo para simplificar es: Cargas la imagen, seleccionas la sección (Movimiento, Resolución, etc.) entonces aparecen los menús para la sección, una vez elejido la operación a realizar (dentro del menú), entonces aparecen los objetos con los que se puede interaccionar, cuando se modifican los sliders en seguida aparece un botón para aplicar los cambios y finalmente cuando es presionado entonces se realiza el procesamiento sobre la imagen y entonces se muestran los resultados en un display.

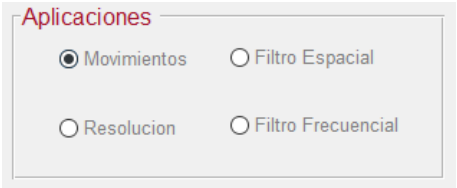


Figura 4: Interfaz - Menú De Aplicaciones

```
function Movimiento_Callback(hObject, eventdata, handles)
MenuText={'- Giro -';'- Traslacion -';'- Inclinacion -'};
set(handles.MenuBarraMR, 'String', MenuText); set(handles.PanelMR, 'Title', 'Movimientos');
set(handles.PanelMR, 'Visible', 'On');        set(handles.PanelF, 'Visible', 'Off');
set(handles.TextXMR,'Visible','Off');         set(handles.SliderXMR,'Visible','Off');
set(handles.TextYMR,'Visible','Off');         set(handles.SliderYMR,'Visible','Off');
set(handles.ValueYMR,'Visible','Off');        set(handles.ValueXMR,'Visible','Off');
set(handles.TextZMR,'Visible','Off');         set(handles.ValueZMR,'Visible','Off');
set(handles.MenuBarraMR,'Value',1);           set(handles.SliderZMR,'Visible','Off');
set(handles.uipanel8,'Visible','Off');        set(handles.AplicarMR,'Visible','Off');
Aopc=1; assignin('base','Aopc', Aopc);
```

Figura 5: Código - Sección Movimientos

```
function Resolu_Callback(hObject, eventdata, handles)
Menu1={'         <Escalado>';'         <Producto Kronecker>';'         <Zero Padding>';'         <Submuestreo>'};
Menu2={'         <Cambio Capas>';'         <Discriminacion Capas>';'         <Factor De Color>'};
Menu3={'         <Número De Bits>'};
MenuText={'- Espacial -';char(Menu1);'- Espectral -';char(Menu2);'- Radiométrica -';char(Menu3)};
set(handles.MenuBarraMR, 'String', MenuText); set(handles.PanelMR, 'Title', 'Resolucion');
set(handles.PanelMR, 'Visible', 'On');   set(handles.PanelF, 'Visible', 'Off');
set(handles.TextXMR,'Visible','Off');    set(handles.SliderXMR,'Visible','Off');
set(handles.TextYMR,'Visible','Off');    set(handles.SliderYMR,'Visible','Off');
set(handles.ValueYMR,'Visible','Off');   set(handles.ValueXMR,'Visible','Off');
set(handles.TextZMR,'Visible','Off');    set(handles.ValueZMR,'Visible','Off');
set(handles.MenuBarraMR,'Value',1);      set(handles.SliderZMR,'Visible','Off');
set(handles.uipanel8,'Visible','Off');   set(handles.AplicarMR,'Visible','Off');
Aopc=2; assignin('base','Aopc', Aopc);
```

Figura 6: Código - Sección Resolución

```
function FiltroEsp_Callback(hObject, eventdata, handles)
SobelMenu={'              Inferior';'              Superior';'              Derecha';'              Izquierda'};
PrewittMenu={'              Horizontal';'              Vertical';'              +45°';'              -45°'};
Menu1={'        <Laplace 9 Puntos>';'        <Laplace 5 Puntos Laterales>';...
    '        <Laplace 5 Puntos Esquinados>';'        <Sobel>';char(SobelMenu);'        <Prewitt>';char(PrewittMenu)};
Menu2={'        <Dimension Impar>'};
MenuText={'- Pasa Altas -';char(Menu1);'- Pasa Bajas -';char(Menu2)};
set(handles.MenuBarraF, 'String', MenuText); set(handles.PanelF, 'Title', 'Filtro Espacial');
set(handles.PanelF, 'Visible', 'On');        set(handles.PanelMR, 'Visible', 'Off');
set(handles.TextXF,'Visible','Off');         set(handles.SliderYF,'Visible','Off');
set(handles.TextYF,'Visible','Off');         set(handles.SliderXF,'Visible','Off');
set(handles.ValueYF,'Visible','Off');        set(handles.ValueXF,'Visible','Off');
set(handles.MenuBarraF,'Value',1);           set(handles.AplicarF,'Visible','Off');
Bopc=1; assignin('base','Bopc', Bopc);
```

Figura 7: Código - Sección Filtros Espaciales

```
function FiltroFrec_Callback(hObject, eventdata, handles)
Menu1={'        <Rectangular>';'        <Gaussiano>';'        <Gauss Modificado>';...
    '        <Coseno>';'        <Hanning>';'        <Bartlet>'};
Menu2={'        <Rectangular>';'        <Gaussiano>';'        <Gauss Modificado>';...
    '        <Coseno>';'        <Hanning>';'        <Bartlet>'};
MenuText={'- Pasa Altas -';char(Menu1);'- Pasa Bajas -';char(Menu2)};
set(handles.MenuBarraF, 'String', MenuText); set(handles.PanelF, 'Title', 'Filtro Frecuencial');
set(handles.PanelF, 'Visible', 'On');        set(handles.PanelMR, 'Visible', 'Off');
set(handles.TextXF,'Visible','Off');         set(handles.SliderYF,'Visible','Off');
set(handles.TextYF,'Visible','Off');         set(handles.SliderXF,'Visible','Off');
set(handles.ValueYF,'Visible','Off');        set(handles.ValueXF,'Visible','Off');
set(handles.MenuBarraF,'Value',1);           set(handles.AplicarF,'Visible','Off');
Bopc=2; assignin('base','Bopc', Bopc);
```

Figura 8: Código - Sección Filtros Frecuenciales

3.0.1. Movimientos

La opción Movimientos Habilita un menú con las siguientes opciones:

- **Giro** - Permite la rotación de la imagen a color o gris en un máximo de 360 grados.
- **Traslación** - Permite la traslación de la imagen a color o gris con un máximo de $\frac{1}{2}$ del tamaño de la imagen en cordenadas "X" y "Y". Dicho proceso se logra a través de la matriz de trasformacion: T=[1 0 Cx ; 0 1 Cy ; 0 0 1].
- **Inclinación** - Permite la inclinación de la imagen a color o gris en cordenadas "X" y "Y" con un maximo de 3 puntos (enteros). Este movimiento se logra a travès de la matriz de transformación: I=[1 Cx 0 ; Cy 1 0 ; 0 0 1].

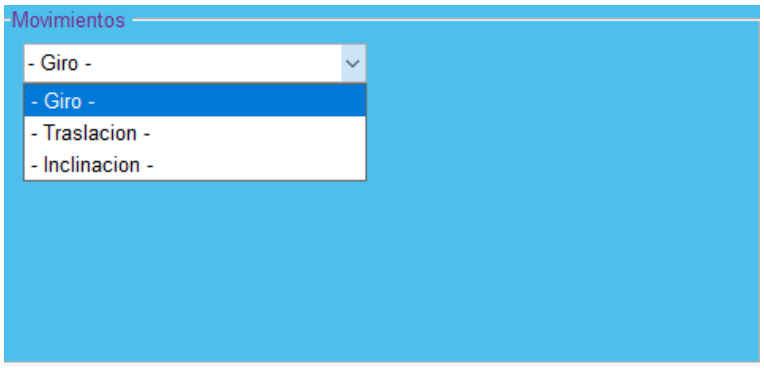
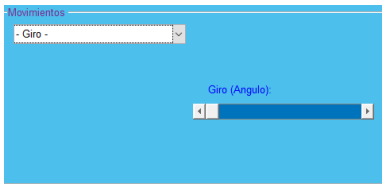
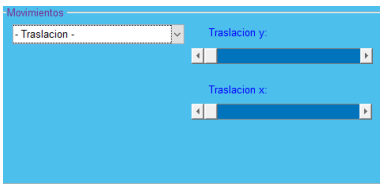


Figura 9: Interfaz - Menú Movimientos

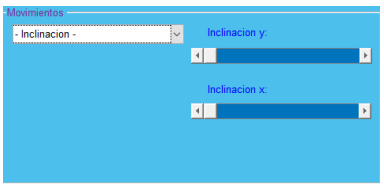
- Cuando se selecciona una de las tres opciones del menú se muestran los objetos requeridos para utilizar la función y modificar la imagen.



(a) Giro



(b) Traslación



(c) Inclinación

Figura 10: Interfaz - Movimientos: Giro, Traslación e Inclinación

En seguida se presenta el código dentro de la función **'MenuBarraMR'**. Esta función engloba las secciones 'Movimiento' y 'Resolución'. Únicamente habilita los objetos necesarios y modifica los textos presentados en los 'StaticText' para cada caso seleccionado en el menú de ambas secciones. Esta función no aplica ningún cambio directo sobre la imagen.

```
function MenuBarraMR_Callback(hObject, eventdata, handles)
opc=get(handles.MenuBarraMR,'Value');       Aopc=evalin('base','Aopc');Steps=[1/100, 1/10];
set(handles.SliderXMR,'SliderStep', Steps); set(handles.SliderYMR,'SliderStep', Steps); set(handles.SliderZMR,'SliderStep', Steps);
set(handles.ValueYMR,'String','');          set(handles.ValueXMR,'String','');        set(handles.ValueZMR,'String','');
set(handles.SliderZMR,'Value',0);           set(handles.SliderXMR,'Value',0);         set(handles.SliderYMR,'Value',0);
set(handles.TextXMR,'Visible','Off');       set(handles.TextYMR,'Visible','Off');     set(handles.TextZMR,'Visible','Off');
set(handles.ValueXMR,'Visible','Off');      set(handles.ValueYMR,'Visible','Off');    set(handles.ValueZMR,'Visible','Off');
set(handles.SliderXMR,'Visible','Off');     set(handles.SliderYMR,'Visible','Off');   set(handles.SliderZMR,'Visible','Off');
set(handles.uipanel8,'Visible','Off');      set(handles.uipanel9,'Visible','Off');    set(handles.AplicarMR,'Visible','Off');

switch Aopc
    case 1 % Movimientos
        switch opc
            case 1  %Giro
                set(handles.TextXMR,'String','Giro (Angulo):');
                set(handles.TextXMR,'Visible','On');   set(handles.SliderXMR,'Visible','On');
                set(handles.ValueXMR,'Visible','On');
            case 2  %Traslacion
                set(handles.TextXMR,'String','Traslacion x:'); set(handles.TextYMR,'String','Traslacion y:');
                set(handles.TextXMR,'Visible','On');           set(handles.SliderXMR,'Visible','On');
                set(handles.TextYMR,'Visible','On');           set(handles.SliderYMR,'Visible','On');
                set(handles.ValueYMR,'Visible','On');           set(handles.ValueXMR,'Visible','On');
            case 3  %Inclinacion
                set(handles.TextXMR,'String','Inclinacion x:'); set(handles.TextYMR,'String','Inclinacion y:');
                set(handles.TextXMR,'Visible','On');            set(handles.SliderXMR,'Visible','On');
                set(handles.TextYMR,'Visible','On');            set(handles.SliderYMR,'Visible','On');
                set(handles.ValueYMR,'Visible','On');            set(handles.ValueXMR,'Visible','On');
        end
```

Figura 11: Código - Menú Movimientos

- Una vez hecho el cambio en alguno de los deslizadores se habilitará el botón para poder aplicar los cambios directamente en la imagen actual. Dicho botón cambiará de color para indicar visualmente que el programa está en ejecución y regresará a su color base cuando el proceso haya terminado.

(a) Aplicar Cambios - No ejecutado

(b) Aplicar Cambios - En Ejecución

Figura 12: Interfaz - Botón Aplicar Cambios

- Una vez aplicados los cambios se muestran los respectivos valores para cada función así como la imagen resultante. La función **'AplicarMR'** realiza todos los procesamientos en las imagenes de la sección 'Movimientos' y 'Resolución'.

```
function AplicarMR_Callback(hObject, eventdata, handles)
opc=get(handles.MenuBarraMR,'Value'); Aopc=evalin('base','Aopc'); x=evalin('base','x'); xi=evalin('base','xi'); [M,N,O]=size(x);
set(handles.AplicarMR, 'Enable', 'off'); set(handles.AplicarMR, 'BackgroundColor', 'red'); pause(.5);
switch Aopc
    case 1 % Movimientos
        switch opc
            case 1 %Giro
                    a=round(evalin('base','Cx')*360); Theta=degtorad(a);  H=[cos(Theta), -sin(Theta);sin(Theta), cos(Theta)];
                    angulo=(Theta>=0 && Theta <pi/2) + 2*(Theta>=pi/2 && Theta <pi)...
                    +3*(Theta>=pi && Theta <3*pi/2) + 4*(Theta>=3*pi/2 && Theta <=2*pi);
                    switch angulo
                      case 1 % entre 0 y pi/2
                        sumu=round(0*cos(Theta)-N*sin(Theta)); sumv=0;
                      case 2 % entre pi/2 y pi
                        sumu=round(M*cos(Theta)-N*sin(Theta)); sumv=round(0*sin(Theta)+N*cos(Theta));
                      case 3 % entre pi y 3*pi/2
                        sumu=round(M*cos(Theta)-0*sin(Theta)); sumv=round(M*sin(Theta)+N*cos(Theta));
                      case 4 % entre 3*pi/2 y 2*pi
                        sumu=0; sumv=round(M*sin(Theta)+0*cos(Theta));
                    end
                    for x1=1:M
                      for y=1:N
                        uv=round(H*[x1;y]); u=uv(1)-sumu; v=uv(2)-sumv; u=u*(u>=1)+(u<1); v=v*(v>=1)+(v<1); Y(u,v,:)=x(x1,y,:);
                      end
                    end
                    Numero=num2str(a); Numero=strcat(Numero,'°'); set(handles.ValueXMR,'String',Numero);
                    axes(handles.axes1);  imshow(Y);  xi=Y; assignin('base','xi',xi);
```

Figura 13: Código - Aplicar: Movimientos (Giro)

```
            case 2 %Traslacion
                Cx=evalin('base','Cx');  Cy=evalin('base','Cy');
                Cx=round((M/2)*Cx); Cy=round((N/2)*Cy);          T=[1 0 Cx ; 0 1 Cy ; 0 0 1];
                set(handles.ValueXMR,'String',num2str(Cx));      set(handles.ValueYMR,'String',num2str(Cy));
                Y=Transformation(x,T);  axes(handles.axes1);  imshow(Y);  xi=Y; assignin('base','xi',xi);
            case 3 %Inclinacion
                Cx=evalin('base','Cx'); Cx=round(3*Cx);
                Cy=evalin('base','Cy'); Cy=round(3*Cy);       T=[1 Cx 0 ; Cy 1 0 ; 0 0 1];
                set(handles.ValueXMR,'String',num2str(Cx));      set(handles.ValueYMR,'String',num2str(Cy));
                Y=Transformation(x,T);  axes(handles.axes1);  imshow(Y);  xi=Y; assignin('base','xi',xi);
        end
    case 2 % Resolucion
        switch opc
            case 2  %Escalado
                Cx=evalin('base','Cx'); Cx=(5*Cx);  Cy=evalin('base','Cy');  Cy=(5*Cy);  T=[Cx 0 0 ; 0 Cy 0 ; 0 0 1];
                set(handles.ValueXMR,'String',num2str(Cx));        set(handles.ValueYMR,'String',num2str(Cy));
                Y=Transformation(x,T);    axes(handles.axes1);  imshow(Y);  xi=Y; assignin('base','xi',xi);
            case 3  %Producto Kronecker
                Cx=evalin('base','Cx');   Cx=round(4*Cx)+1;      Cy=evalin('base','Cy');   Cy=round(4*Cy)+1;
                set(handles.ValueXMR,'String',num2str(Cx));      set(handles.ValueYMR,'String',num2str(Cy));
                if O==3
                h=ones(Cy,Cx);  r=x(:,:,1);  g=x(:,:,2);  b=x(:,:,3);
                y1=kron(r,h);  y2=kron(g,h); y3=kron(b,h); Y(:,:,1)=y1;   Y(:,:,2)=y2;  Y(:,:,3)=y3;
                else
                h=ones(Cy,Cx); x=Gray(x); Y=kron(x,h);
                axes(handles.axes1); imshow(Y);   xi=Y; assignin('base','xi',xi);
                end
                axes(handles.axes1); imshow(Y);   xi=Y; assignin('base','xi',xi);
```

Figura 14: Código - Aplicar: Movimientos (Traslación, Inclinación) y Resolución (Escalado, Producto Kronecker)

3.0.2. Resolución

Resolución Habilita un menú con las siguientes opciones, cada opción contiene un submenú:

- **Espacial** - Permite la modificación de las dimensiones de la imagen en en cordenadas "X" y "Y". Si el tamaño de la imagen en ancho y altura no es menor a 1500 pixeles, entonces no se podrá ingresar a cualquier función de 'Sobremuestreo'.
  Escalado - Permite la amplificación mediante la matriz de transformación aplicado a la imagen con un máximo de 5 veces el tamaño de la imagen en cordenadas "X" y "Y". Dicho Sobremuestreo se logra a través de la matriz de transformaciòn: E=[Cx 0 0 ; 0 Cy 0 ; 0 0 1].

  Producto Kronecker - Permite la amplificación mediante el productor kronecker con un máximo de 5 veces el tamaño de la imagen en cordenadas "X" y "Y".

  Zero Padding - Permite la amplificación mediante el algoritmo zero padding con un máximo de 5 veces el tamaño de la imagen en cordenadas "X" y "Y".

  Submuestreo - Permite la modificación de las muestras que puede tener la imagen en las cordenadas "X" y "Y". El sobremuestreo se logra a travez del siguiente comando $X_{sub} = X_{image}[1 : M : end; 1 : N : end; :]$.

- **Espectral** - Permite la modificación del orden, agrupación e intensidad de las capas RGB. La función espectral trabaja con imagenes a color por lo que si la imagen no tiene una profundidad de 3 capas, la interfaz no permitirá ingresar a realizar cambios en este apartado.

  Cambio Capas - Permite el reordenamiento en la secuencia de las capas de la imagen (RGB, GBR, BRG, BGR, GRB y RBG). La secuencia se introduce mediante tres botónes los cuales cambian de color para indicar que fue presionado, además se inhabilitan al momento de ser activados.

  Discriminación Capas - Permite la selección de habilitar o deshabilitar las distintas capas de la imagen (RGB, RG, RB, GB, R, G y B). Se logra mediante la verificación de 3 checkbox los cuales añadiran o no su respectiva capa a la imagen con el orden de Capas que actualmente posee la imagen. Si el orden de estas es previamente alterada entonces puede que alguna de las 3 capas no corresponda al momento de realizar la discriminación de capas, dicho esto está la posibilidad de observar una incongruencia visual al momento de discriminar las capas con un orden distinto al original (RGB).

  Factor De Color - Permite la modulación de intensidad de cada capa (RGB) que tiene la imagen, dando como resultado una combinación de las distintas intensidades.

- **Radiométrica** - Permite la modificación del número de bits que puede tener la imagen por pixel.

Número De Bits - Permite la modificación del número de canales (niveles de grises) que tiene la imagen. Convierte la imagen a escala de grises para realizar la modificación. La modificación del número de bits se logra mediante: $n = 2^k$ y $X_{rad} = round(n * x_{image})/n$, siendo 'k' el valor de bits que va desde 1 hasta 8 (o más).

A continuación se ilustrará el menú de la sección **'Resolución'** y su respectivo código alojado en la función **'MenuBarraMR'**. Dicha función sólo modifica la apariencia del panel; activando y desactivando los objetos dependiendo de cada caso. Cabe señalar que la función 'MenuBarraMR' aloja las modificaciones de objetos del menú de 'Movimientos' y el menú 'Resolución'.

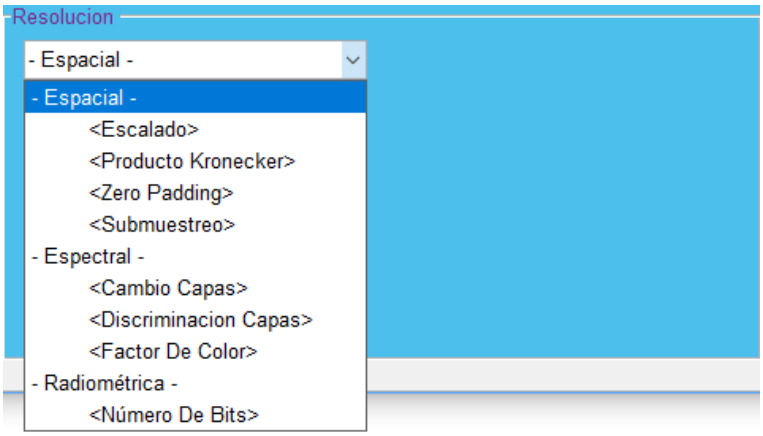


Figura 15: Menú Movimientos

```
case 2 % Resolucion
    switch opc
        case 2  %Escalado
            [M,N,Q]=size(evalin('base','x'));
            if M<1500&&N<1500
            set(handles.TextXMR,'String','Escalado x:'); set(handles.TextYMR,'String','Escalado y:');
            set(handles.TextXMR,'Visible','On');         set(handles.SliderXMR,'Visible','On');
            set(handles.TextYMR,'Visible','On');         set(handles.SliderYMR,'Visible','On');
            set(handles.ValueYMR,'Visible','On');        set(handles.ValueXMR,'Visible','On');
            else
                warndlg('Elige Una Imagen Más Pequeña', 'Mensaje');
            end
        case 3  %Kronecker
            [M,N,Q]=size(evalin('base','x'));
            if M<1500&&N<1500
            set(handles.TextXMR,'String','Kronecker en x:'); set(handles.TextYMR,'String','Kronecker en y:');
            set(handles.TextXMR,'Visible','On');             set(handles.SliderXMR,'Visible','On');
            set(handles.TextYMR,'Visible','On');             set(handles.SliderYMR,'Visible','On');
            set(handles.ValueYMR,'Visible','On');            set(handles.ValueXMR,'Visible','On');
            else
                warndlg('Elige Una Imagen Más Pequeña', 'Mensaje');
            end
```

Figura 16: Código - Menú Resolución

```
case 4  %Padding
    [M,N,O]=size(evalin('base','x'));
    if M<1500&&N<1500
    set(handles.TextXMR,'String','Padding en x:'); set(handles.TextYMR,'String','Padding en y:');
    set(handles.TextXMR,'Visible','On');           set(handles.SliderXMR,'Visible','On');
    set(handles.TextYMR,'Visible','On');           set(handles.SliderYMR,'Visible','On');
    set(handles.ValueYMR,'Visible','On');          set(handles.ValueXMR,'Visible','On');
    else
        warndlg('Elige Una Imagen Más Pequeña', 'Mensaje');
    end
case 5  %Submuestreo
    x=evalin('base','x');  [M,N,O]=size(x);
    set(handles.TextXMR,'String','Submuestreo x:'); set(handles.TextYMR,'String','Submuestreo y:');
    set(handles.TextXMR,'Visible','On');            set(handles.SliderXMR,'Visible','On');
    set(handles.TextYMR,'Visible','On');            set(handles.SliderYMR,'Visible','On');
    set(handles.ValueYMR,'Visible','On');           set(handles.ValueXMR,'Visible','On');
    Steps1=[1/((M)-1), 1/((M*10)-1)];               Steps2=[1/((N)-1), 1/((N*10)-1)];
    set(handles.SliderXMR,'SliderStep', Steps1);    set(handles.SliderYMR,'SliderStep', Steps2);
case 7  %Cambio De Orden Capas RGB
    x=evalin('base','x');  [M,N,O]=size(x);
    if O<3
        warndlg('Elige Una Imagen A Color', 'Mensaje');
    else
        set(handles.uipanel9,'Visible','On');                set(handles.pushbuttonB,'Enable','On');
        set(handles.pushbuttonG,'Enable','On');              set(handles.pushbuttonR,'Enable','On');
        set(handles.pushbuttonG,'BackgroundColor','White');  set(handles.pushbuttonR,'BackgroundColor','White');
        set(handles.pushbuttonB,'BackgroundColor','White');  RedButton=0; GreenButton=0; BlueButton=0;
        assignin('base','RedButton',RedButton); assignin('base','GreenButton',GreenButton); assignin('base','BlueButton',BlueButton);
        set(handles.uipanel9,'Visible','On');   x=evalin('base','x'); [M,N,O]=size(x);      xi=zeros(M,N,O);  assignin('base','xi',xi);
    end
```

Figura 17: Código - Menú Resolución

```
    case 8  %Discrimacion De Capas
            x=evalin('base','x');  [M,N,O]=size(x);
        if O<3
            warndlg('Elige Una Imagen A Color', 'Mensaje');
        else
            set(handles.uipanel8,'Visible','On');  x=evalin('base','x'); [M,N,O]=size(x); XR=zeros(M,N,O);  XG=XR;   XB=XR;
            assignin('base','XR',XR);  assignin('base','XB',XB);  assignin('base','XG',XG);
            set(handles.checkboxR,'Value',0);  set(handles.checkboxG,'Value',0);  set(handles.checkboxB,'Value',0);
        end
    case 9  %Modificacion Del Factor De Color
        x=evalin('base','x');  [M,N,O]=size(x);
        if O<3
            warndlg('Elige Una Imagen A Color', 'Mensaje');
        else
            set(handles.TextYMR,'String','RED:');  set(handles.TextXMR,'String','GREEN:');   set(handles.TextZMR,'String','BLUE:');
            set(handles.TextXMR,'Visible','On');   set(handles.TextYMR,'Visible','On');      set(handles.TextZMR,'Visible','On');
            set(handles.ValueYMR,'Visible','On');  set(handles.ValueXMR,'Visible','On');     set(handles.ValueZMR,'Visible','On');
            set(handles.SliderYMR,'Visible','On'); set(handles.SliderXMR,'Visible','On');    set(handles.SliderZMR,'Visible','On');
        end
    case 11  %Radiometrica
        set(handles.TextXMR,'String','Numero De Bits:');  set(handles.ValueXMR,'Visible','On');
        set(handles.TextXMR,'Visible','On');              set(handles.SliderXMR,'Visible','On');
end
```

Figura 18: Código - Menú Resolución

- Cuando se realiza un cambio en algún deslizador se tiene la opción de aplicar el cambio con el pushbutton **'Aplicar'**. Todo los algorítmos para realizar cambios o modificaciones en la 'resolución' y 'movimiento' se alojan en el ya mencionado pushbutton y su función en el código se llama 'AplicarMR'.

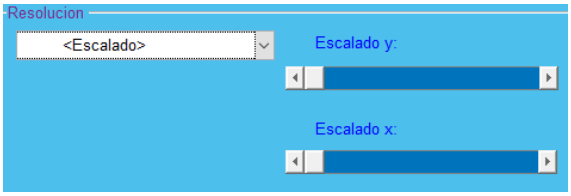
```
case 4  %Zero Padding
    Cx=evalin('base','Cx');   Cy=evalin('base','Cy');
    k1=(round(4*Cx)+1);  k2=(round(4*Cy)+1);
    [M,N,O]=size(x);          Xf=fftshift(fft2(x));
    newM=round(M.*k1);        newN=round(N.*k2);   Yf=zeros(newM ,newN,O);
    A=(newM-M)/2+1:(newM-M)/2+M;                   B=(newN-N)/2+1:(newN-N)/2+N;      Yf(A,B,:)=Xf;
    Y=ifft2(ifftshift(Yf));                        Y=Y/max(Y(:));
    set(handles.ValueXMR,'String',num2str(k1));    set(handles.ValueYMR,'String',num2str(k2));
    axes(handles.axes1);    imshow(Y);    xi=Y;    assignin('base','xi',xi);
case 5  %Submuestreo
    n1=round((M)*get(handles.SliderXMR,'Value'))+1;  n2=round((N)*get(handles.SliderYMR,'Value'))+1;
    set(handles.ValueXMR,'String',num2str(n1));       set(handles.ValueYMR,'String',num2str(n2));
    Y=x(1:n1:end,1:n2:end,:); axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi);
case 7  %Cambio De Orden En Capas RGB
    axes(handles.axes1); imshow(xi); set(handles.AplicarMR,'Visible','Off');
case 8  %Discriminacion De Capas
    XR=evalin('base','XR'); XG=evalin('base','XG'); XB=evalin('base','XB');
    Y(:,:,1)=XR(:,:,1);  Y(:,:,2)=XG(:,:,2);  Y(:,:,3)=XB(:,:,3);
    axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi);
case 9  %Modificacion Factor Del Color
    [M,N,O]=size(x); XR=zeros(M,N,O); XG=XR;  XB=XR;
    XR(:,:,1)=x(:,:,1); XG(:,:,2)=x(:,:,2);   XB(:,:,3)=x(:,:,3);
    R=get(handles.SliderYMR,'Value')+.1;      G=get(handles.SliderXMR,'Value')+.1; B=get(handles.SliderZMR,'Value')+.1;
    XR=XR.*R; XG=XG.*G; XB=XB.*B; set(handles.ValueYMR,'String',num2str(R));set(handles.ValueXMR,'String',num2str(G));
    set(handles.ValueZMR,'String',num2str(B)); Y(:,:,1)=XR(:,:,1);  Y(:,:,2)=XG(:,:,2);  Y(:,:,3)=XB(:,:,3);
    axes(handles.axes1); imshow(Y); xi=Y;      assignin('base','xi',xi);
```

Figura 19: Código - Aplicar Resolución

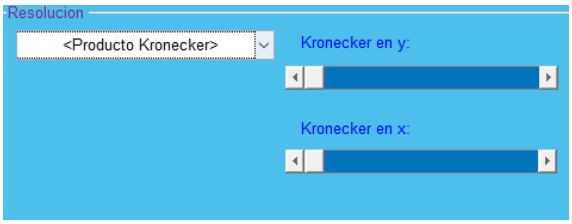
```
            case 11  %Resolucion Radiométrica
                k= round(8*get(handles.SliderXMR,'Value')); n=2^k;  Xrad=round(n*x)/n;
                set(handles.ValueXMR,'String',num2str(k)); axes(handles.axes1); imshow(Xrad); xi=Xrad; assignin('base','xi',xi);
        end
end
set(handles.AplicarMR, 'Value', 0);    set(handles.AplicarMR, 'BackgroundColor', 'white');
set(handles.MenuBarraMR, 'Enable', 'on');  set(handles.AplicarMR, 'Enable', 'on');
set(handles.SliderXMR, 'Enable', 'on');    set(handles.SliderYMR, 'Enable', 'on');
[M,N,O]=size(xi);  str1 = [num2str(N)];str2 = [num2str(M)];
str3=[num2str(O)];str1=strcat(str1,'x'); str2=strcat(str2,'x');
str=strcat(str1,str2); str=strcat(str,str3); set(handles.Dimentions,'String',str);
```

Figura 20: Código - Aplicar Resolución

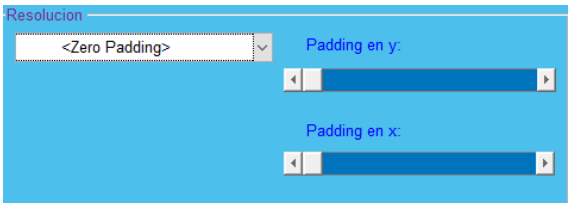
- Cabe mencionar que el slider que corresponde a la función **'submuestreo'** es el más preciso y único que tiene esta característica ya que se requiere que por cada movimiento del slider sea una muestra más o menos.
- En seguida se mostrará el diseño gráfico que tiene cada opción del menú **'Resolución'**.



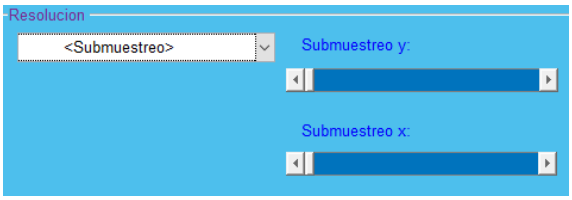
(a) Resolución Espacial - Escalado Matriz Transformación



(b) Resolución Espacial - Producto Kronecker

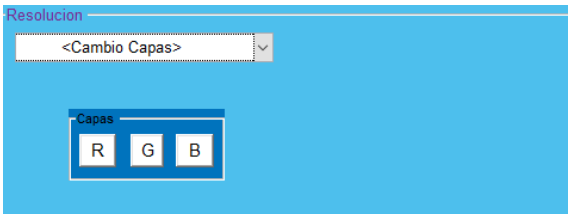


(c) Resolución Espacial - Zero Padding



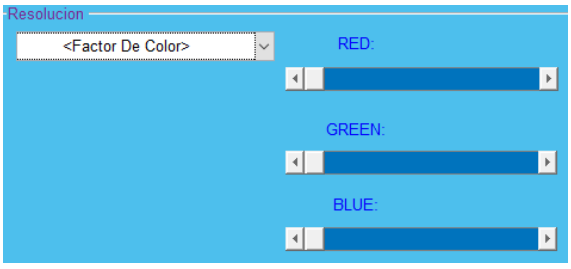
(d) Resolución Espacial - Submuestreo

Figura 21: Interfaz - Resolución Espacial

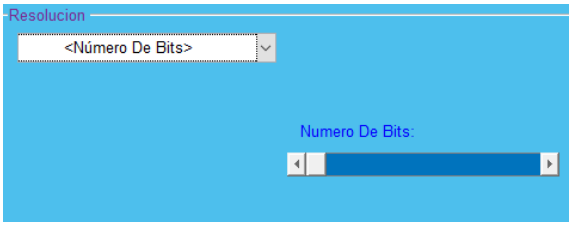


(a) Resolución Espectral - Cambio De Capas

(b) Resolución Espectral - Discrimación De Capas



(c) Resolución Espectral - Factor De Color



(d) Resolución Radiométrica - Número De Bits

Figura 22: Interfaz - Resolución Espectral y Radiométrica

- El Código de ordenamiento de las capas consiste en la verificación de los demás botones que se presentan a traves de variables globales tipo booleanas y así realizar un ordenamiento desde la primera capa hasta la tercera.

```
% --- Executes on button press in pushbuttonR.
function pushbuttonR_Callback(hObject, eventdata, handles)
x=evalin('base','x'); xi=evalin('base','xi'); [M,N,O]=size(x); XR=zeros(M,N,O); XR(:,:,1)=x(:,:,1);
RedButton=evalin('base','RedButton'); GreenButton=evalin('base','GreenButton');  BlueButton=evalin('base','BlueButton');
if GreenButton==1&&BlueButton==1
    xi(:,:,3)=XR(:,:,1);
elseif BlueButton==1||GreenButton==1
    xi(:,:,2)=XR(:,:,1);
else
    xi(:,:,1)=XR(:,:,1);
end
set(handles.AplicarMR,'Enable','On'); set(handles.pushbuttonR,'BackgroundColor','Red');
set(handles.pushbuttonR,'Enable','Off'); RedButton=get(handles.pushbuttonR,'Value');
assignin('base','RedButton',RedButton); assignin('base','xi',xi);
if BlueButton==1&&RedButton==1&&GreenButton==1
    set(handles.AplicarMR,'Visible','On');
end
```

Figura 23: Código - Cambiar Capas: Capa Roja

```
% --- Executes on button press in pushbuttonG.
function pushbuttonG_Callback(hObject, eventdata, handles) %GreenButton
x=evalin('base','x'); xi=evalin('base','xi');[M,N,O]=size(x); XG=zeros(M,N,O); XG(:,:,2)=x(:,:,2);
RedButton=evalin('base','RedButton'); GreenButton=evalin('base','GreenButton');  BlueButton=evalin('base','BlueButton');
if RedButton==1&&BlueButton==1
    xi(:,:,3)=XG(:,:,2);
elseif RedButton==1||BlueButton==1
    xi(:,:,2)=XG(:,:,2);
else
    xi(:,:,1)=XG(:,:,2);
end
set(handles.AplicarMR,'Enable','On'); set(handles.pushbuttonG,'BackgroundColor','Green');
set(handles.pushbuttonG,'Enable','Off'); GreenButton=get(handles.pushbuttonG,'Value');
assignin('base','GreenButton',GreenButton); assignin('base','xi',xi);
if BlueButton==1&&RedButton==1&&GreenButton==1
    set(handles.AplicarMR,'Visible','On');
end
```

Figura 24: Código - Cambiar Capas: Capa Verde

```
% --- Executes on button press in pushbuttonB.
function pushbuttonB_Callback(hObject, eventdata, handles)
x=evalin('base','x'); xi=evalin('base','xi');[M,N,O]=size(x); XB=zeros(M,N,O); XB(:,:,3)=x(:,:,3);
RedButton=evalin('base','RedButton'); GreenButton=evalin('base','GreenButton');  BlueButton=evalin('base','BlueButton');
if RedButton==1&&GreenButton==1
    xi(:,:,3)=XB(:,:,3);
elseif RedButton==1||GreenButton==1
    xi(:,:,2)=XB(:,:,3);
else
    xi(:,:,1)=XB(:,:,3);
end
set(handles.AplicarMR,'Enable','On'); set(handles.pushbuttonB,'BackgroundColor','Blue');
set(handles.pushbuttonB,'Enable','Off');BlueButton=get(handles.pushbuttonB,'Value');
assignin('base','BlueButton',BlueButton); assignin('base','xi',xi);
if BlueButton==1&&RedButton==1&&GreenButton==1
    set(handles.AplicarMR,'Visible','On');
end
```

Figura 25: Código - Cambiar Capas: Capa Azul

- Por otro lado, la discrimación se realiza sólo verificando el estado de las tres casillas activables:

```
% --- Executes on button press in checkboxB.
function checkboxB_Callback(hObject, eventdata, handles) %Blue Color
set(handles.AplicarMR, 'Visible', 'On'); set(handles.AplicarMR, 'Enable', 'on');
x=evalin('base','x');      [M,N,O]=size(x); XB=zeros(M,N,O); B=XB;
if (get(handles.checkboxB,'Value'))==1
    XB(:,:,3)=x(:,:,3); assignin('base','XB',XB);
else
    XB=B; assignin('base','XB',XB);
end

% --- Executes on button press in checkboxG.
function checkboxG_Callback(hObject, eventdata, handles) % Green Color
set(handles.AplicarMR, 'Visible', 'On'); set(handles.AplicarMR, 'Enable', 'on');
x=evalin('base','x');      [M,N,O]=size(x); XG=zeros(M,N,O); G=XG;
if (get(handles.checkboxG,'Value'))==1
    XG(:,:,2)=x(:,:,2); assignin('base','XG',XG); ;
else
    XG=G; assignin('base','XG',XG);
end

% --- Executes on button press in checkboxR.
function checkboxR_Callback(hObject, eventdata, handles)  % Red Color
set(handles.AplicarMR, 'Visible', 'On'); set(handles.AplicarMR, 'Enable', 'on');
x=evalin('base','x');      [M,N,O]=size(x); XR=zeros(M,N,O); R=XR;
if (get(handles.checkboxR,'Value'))==1
    XR(:,:,1)=x(:,:,1); assignin('base','XR',XR);
else
    XR=zeros(M,N,O); assignin('base','XR',XR);
end
```

Figura 26: Código - Discrimación De Capas

3.0.3. Filtros Espaciales

Los **'Filtros Espaciales'** realizan el proceso a traves del dominio espacial mediante la convoluciòn de la imagen y los nùcleos (Kernels) del filtro. Existen de muchos tipos y formas, pero sòlo trabajeros algunos filtros pasa altas y pasa bajas.

- **Pasa Altas** - Permite la detección de bordes con la opción de agregar un porcetanje de los bordes (realzado) a la imagen.

  Laplace 9 Puntos - Permite la detección de bordes a través de la convolución de la imagen y el Kernel Laplaciano de 9 puntos (H=[-1 -1 -1 ; -1 8 -1 ; -1 -1 -1]).

  Laplace 5 Puntos Laterales - Permite la detección de bordes a través de la convolución de la imagen y el Kernel Laplaciano de 5 puntos laterales (H=[0 -1 0 ; -1 4 -1 ; 0 -1 0]).

  Laplace 5 Puntos Esquinados - Permite la detección de bordes a través de la convolución de la imagen y el Kernel Laplaciano de 5 puntos esquinados (H=[-1 0 -1 ; 0 4 0 ; -1 0 -1]).

- **Sobel** - Permite la filtraciòn por medio los nùcleos tipo 'Sobel'.

  Inferior - Permite la detección 'inferior' de bordes a través de la convolución de la imagen y el Kernel Sobel (H= -[-1 -2 -1 ; 0 0 0 ; 1 2 1]).

  Superior - Permite la detección 'Superior' de bordes a través de la convolución de la imagen y el Kernel Sobel (H=[-1 -2 -1 ; 0 0 0 ; 1 2 1]).

  Lateral Derecha - Permite la detección 'lateral derecha' de bordes a través de la convolución de la imagen y el Kernel Sobel (H= -[-1 0 1 ; -2 0 2 ; -1 0 1]).

  Lateral Izquierda - Permite la detección 'lateral izquierda' de bordes a través de la convolución de la imagen y el Kernel Sobel (H=[-1 0 1 ; -2 0 2 ; -1 0 1]).

- **Prewitt** - Permite la filtraciòn por medio los nùcleos tipo 'Prewitt'.

  Horizontal - Permite la detección 'Horizontal' de bordes a través de la convolución de la imagen y el Kernel Prewitt (H=[-1 0 1 ; -1 0 1 ; -1 0 1]).

  Vertical - Permite la detección 'Vertical' de bordes a través de la convolución de la imagen y el Kernel Prewitt (H=[-1 -1 -1 ; 0 0 0 ; 1 1 1]).

  +45 Grados - Permite la detección de bordes a '+45 Grados' a través de la convolución de la imagen y el Kernel Prewitt (H=[1 1 0; 1 0 -1; 0 -1 -1]).

-45 Grados - Permite la detección de bordes a '-45 Grados' a través de la convolución de la imagen y el Kernel Prewitt (H=[0 1 1; -1 0 1; -1 -1 0]).

- **Pasa Bajas** - Permite la eliminaciòn de ruido (suavizado) con la opción de agregar un porcetanje de ruido (interferencia) a la imagen.

  Dimensión Impar - Permite la eliminaciòn de ruido o suavizado de los bordes mediante la convoluciòn de la imagen y el Kernel 'Box Blur' (H=$(1/n^2)$*ones(n)). Teniendo la posibilidad de modificar el grado del filtro (n) para tener una mayor o menor eliminaciòn de ruido. El grado siempre es de tipo impar (1,3,5,7...,51).

- Para continuar, se mostrará el código y el diseño gráfico que toma el menú de los 'Filtros Espaciales' el cual es muy similar en cada uno de los filtros pasa altas ya que solamente se requiere agregar bordes, en cambio en el filtro pasa bajas se requiere modular el ruido y además el grado del filtro. El menú se encuentra en la función 'MenuBarraF' donde además contiene las modificaciones del menú de los 'Filtros Frecuenciales'.

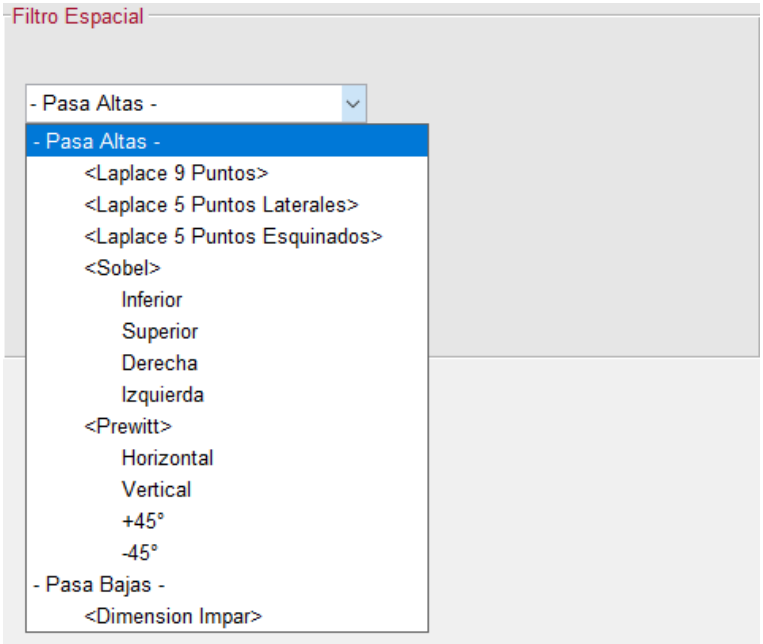


Figura 27: Interfaz - Menú Filtros Espaciales

```
function MenuBarraF_Callback(hObject, eventdata, handles)
Bopc=evalin('base', 'Bopc'); opc=get(handles.MenuBarraF,'Value');
set(handles.TextYF,'Visible', 'Off');    set(handles.TextXF,'Visible', 'Off');
set(handles.ValueYF,'Visible', 'Off');   set(handles.ValueXF,'Visible', 'Off');
set(handles.SliderYF,'Visible', 'Off');  set(handles.SliderXF,'Visible', 'Off');   set(handles.SliderZF,'Visible', 'Off');
set(handles.TextYF,'String', '');set(handles.TextXF,'String', ''); set(handles.TextZF,'String', '');
set(handles.ValueYF,'String', '');set(handles.ValueXF,'String', ''); set(handles.ValueZF,'String', '');
set(handles.AplicarF,'Visible', 'On');   set(handles.SliderXF,'Value', 0);         set(handles.SliderYF,'Value', 0);

switch Bopc
    case 1  %Filtros Espaciales
        switch opc  %Pasa Altas
            case 2  %Laplace 9 Puntos
                set(handles.TextXF,'Visible', 'On');   set(handles.ValueXF,'Visible', 'On'); set(handles.SliderXF,'Visible', 'On');
                set(handles.SliderXF,'Visible', 'On'); set(handles.TextXF,'String', 'Factor De Bordes:');
            case 3  %Laplace 5 Puntos Laterales
                set(handles.TextXF,'Visible', 'On');   set(handles.ValueXF,'Visible', 'On'); set(handles.SliderXF,'Visible', 'On');
                set(handles.SliderXF,'Visible', 'On'); set(handles.TextXF,'String', 'Factor De Bordes:');
            case 4  %Laplace 5 Puntos Esquinados Laterales
                set(handles.TextXF,'Visible', 'On');   set(handles.ValueXF,'Visible', 'On'); set(handles.SliderXF,'Visible', 'On');
                set(handles.SliderXF,'Visible', 'On'); set(handles.TextXF,'String', 'Factor De Bordes:');
            case 6  %Sobel - Inferior
                set(handles.TextXF,'Visible', 'On');   set(handles.ValueXF,'Visible', 'On'); set(handles.SliderXF,'Visible', 'On');
                set(handles.SliderXF,'Visible', 'On'); set(handles.TextXF,'String', 'Factor De Bordes:');
            case 7  %Sobel - Superior
                set(handles.TextXF,'Visible', 'On');   set(handles.ValueXF,'Visible', 'On'); set(handles.SliderXF,'Visible', 'On');
                set(handles.SliderXF,'Visible', 'On'); set(handles.TextXF,'String', 'Factor De Bordes:');
```

Figura 28: Código - MenuBarraF: Filtros Espaciales

```
            case 8  %Sobel - Derecha
                set(handles.TextXF,'Visible', 'On');   set(handles.ValueXF,'Visible', 'On'); set(handles.SliderXF,'Visible', 'On');
                set(handles.SliderXF,'Visible', 'On'); set(handles.TextXF,'String', 'Factor De Bordes:');
            case 9  %Sobel - Izquierda
                set(handles.TextXF,'Visible', 'On');   set(handles.ValueXF,'Visible', 'On'); set(handles.SliderXF,'Visible', 'On');
                set(handles.SliderXF,'Visible', 'On'); set(handles.TextXF,'String', 'Factor De Bordes:');
            case 11  %Prewitt - Horizontal
                set(handles.TextXF,'Visible', 'On');   set(handles.ValueXF,'Visible', 'On'); set(handles.SliderXF,'Visible', 'On');
                set(handles.SliderXF,'Visible', 'On'); set(handles.TextXF,'String', 'Factor De Bordes:');
            case 12  %Prewitt - Vertical
                set(handles.TextXF,'Visible', 'On');   set(handles.ValueXF,'Visible', 'On'); set(handles.SliderXF,'Visible', 'On');
                set(handles.SliderXF,'Visible', 'On'); set(handles.TextXF,'String', 'Factor De Bordes:');
            case 13  %Prewitt -  +45°
                set(handles.TextXF,'Visible', 'On');   set(handles.ValueXF,'Visible', 'On'); set(handles.SliderXF,'Visible', 'On');
                set(handles.SliderXF,'Visible', 'On'); set(handles.TextXF,'String', 'Factor De Bordes:');
            case 14  %Prewitt -  -45°
                set(handles.TextXF,'Visible', 'On');   set(handles.ValueXF,'Visible', 'On'); set(handles.SliderXF,'Visible', 'On');
                set(handles.SliderXF,'Visible', 'On'); set(handles.TextXF,'String', 'Factor De Bordes:');
            case 16  %Dimension Impar - Pasa Bajas
                set(handles.TextXF,'Visible', 'On');     set(handles.ValueXF,'Visible', 'On');
                set(handles.TextYF,'Visible', 'On');     set(handles.ValueYF,'Visible', 'On');
                set(handles.SliderXF,'Visible', 'On');   set(handles.SliderYF,'Visible', 'On');
                set(handles.TextXF,'String', 'Factor De Ruido:');  set(handles.TextYF,'String', 'Grado Del Filtro:');
        end
```

Figura 29: Código - MenuBarraF: Filtros Espaciales

- Dichos cambios se encuentran en la función **'AplicarF'** la cual se programó de la siguiente manera:

```
function AplicarF_Callback(hObject, eventdata, handles)
Bopc=evalin('base','Bopc'); opc=get(handles.MenuBarraF,'Value');
x=evalin('base','x');  [M,N,O]=size(x); xi=evalin('base','xi');
Factor=get(handles.SliderXF,'Value');
set(handles.AplicarF, 'Enable', 'off'); set(handles.AplicarF, 'BackgroundColor', 'red'); pause(.5);

switch Bopc
    case 1  %Filtros Espaciales
        switch opc
            case 2  %Laplace 9 Puntos
                H=Kernel('High','Laplacian','Diagonals');  x=x*(Factor+.01);  set(handles.ValueXF,'String',num2str(Factor+.01));
                Y=convn(x,H,'same');  axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi);

            case 3  %Laplace 5 Puntos Laterales
                H=Kernel('High','Laplacian','Cross'); x=x*(Factor+.01);  Y=convn(x,H,'same');
                axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi); set(handles.ValueXF,'String',num2str(Factor+.01));

            case 4  %Laplace 5 Puntos Esquinados
                H=Kernel('High','Laplacian','Corners'); x=x*(Factor+.01);  Y=convn(x,H,'same');
                axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi); set(handles.ValueXF,'String',num2str(Factor+.01));

            case 6  %Sobel - Inferior
                H=-1*(Kernel('High','Sobel','Horizontal')); x=x*(Factor+.01);  Y=convn(x,H,'same');
                axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi); set(handles.ValueXF,'String',num2str(Factor+.01));

            case 7  %Sobel - Superior
                H=Kernel('High','Sobel','Horizontal'); x=x*(Factor+.01); Y=convn(x,H,'same');
                axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi); set(handles.ValueXF,'String',num2str(Factor+1.01));
```

Figura 30: Código - AplicarF: Filtros Espaciales

```
case 8  %Sobel - Derecha
    H=-1*(Kernel('High','Sobel','Vertical')); x=x*(Factor+.01); Y=convn(x,H,'same');
    axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi); set(handles.ValueXF,'String',num2str(Factor+.01));

case 9  %Sobel - Izquierda
    H=Kernel('High','Sobel','Vertical'); x=x*(Factor+.01); Y=convn(x,H,'same');
    axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi); set(handles.ValueXF,'String',num2str(Factor+.01));

case 11  %Prewitt - Horizontal
    H=Kernel('High','Prewitt','Horizontal'); x=x*(Factor+.01); Y=convn(x,H,'same');
    axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi); set(handles.ValueXF,'String',num2str(Factor+.01));

case 12  %Prewitt - Vertical
    H=Kernel('High','Prewitt','Vertical'); x=x*(Factor+.01); Y=convn(x,H,'same');
    axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi); set(handles.ValueXF,'String',num2str(Factor+.01));

case 13  %Prewitt -  +45°
    H=Kernel('High','Prewitt','+45'); x=x*(Factor+.01); Y=convn(x,H,'same');
    axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi); set(handles.ValueXF,'String',num2str(Factor+.01));

case 14  %Prewitt -  -45°
    H=Kernel('High','Prewitt','-45'); x=x*(Factor+.01);  Y=convn(x,H,'same');
    axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi); set(handles.ValueXF,'String',num2str(Factor+.01));

case 16  %Pasa Bajas-Dimension Impar
    n=2*(round(25*get(handles.SliderYF,'Value'))+1)-1;  H=Kernel('Low','Box','',n); x=x+Factor*randn(M,N,O);
    Y=convn(x,H,'same');  axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi);
    set(handles.ValueYF,'String',num2str(n));  set(handles.ValueXF,'String',num2str(Factor));
```

Figura 31: Código - AplicarF: Filtros Espaciales

- La interfaz gráfica de los filtros pasa altas se vería así ya que sólo podemos agregar más bordes a la imagen:

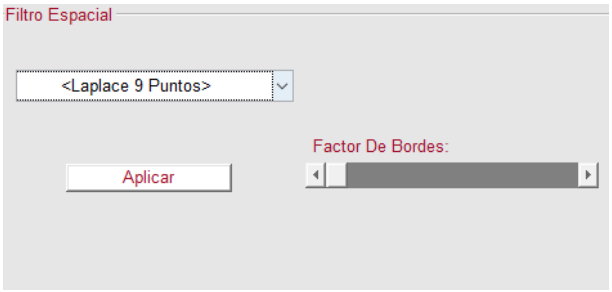


Figura 32: Interfaz - Filtros Espaciales: Laplace 9 Puntos (Pasa Altas)

- Por otro lado, la interfaz gráfica de los filtros pasa bajas se vería así ya que tenemos un valor más que modificar aplicado directamente al núcleo:

Figura 33: Interfaz - Filtros Espaciales: Dimensión Impar (Pasa Bajas)

3.0.4. Filtros Frecuenciales

Los **'Filtro Frecuenciales'** se caracterizan por realizar el filtrado en el dominio frecuencial a través la traformada rápida de Fourier, una vez que se ha realizado el filtrado entonces se regresa al dominio espacial mediante la transformada inversa de Fourier y así se muestra la imagen filtrada. En el caso de los fltros 'pasa altas' se calcula la inversa (negativo) del filtro para poder realizar la detección de bordes. Una vez realizada la transformada rápida de Fourier, mediante dos ciclos for anidados uno de 1 a M y otro de 1 a N, se realiza la filtración de las frecuencias deseadas.
Siendo M y N las medidas de la imagen en 'x' y 'y' se calculan entonces los diferenciales 'd_x' y 'd_y' para después calcular el diferencial 'd_{xy}' para lograr centralizar el filtro y realizarlo.
Siendo sigma el radio del filtro el cuál es posible variar con cualquier dezlizador en la interfaz el usuario consigue modificar la ventana de cada filtro.

$$d_x = \frac{x - \frac{M}{2}}{\frac{M}{2}} \tag{1}$$

$$d_y = \frac{y - \frac{N}{2}}{\frac{N}{2}} \tag{2}$$

$$d_{xy} = \sqrt{d_x^2 + d_y^2} \tag{3}$$

- **Pasa Altas** - Permite realizar una detección de bordes mediante la creación y caracterización del filtro el cual es aplicado a la frecuencia de la imagen, una vez hecho el filtro de las frecuencias entonces se aplica el negativo (1-W) y se multiplica por la transformada rápida de Fourier aplicada a la imagen para así lograr un filtrado de bordes.

  Rectangular - Mediante la condición $(x-a)^2+(y-b)^2 > \sigma^2$ se logra filtrar aquellas frecuencias las cuales estén dentro del radio de la circunferencia por lo que $W(x,y) = 1$.

  Gaussiano - Mediante la ventana $W(x,y) = e^{-\frac{d_{x,y}^2}{2\sigma^2}}$, se filtran las frecuencias con respecto al d_{xy}.

  Gaussiano Modificado - Mediante la ventana $W(x,y) = e^{-\frac{d_{x,y}^4}{2\sigma^2}}$, se filtran las frecuencias con respecto al d_{xy}.

  Coseno - Mediante la condición $d_x^2 || d_y^2 <= \sigma^2$ se filtran las frecuencias para $W(x,y) = \cos(\frac{\pi d_x}{2\sigma}) * \cos(\frac{\pi d_y}{2\sigma})$.

  Hanning - Mediante la condición $(d_x^2 + d_y^2) <= \sigma^2$ se filtran las frecuencias para $W(x,y) = (.5 * \cos(\pi dxy/sigma) + .5)$.

  Bartlet - Mediante la condicionante $(\frac{1-d_{xy}}{\sigma}) > 0$ se filtran las señales para $W(x,y) = (\frac{1-d_{xy}}{\sigma})$.

- **Pasa Bajas** - Permite realizar una supresión de ruido mediante el diseño y caracterización del filtro el cual es aplicado a la frecuencia de la imagen.

Rectangular - Mediante la condición $(x-a)^2+(y-b)^2 > \sigma^2$ se logra filtrar aquellas frecuencias las cuales estén dentro del radio de la circunferencia por lo que $W(x,y) = 1$.

Gaussiano - Mediante la ventana $W(x,y) = e^{-\frac{d^2_{x,y}}{2\sigma^2}}$, se filtran las frecuencias con respecto al d_{xy}.

Gaussiano Modificado - Mediante la ventana $W(x,y) = e^{-\frac{d^4_{x,y}}{2\sigma^2}}$, se filtran las frecuencias con respecto al d_{xy}.

Coseno - Mediante la condición $d_x^2||d_y^2 <= \sigma^2$ se filtran las frecuencias para $W(x,y) = \cos(\frac{\pi d_x}{2\sigma}) * \cos(\frac{\pi d_y}{2\sigma})$.

Hanning - Mediante la condición $(d_x^2 + d_y^2) <= \sigma^2$ se filtran las frecuencias para $W(x,y) = (.5 * \cos(\pi dxy/sigma) + .5)$.

Bartlet - Mediante la condicionante $(\frac{1-d_{xy}}{\sigma}) > 0$ se filtran las señales para $W(x,y) = (\frac{1-d_{xy}}{\sigma})$.

- Más adelante se muestra el menú de la sección 'Filtros Frecuenciales' así como su respectiva codificación la cual se encuentra en la función 'MenuBarraF' donde también se encuentra el menú para los filtros espaciales:

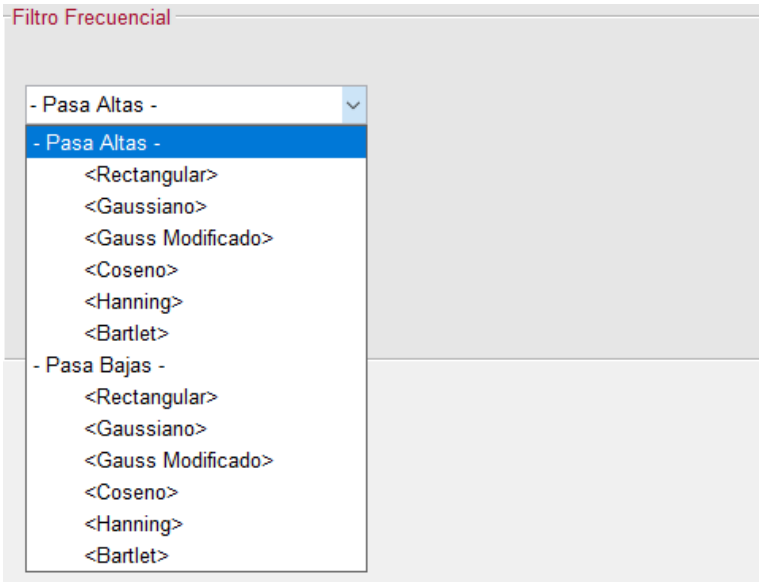


Figura 34: Interfaz - Menú Filtros Frecuenciales

```
case 2  %Filtros Frecuenciales
    switch opc
        %  Pasa Altas
        case 2  %Rectangular Gauss
            set(handles.TextXF,'Visible', 'On');              set(handles.ValueXF,'Visible', 'On');
            set(handles.TextYF,'Visible', 'On');              set(handles.ValueYF,'Visible', 'On');
            set(handles.SliderXF,'Visible', 'On');            set(handles.SliderYF,'Visible', 'On');
            set(handles.TextXF,'String', 'Factor De Bordes:'); set(handles.TextYF,'String', 'Radio Del Filtro:');
        case 3  %Gaussiano
            set(handles.TextXF,'Visible', 'On');              set(handles.ValueXF,'Visible', 'On');
            set(handles.TextYF,'Visible', 'On');              set(handles.ValueYF,'Visible', 'On');
            set(handles.SliderXF,'Visible', 'On');            set(handles.SliderYF,'Visible', 'On');
            set(handles.TextXF,'String', 'Factor De Bordes:'); set(handles.TextYF,'String', 'Radio Del Filtro:');
        case 4  %Gauss Modificado
            set(handles.TextXF,'Visible', 'On');              set(handles.ValueXF,'Visible', 'On');
            set(handles.TextYF,'Visible', 'On');              set(handles.ValueYF,'Visible', 'On');
            set(handles.SliderXF,'Visible', 'On');            set(handles.SliderYF,'Visible', 'On');
            set(handles.TextXF,'String', 'Factor De Bordes:'); set(handles.TextYF,'String', 'Radio Del Filtro:');
        case 5  %Coseno
            set(handles.TextXF,'Visible', 'On');              set(handles.ValueXF,'Visible', 'On');
            set(handles.TextYF,'Visible', 'On');              set(handles.ValueYF,'Visible', 'On');
            set(handles.SliderXF,'Visible', 'On');            set(handles.SliderYF,'Visible', 'On');
            set(handles.TextXF,'String', 'Factor De Bordes:'); set(handles.TextYF,'String', 'Radio Del Filtro:');
        case 6  %Hanning
            set(handles.TextXF,'Visible', 'On');              set(handles.ValueXF,'Visible', 'On');
            set(handles.TextYF,'Visible', 'On');              set(handles.ValueYF,'Visible', 'On');
            set(handles.SliderXF,'Visible', 'On');            set(handles.SliderYF,'Visible', 'On');
            set(handles.TextXF,'String', 'Factor De Bordes:'); set(handles.TextYF,'String', 'Radio Del Filtro:');
```

Figura 35: Código - MenuBarraF: Filtros Frecuenciales

```
        case 7  %Barlet
            set(handles.TextXF,'Visible', 'On');              set(handles.ValueXF,'Visible', 'On');
            set(handles.TextYF,'Visible', 'On');              set(handles.ValueYF,'Visible', 'On');
            set(handles.SliderXF,'Visible', 'On');            set(handles.SliderYF,'Visible', 'On');
            set(handles.TextXF,'String', 'Factor De Bordes:'); set(handles.TextYF,'String', 'Radio Del Filtro:');

        %  Pasa Bajas
        case 9  %Rectangular
            set(handles.TextXF,'Visible', 'On');               set(handles.ValueXF,'Visible', 'On');
            set(handles.TextYF,'Visible', 'On');               set(handles.ValueYF,'Visible', 'On');
            set(handles.SliderXF,'Visible', 'On');             set(handles.SliderYF,'Visible', 'On');
            set(handles.TextXF,'String', 'Factor De Ruido:');  set(handles.TextYF,'String', 'Radio Del Filtro:');
        case 10  %Gaussiano
            set(handles.TextXF,'Visible', 'On');               set(handles.ValueXF,'Visible', 'On');
            set(handles.TextYF,'Visible', 'On');               set(handles.ValueYF,'Visible', 'On');
            set(handles.SliderXF,'Visible', 'On');             set(handles.SliderYF,'Visible', 'On');
            set(handles.TextXF,'String', 'Factor De Ruido:');  set(handles.TextYF,'String', 'Radio Del Filtro:');
        case 11  %Gaussiano Modificado
            set(handles.TextXF,'Visible', 'On');               set(handles.ValueXF,'Visible', 'On');
            set(handles.TextYF,'Visible', 'On');               set(handles.ValueYF,'Visible', 'On');
            set(handles.SliderXF,'Visible', 'On');             set(handles.SliderYF,'Visible', 'On');
            set(handles.TextXF,'String', 'Factor De Ruido:');  set(handles.TextYF,'String', 'Radio Del Filtro:');
        case 12  %Coseno
            set(handles.TextXF,'Visible', 'On');               set(handles.ValueXF,'Visible', 'On');
            set(handles.TextYF,'Visible', 'On');               set(handles.ValueYF,'Visible', 'On');
            set(handles.SliderXF,'Visible', 'On');             set(handles.SliderYF,'Visible', 'On');
            set(handles.TextXF,'String', 'Factor De Ruido:');  set(handles.TextYF,'String', 'Radio Del Filtro:');
```

Figura 36: Código - MenuBarraF: Filtros Frecuenciales

```
            case 13  %Hanning
                set(handles.TextXF,'Visible', 'On');                set(handles.ValueXF,'Visible', 'On');
                set(handles.TextYF,'Visible', 'On');                set(handles.ValueYF,'Visible', 'On');
                set(handles.SliderXF,'Visible', 'On');              set(handles.SliderYF,'Visible', 'On');
                set(handles.TextXF,'String', 'Factor De Ruido:');   set(handles.TextYF,'String', 'Radio Del Filtro:');
            case 14  %Barlet
                set(handles.TextXF,'Visible', 'On');                set(handles.ValueXF,'Visible', 'On');
                set(handles.TextYF,'Visible', 'On');                set(handles.ValueYF,'Visible', 'On');
                set(handles.SliderXF,'Visible', 'On');              set(handles.SliderYF,'Visible', 'On');
                set(handles.TextXF,'String', 'Factor De Ruido:');   set(handles.TextYF,'String', 'Radio Del Filtro:');
        end
end
```

Figura 37: Código - MenuBarraF: Filtros Frecuenciales

- La parte donde se aplican los filtros frecuenciales dentro del código se encuentra en la función **'AplicarF'** que a su vez contiene las aplicaciones de los 'filtros espaciales'.

```
    end

case 2  %Filtros Frecuenciales
    x=Gray(x); [M,N,O]=size(x); a=M/2; b=N/2; H=zeros(M,N);
    sigma=get(handles.SliderYF,'Value')+.1;

    switch opc  %Pasa ALtas
        case 2  %Rectangular Gauss
            sigma=round(149*get(handles.SliderYF,'Value')+1); x=x*(2*Factor+.1); Xf=fftshift(fft2(x));
            for x1=1:M
                dx=(x1-a)/a;
                for y=1:N
                    dy=(y-b)/b;
                    if (x1-a)^2+(y-b)^2>sigma^2
                        H(x1,y)=1;                          %Filtro Ideal
                    end
                end
            end
            Yf=Xf.*(1-H); Y=ifft2(ifftshift(Yf));
            axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi);
            set(handles.ValueYF,'String',num2str(sigma));  set(handles.ValueXF,'String',num2str(Factor+.1));
```

Figura 38: Código - AplicarF: Filtros Frecuenciales

```
case 3  %Gaussiano
    x=x*(2*Factor+.1); Xf=fftshift(fft2(x));
    for x1=1:M
        dx=(x1-a)/a;
        for y=1:N
            dy=(y-b)/b;
            dxy=sqrt(dx^2+dy^2);
            H(x1,y)=exp((-dxy^4)/(2*sigma^2));
        end
    end
    Yf=Xf.*(1-H);
    Y=ifft2(ifftshift(Yf));
    axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi);
    set(handles.ValueYF,'String',num2str(sigma));  set(handles.ValueXF,'String',num2str(Factor+.1));

case 4  %Gauss Modificado
    x=x*(2*Factor+.1); Xf=fftshift(fft2(x));
    for x1=1:M
        dx=(x1-a)/a;
        for y=1:N
            dy=(y-b)/b;
            dxy=sqrt(dx^2+dy^2);
            H(x1,y)=exp((-dxy^4)/(2*sigma^2));
        end
    end
    Yf=Xf.*(1-H);
    Y=ifft2(ifftshift(Yf));
    axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi);
    set(handles.ValueYF,'String',num2str(sigma));  set(handles.ValueXF,'String',num2str(Factor+.1));
```

Figura 39: Código - AplicarF: Filtros Frecuenciales

```
case 5  %Coseno
    x=x*(2*Factor+.1); Xf=fftshift(fft2(x));
    for x1=1:M
        dx=(x1-a)/a;
        for y=1:N
            dy=(y-b)/b;
            if dx^2||dy^2<=sigma^2
                H(x1,y)=cos(pi*dx/(2*sigma))*cos(pi*dy/(2*sigma));    %Coseno
            end
        end
    end
    Yf=Xf.*(1-H);
    Y=ifft2(ifftshift(Yf));
    axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi);
    set(handles.ValueYF,'String',num2str(sigma));  set(handles.ValueXF,'String',num2str(Factor+.1));
case 6  %Hanning
    x=x*(2*Factor+.1); Xf=fftshift(fft2(x));
    for x1=1:M
        dx=(x1-a)/a;
        for y=1:N
            dy=(y-b)/b;
            dxy=sqrt(dx^2+dy^2);
            if (dx^2+dy^2)<=sigma^2
                H(x1,y)=(.5*cos(pi*dxy/sigma)+.5);     %Hanning
            end
        end
    end
    Yf=Xf.*(1-H); Y=ifft2(ifftshift(Yf));
    axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi);
    set(handles.ValueYF,'String',num2str(sigma));  set(handles.ValueXF,'String',num2str(Factor+.1));
```

Figura 40: Código - AplicarF: Filtros Frecuenciales

```
    case 7  %Barlet
        x=x*(2*Factor+.1); Xf=fftshift(fft2(x));
        for x1=1:M
            dx=(x1-a)/a;
            for y=1:N
                dy=(y-b)/b;
                dxy=sqrt(dx^2+dy^2);
                if (1-dxy/sigma)>0
                    H(x1,y)=(1-dxy/sigma);
                end
            end
        end
        Yf=Xf.*(1-H); Y=ifft2(ifftshift(Xf));
        axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi);
        set(handles.ValueYF,'String',num2str(sigma));  set(handles.ValueXF,'String',num2str(Factor+.1));
    %Pasa Bajas
    case 9  %Rectangular
        x=x+(Factor/4)*randn(M,N,0);  Xf=fftshift(fft2(x));
        for x1=1:M
            dx=(x1-a)/a;
            for y=1:N
                dy=(y-b)/b;
                if (x1-a)^2+(y-b)^2>sigma^2
                    H(x1,y)=1;                          %Filtro Ideal
                end
            end
        end
        Yf=Xf.*H; Y=ifft2(ifftshift(Yf));
        axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi);
        set(handles.ValueYF,'String',num2str(sigma));  set(handles.ValueXF,'String',num2str(Factor/4));
```

Figura 41: Código - AplicarF: Filtros Frecuenciales

```
case 10  %Gaussiano
    x=x+(Factor/4)*randn(M,N,0);  Xf=fftshift(fft2(x));
    for x1=1:M
        dx=(x1-a)/a;
        for y=1:N
            dy=(y-b)/b;
            dxy=sqrt(dx^2+dy^2);
            H(x1,y)=exp((-dxy^4)/(2*sigma^2));
        end
    end
    Yf=Xf.*H;
    Y=ifft2(ifftshift(Yf));
    axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi);
    set(handles.ValueYF,'String',num2str(sigma));  set(handles.ValueXF,'String',num2str(Factor/4));

case 11  %Gaussiano Modificado
    x=x+(Factor/4)*randn(M,N,0);  Xf=fftshift(fft2(x));
    for x1=1:M
        dx=(x1-a)/a;
        for y=1:N
            dy=(y-b)/b;
            dxy=sqrt(dx^2+dy^2);
            H(x1,y)=exp((-dxy^4)/(2*sigma^2));
        end
    end
    Yf=Xf.*H; Y=ifft2(ifftshift(Yf));
    axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi);
    set(handles.ValueYF,'String',num2str(sigma));  set(handles.ValueXF,'String',num2str(Factor/4));
```

Figura 42: Código - AplicarF: Filtros Frecuenciales

```
case 12  %Coseno
    x=x+(Factor/4)*randn(M,N,0);  Xf=fftshift(fft2(x));
    for x1=1:M
        dx=(x1-a)/a;
        for y=1:N
            dy=(y-b)/b;
            if dx^2||dy^2<=sigma^2
                H(x1,y)=cos(pi*dx/(2*sigma))*cos(pi*dy/(2*sigma));    %Coseno
            end
        end
    end
    Yf=Xf.*H; Y=ifft2(ifftshift(Yf));
    axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi);
    set(handles.ValueYF,'String',num2str(sigma));  set(handles.ValueXF,'String',num2str(Factor/4));
case 13  %Hanning
    x=x+(Factor/4)*randn(M,N,0);  Xf=fftshift(fft2(x));
    for x1=1:M
        dx=(x1-a)/a;
        for y=1:N
            dy=(y-b)/b;
            dxy=sqrt(dx^2+dy^2);
            if (dx^2+dy^2)<=sigma^2
                H(x1,y)=(.5*cos(pi*dxy/sigma)+.5);     %Hanning
            end
        end
    end
    Yf=Xf.*H; Y=ifft2(ifftshift(Yf));
    axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi);
    set(handles.ValueYF,'String',num2str(sigma));  set(handles.ValueXF,'String',num2str(Factor/4));
```

Figura 43: Código - AplicarF: Filtros Frecuenciales

```
        case 14  %Barlet
            x=x+(Factor/4)*randn(M,N,0);  Xf=fftshift(fft2(x));
            for x1=1:M
                dx=(x1-a)/a;
                for y=1:N
                    dy=(y-b)/b;
                    dxy=sqrt(dx^2+dy^2);
                    if (1-dxy/sigma) > 0
                        H(x1,y)=(1-dxy/sigma);
                    end
                end
            end
            Yf=Xf.*H; Y=ifft2(ifftshift(Yf));
            axes(handles.axes1); imshow(Y); xi=Y; assignin('base','xi',xi);
            set(handles.ValueYF,'String',num2str(sigma));  set(handles.ValueXF,'String',num2str(Factor/4));

    end
end
set(handles.AplicarF, 'Enable', 'on'); set(handles.AplicarF, 'BackgroundColor', 'white');
[M,N,O]=size(xi);  str1=[num2str(N)];     str2 = [num2str(M)];
str3=[num2str(O)]; str1=strcat(str1,'x'); str2=strcat(str2,'x');
str=strcat(str1,str2); str=strcat(str,str3); set(handles.Dimentions,'String',str);
```

Figura 44: Código - AplicarF: Filtros Frecuenciales

- La interfaz que le pertenece a los filtros frecuenciales de forma resumida todos los filtros pasa altas se verían de manera similar como se muestra a continuación:

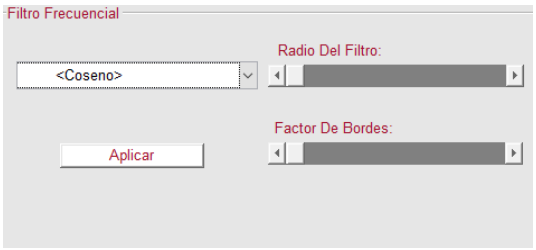


Figura 45: Interfaz - Filtros Frecuenciales: Coseno (Pasa ALtas)

- En cambio los filtros pasa bajas se verían de manera similar:

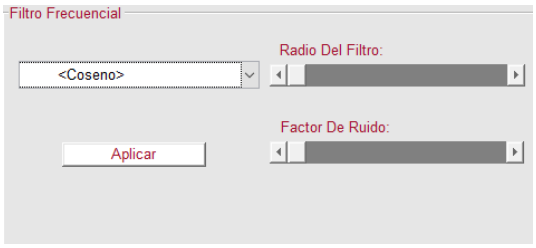


Figura 46: Interfaz - Filtros Frecuenciales: Coseno (Pasa Bajas)

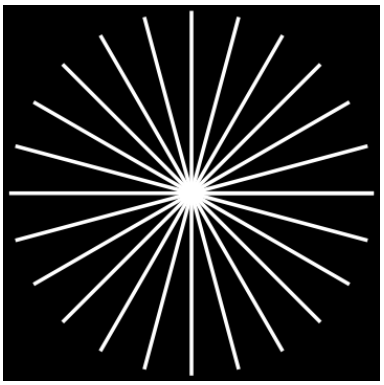
4. Resultados

- En esta sección se mostrarán algunos resultados de cada sección. Los resultados fueron guardados como imagenes directamente con la opción de 'Almacenar imagen automáticamente' que se encuentra en la interfaz.

- **Primer Parte Del Proyecto (video)** - https://youtu.be/plzuL1mBoCw
- **Segunda Parte Del Proyecto (video)** - https://youtu.be/v5SvFIlc894
- A continuación se muestran las imagenes originales con las que se realizaron las pruebas en cada sección:

(a) Imagen Prueba 1

(b) Imagen Prueba 2



(c) Imagen Prueba 3

Figura 47: Imagenes De Prueba

4.1. Movimientos

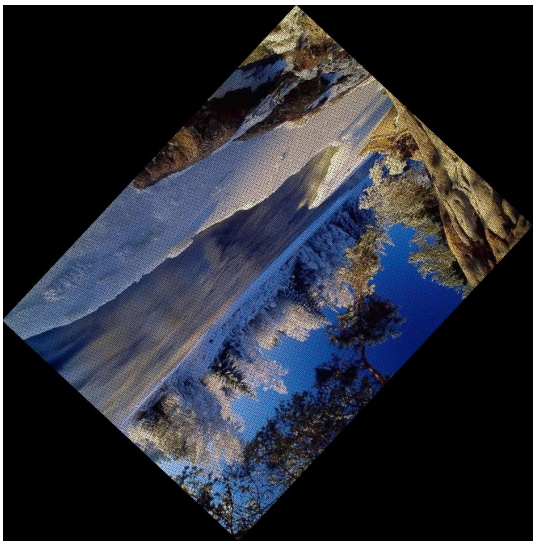


Figura 48: Imagen Con Giro A 139 Grados

4.2. Resolución

Figura 49: Imagen Submuestreada cada 3 pixeles en 'X' y 5 en 'Y'

Figura 50: Imagen Discriminada En Su Capa Verde (Green). Combinación De RB

Figura 51: Imagen Con Resolución Radiométrica De 0 Bits

4.3. Filtro Espacial

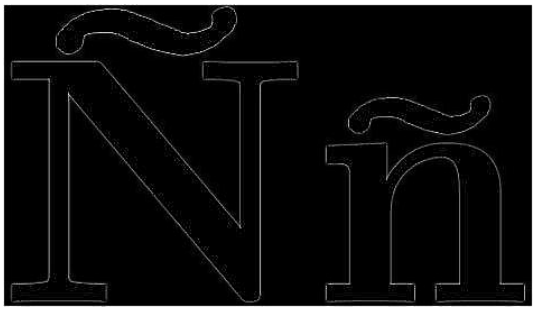


Figura 52: Imagen Filtrada Con El Núcleo Laplaciano 9 puntos

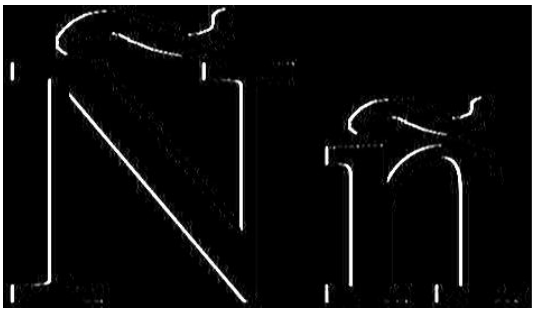


Figura 53: Imagen Filtrada Con El Núcleo Sobel Lateral Izquierda

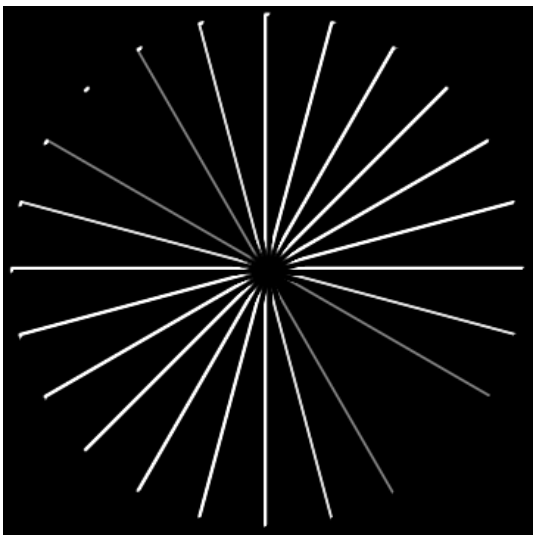


Figura 54: Imagen Filtrada Con El Núcleo Prewitt A +45 Grados

4.4. Filtro Frecuencial

Figura 55: Imagen Filtrada Con La Ventana Rectangular (Pasa Altas)

Figura 56: Imagen Filtrada Con La Ventana Coseno (Pasa Altas)

Figura 57: Imagen Filtrada Con La ventana Hanning (Pasa Altas)

5. Conclusiones

- Haber creado una aplicación la cual incluya en su mayoría todos los algorítmos vistos en la clase hace la comprensión de los mismos y su aplicación más fácil de ver.

- Por otro lado se ha creado una herramienta que puede ser de mucha utilidad cuando se requiere previamente trabajar la imagen para realizar ya sea un análisis de cualquier tipo que se deba realizar a la imagen modificada.

- Trabajar la interfaz en Matlab ha sido un poco complicado por el lado de tratar de realizarla lo más funcional y entendible para el usuario, pero ciertamente ha sido de gran aprendizaje.

6. Bibliografías

[1] R.C. Gonzalez, R.E. Woods, S.L. Eddins, (2004) Digital Image Processing Using Matlab, Prentice Hall, Upper Saddle River, New Jersey.

[2] Dr. Tim Morris, Image Processing with MATLAB, Prentice Hall, Kilburn Building, 26 pages.

[3] R.C. Gonzalez, R.E. Woods, (2008) Digital Image Processing, Third Edition, Prentice Hall, University of Tennessee, 976 pages.