Herramienta de Apoyo para Cursos de Procesamiento Digital de Imagen

J. G. Velásquez-Aguilar, U. Alcázar-Carreño, R. Mañón-Abarca, L. Martínez-Rebollar, G. Ortiz-Ojeda

Centro de Investigación en Ingeniería y Ciencias Aplicadas, Universidad Autónoma del Estado de Morelos,
Cuernavaca, Morelos, 62209, México
{jgpeva, raul_manon, lucio_martinez}@uaem.mx

Resumen. Actualmente existen muchas herramientas prácticas que permiten el procesamiento digital de imágenes las cuales son útiles para diversas áreas tales como: Biomedicina, Fotónica, Procesamiento de Imágenes, etc. Sin embargo todas estás herramientas son especificas a una tarea determinada, lo cual resulta problemático cuando se intenta resolver varias aplicaciones (histogramas, transformadas, filtros, umbralizaciónes, transformaciones geométricas, operaciones morfológicas, etc.) a una imagen. En este artículo se propone una herramienta que permita ilustrar diversas operaciones sobre una imagen. La aplicación consta de una interfaz grafica que facilita su uso, además, no se necesita tener instalado un software específico, lo que permite su instalación fácilmente.

Palabras Clave: Procesamiento de Imagen, MatLab.

1 INTRODUCCIÓN

En los últimos años el Procesamiento Digital de Imágenes ha sido ampliamente utilizado en diversas disciplinas tales como: Medicina, Biología, Física e Ingeniería. Mediante el Procesamiento Digital de Imágenes es posible manipular imágenes digitales en una computadora con el fin de obtener información objetiva. Dos de las tareas fundamentales del procesamiento son: Mejoramiento de una imagen digital con fines interpretativos. Toma de decisiones de manera automática de acuerdo al contenido de la imagen digital.

Diversas aplicaciones hacen uso del análisis de imagen tales como: detección de presencia de objetos, inspección visual automática, medición de características geométricas y de color de objetos, clasificación de objetos, restauración de imágenes y mejoramiento de la calidad de las imágenes. Estas aplicaciones utilizan diversos algoritmos que pueden ser desde el cálculo de un histograma hasta operaciones más

J. C. Zavala-Díaz y M. A. Cruz-Chávez (Eds.): AGECOMP 2006, ISBN: 968-878-273-4, pp. 91-105, 2006.

complejas tales como la FFT (Fast Fourier Transform), DCT (Discret Coseno Transform), Transformada Wavelets entre otras.

El presente trabajo está basado en la plataforma de MatLab, permitiendo reunir en una aplicación distintas tareas para el procesamiento digital de imágenes. La sección 2 describe la plataforma de MatLab con la cual se efectuó la herramienta de apoyo para el procesamiento de imagen. La sección 3 contiene los detalles de ejecución de algunos algoritmos y los resultados obtenidos.

2 PLATAFORMA DE MATLAB

MATLAB es una herramienta de programación cuyo nombre proviene de la abreviación de "MATrix LABoratory". Esta herramienta, permite realizar cálculos numéricos con vectores y matrices. Como caso particular puede trabajar con números escalares reales o complejos, con cadenas de caracteres y con otras estructuras de información más complejas. Una de las capacidades más atractivas es la de realizar una amplia variedad de gráficos en dos y tres dimensiones. MATLAB es un gran programa de cálculo técnico y científico. Para ciertas operaciones es muy rápido, cuando puede ejecutar sus funciones en código nativo con los tamaños más adecuados para aprovechar sus capacidades de vectorización. En otras aplicaciones resulta bastante más lento que el código equivalente desarrollado en C/C++ o Fortran. Sin embargo, siempre es una magnífica herramienta de alto nivel para desarrollar aplicaciones técnicas, fácil de utilizar y que aumenta significativamente la productividad de los programadores respecto a otros entornos de desarrollo.
MATLAB dispone de un código básico y de varias librerías especializadas (toolboxes). Para el presente trabajo se hizo uso del Toolbox de Procesamiento de Imagen.

El Toolbox de Procesamiento de Imagen es una colección de funciones implementadas en Matlab que comprenden una extensa colección de operaciones para el tratamiento digital de imágenes, estas van desde la representación de imágenes hasta el filtrado, análisis o transformación de las mismas. La mayoría de las funciones del Toolbox de Imagen son ficheros .m, cuyo código puede visualizarse tecleando: type nombre_función.

El tipo de dato básico en Matlab es la matriz rectangular: un conjunto de elementos reales o complejos. Por lo tanto, una imagen es representada como una matriz, es decir, un conjunto ordenado de datos con valores reales de color o intensidad. De esta forma, un píxel (picture element) es un elemento de una matriz (imagen). Para acceder a un pixel, por ejemplo se utiliza la nomeclatura I (2,15) donde I nos da el valor de intensidad del píxel de la fila 2 y columna 15 en la matriz de imagen. Los tipos de imágenes que Matlab soporta son: imágenes indexadas, imágenes de intensidad, imágenes binarias e imágenes RGB (de color real).

3 PROCESAMIENTO DIGITAL DE IMAGEN V 1.0

La aplicación se implementó como herramienta de GUI del Matlab y se compiló externamente con Visual Basic.

El entorno visual de la aplicación se muestra en la Figura 1.

Fig. 1. Entorno visual del software.

La figura anterior muestra el entorno visual de la software, donde podemos apreciar en la parte superior izquierda la imagen cargada por el usuario, en la parte inferior izquierda muestra una operación realizada, en este caso es una transformación geométrica de rotación. En la parte superior derecha y parte inferior derecha se aprecian algunas de las múltiples operaciones (Transf. Geométricas, histogramas, umbralización, morfología, etc.) que se realizan con este software.

La imagen a procesar es cargada con el menú Archivo y puede ser obtenida de un archivo en disco duro u obtenida por una cámara digital.

A continuación se muestran las pruebas realizadas con los diferentes algoritmos programados.

3.1 Desarrollo del Software.

Como se menciono el software esta basado en la plataforma GUI de Matlab y a continuación se describe partes de como se desarrolló el código fuente.

Como primer punto mostraremos los pasos que seguimos para realizar el programa ejecutable, que no depende de Matlab.

3.1.1 Extracción de los Componentes de las Librerías de Matlab

Separación de los componentes de la librería utilizada de MatLab para compactarlos con el ejecutable creado en este mismo software.

Para realizar la extracción de los archivos contenidos en el msi de la librería de matlab se utiliza el programa WinINSTALL LE 8.0. , a continuación se explican los pasos para realizar esta tarea:

1. Después de instalar el software se ejecuta, al ejecutarlo realizamos los siguiente, (ver Figura2):

   File ⟶ Import Package…

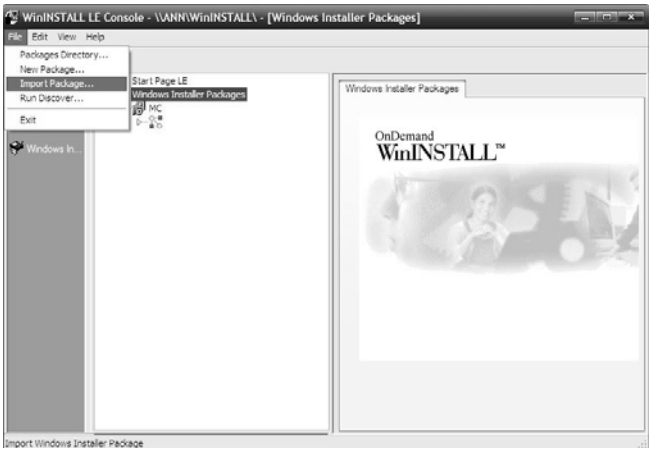
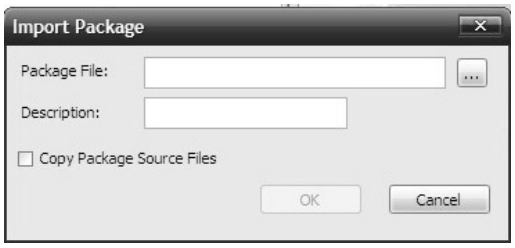


Fig. 2. Importación del paquete WinINSTALL LE 8.0

Al realizar esto aparecerá una nueva ventana en donde se indica la ubicación donde se encuentra la librería a importar:



Se le da una descripción para poder distinguirlo y se selecciona la opción de “copy package source files”, para llevar todos los archivos que contenga la librería.

2. Teniendo exportada la librería, se selecciona y aparecerán nuevos menús, en la Figura 3 de abajo se muestran los nuevos menús ya mencionados:

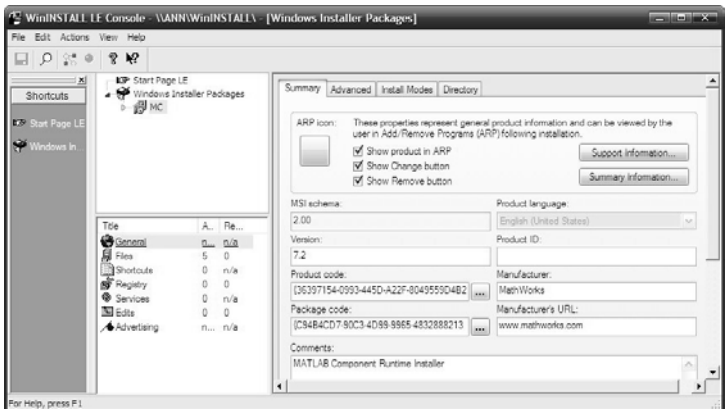


Fig. 3. Menú de las librerías exportadas

3. Para descomprimir los archivos del msi se realiza lo siguiente,(Ver Figura 4)

Actions Descompress…

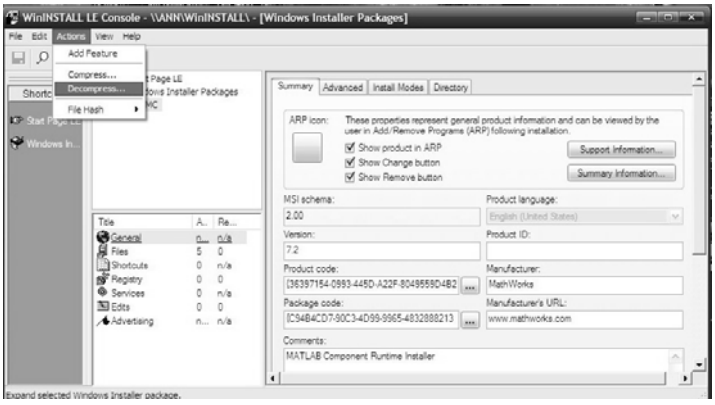


Fig. 4. Menú para Descomprimir Archivos

Aparecerá una nueva ventana, la cual muestra el progreso de la descompresión del archivo (Figura 5).

Fig. 5. Figura 5 Ventana de Progreso de Descompresión.

4. Los archivos ya descomprimidos se encuentran en la carpeta de instalación del programa, y están listos para compactarse con el ejecutable creado en MatLab.

Creación de un archivo ejecutable en MatLab

Pasos para crear el ejecutable:

1. Configurar el compilador a utilizar mediante el comando: *mbuild –setup,* en la ventana de comando.

2. Teclear el comando: *mcc -B sgl %nombre del archivo.m,* para crear el ejecutable.

Nota: Es necesario al instalar el Matlab, seleccionar las herramientas del compilador ya que sin estas no se podrá crear el archivo *.exe*.

3.1.2 Uso de Transformadas

Dentro de las diversas herramientas disponibles en esta aplicación existen las referentes al diseño de filtros mediante la implementación de transformadas de Fourier en dos dimensiones. He aquí algunos del diseño de filtros creados por el software:

3.1.2.1 Filtro Pasa-bajas

Al aplicar un Filtro Pasa Bajas (FPB) a una imagen lo que obtenemos como resultado es una imagen suavizada debido a la eliminación de las frecuencias altas que hay en la imagen, es decir, disminuye los cambios en intensidad entre píxeles consecutivos. En la Figura 6 se muestra un FPB con una frecuencia de corte normalizada W= 0.05.

El código que se utilizo para esta operación es el siguiente:

```
%FILTRADO Y DECIMACIÓN

%pasa bajas bajas

    bbdf1=filter2(hb',bhd1); bbDF1=bbdf1(1:2:F,:);
bbDF1a=bbDF1/max(max(bbDF1));

set(handles.temptext10,'String',num2str(bbDF1))

    bbdf2=filter2(hb',bhd2); bbDF2=bbdf2(1:2:F,:);
bbDF2a=bbDF2/max(max(bbDF2));

set(handles.temptext11,'String',num2str(bbDF2))

    bbdf3=filter2(hb',bhd3); bbDF3=bbdf3(1:2:F,:);
bbDF3a=bbDF3/max(max(bbDF3));

set(handles.temptext12,'String',num2str(bbDF3))

    bb=cat(3,bbDF1a,bbDF2a,bbDF3a);

%pasa bajas altas

badf1=filter2(ha',bhd1); baDF1=badf1(1:2:F,:);

set(handles.temptext1,'String',num2str(baDF1))

    badf2=filter2(ha',bhd2); baDF2=badf2(1:2:F,:);

set(handles.temptext2,'String',num2str(baDF2))

    badf3=filter2(ha',bhd3); baDF3=badf3(1:2:F,:);

set(handles.temptext3,'String',num2str(baDF3))

  ba=cat(3,baDF1,baDF2,baDF3)
```

La instrucción “filter2” filtra en dos dimensiones (imagen) por medio de convolución. La instrucción “cat” concatena las tres capas (RGB) para formar la imagen.

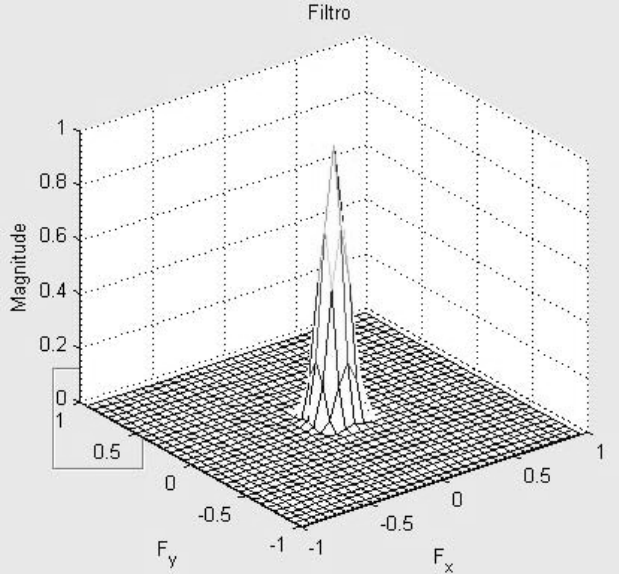


Fig. 6. FPB con frecuencia de corte W = 0.05.

a) b)

Fig. 7. a) Imagen Original, b) Imagen Filtrada.

3.1.2.2 Filtro Pasa-altas

Los Filtros Pasa Altas (FPA), en contraste con el anterior, incrementa los cambios entre píxeles consecutivos, lo que se refleja en el realce de bordes en la imagen. En la Figura 4 se muestran el diseño de un FPA y su forma tridimensional. Los resultados al aplicarse este filtro sobre una imagen se muestran en la Figura 9.

El código que se utilizo en esta operación es el siguiente

```
  %pasa altas bajas

  abdf1=filter2(hb',ahd1); abDF1=abdf1(1:2:F,:);

  set(handles.temptext4,'String',num2str(abDF1))

  abdf2=filter2(hb',ahd2); abDF2=abdf2(1:2:F,:);

  set(handles.temptext5,'String',num2str(abDF2))

  abdf3=filter2(hb',ahd3); abDF3=abdf3(1:2:F,:);

  set(handles.temptext6,'String',num2str(abDF3))

  ab=cat(3,abDF1,abDF2,abDF3);


  %pasa altas altas

      aadf1=filter2(ha',ahd1); aaDF1=aadf1(1:2:F,:);

      set(handles.temptext7,'String',num2str(aaDF1))


      aadf2=filter2(ha',ahd2); aaDF2=aadf2(1:2:F,:);

      set(handles.temptext8,'String',num2str(aaDF2))


      aadf3=filter2(ha',ahd3); aaDF3=aadf3(1:2:F,:);
  set(handles.temptext9,'String',num2str(aaDF3))

      aa=cat(3,aaDF1,aaDF2,aaDF3)
```

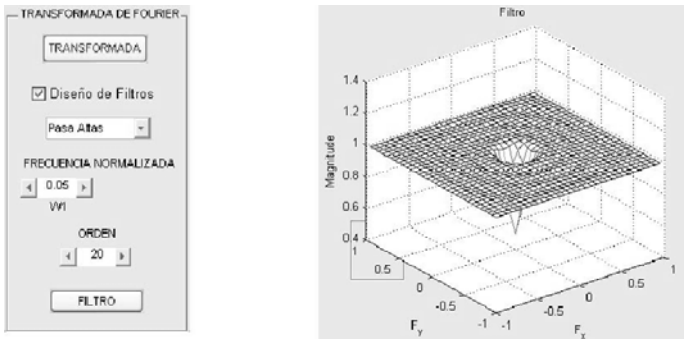


Fig. 8. a) Cuadro de Filtro Pasa Altas, b) Gráfico de un Filtro Pasa Altas.

a) b)

Fig. 9. a) Imagen Original, b) Imagen Filtrada.

3.1.2.3 Filtro Pasa-bandas

Este tipo de filtros solo seleccionan una banda de frecuencia, por lo tanto, si el usuario define una banda muy ancha se obtendrá la mayor parte de la información contenida en la imagen. Si por el contrario, el usuario define una banda muy pequeña, el resultado dependerá de las frecuencias que se dejen pasar ver (Figura 10).

Fig. 10. a) Imagen Original, b)Imagen Filtrada. .

3.2 Capas RGB y Umbralización

La aplicación permite tambien procesar imágenes en color, ya sea capas por separado R (Red), G (Green) o B(Blue) ó de la imagen completa ver (Figura 11.).

Fig. 11. Ejemplo de la extracción de la Capa Roja en una Imagen a Color en 256 tonos de rojo.

La umbralización es una de las operaciones que pueden aplicarse sobre una capa de la imagen a color, se selecciona un tono en especifico y se binariza la imagen en solo dos tonos, dando el valor cero al mínimo (negro) y un valor 255 al máximo (blanco), Figura 12.

Fig. 12. Figura 8. Ejemplo de Umbralización.

3.2 Transformaciones Geométricas

Esta herramienta permite hacer diferentes transformaciones geométricas a la cualquier imagen que se deseé procesar, aquí se explican algunas de las más comunes.

3.2.1 Interpolaciones

Cuando se requiere de ampliar o reducir una imagen se requiere de la implementación de algunas herramientas matemáticas para realizar dicha tarea. Independientemente de la técnica que se aplique, se requiere de cierto tiempo para ser procesada y mientras más calidad se requiera en la imagen mucho más pesado será el cálculo de los nuevos

valores en la interpolación. La Figura 13 muestra un ejemplo de los tipos de interpolación que se implementan en este trabajo, siendo la interpolación bicúbica la que presenta mejor desempeño en definición de imagen.

Original 25x26 píxeles

Vecino más próximo 250x260

Fig. 13. Imagen amplificada 10x por las diferentes interpolaciones disponibles en el panel de transformaciones geométricas.

3.2.2 Rotación y Perspectivas

Muchas aplicaciones requieren rotación de imagen, nuestra aplicación cuenta con esta operación y es posible también cambiar la perspectiva de la imagen. La Figura 14 muestra un ejemplo.

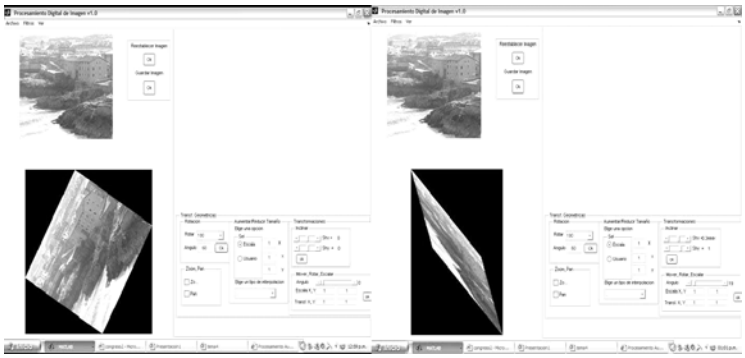


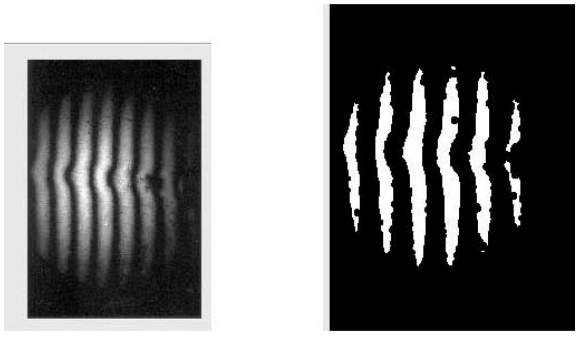
Fig. 14. Ejemplos de rotación y perspectiva.

3.3 Morfología

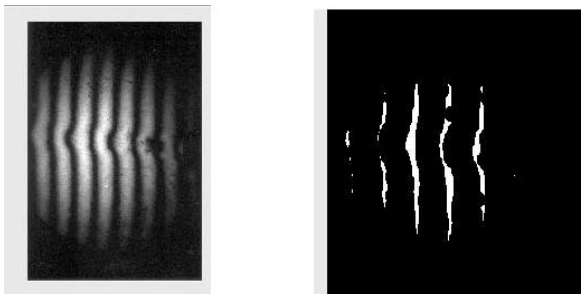
La morfología de las imágenes nos ayuda a determinar la forma de los cuerpos u objetos contenidos en una imagen. La operaciones implementadas en esta técnica son de naturaleza binaria, para efectuar cambios en la imagen se utilizan mascaras.

3.3.1 Erosión

Dentro de las herramientas de morfología que utilizamos se encuentra la erosión de imágenes binarias o umbralizadas, lo que se desea obtener con esta operación es el definir de una manera mas exacta, siguiendo ciertos criterios, la forma de los objetos a estudiar. La erosión “desgasta” los bordes de los cuerpos Figura 15.



a).

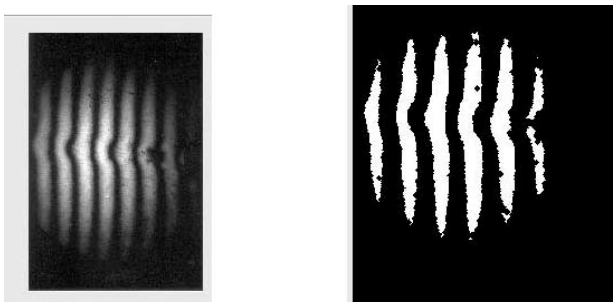


b)

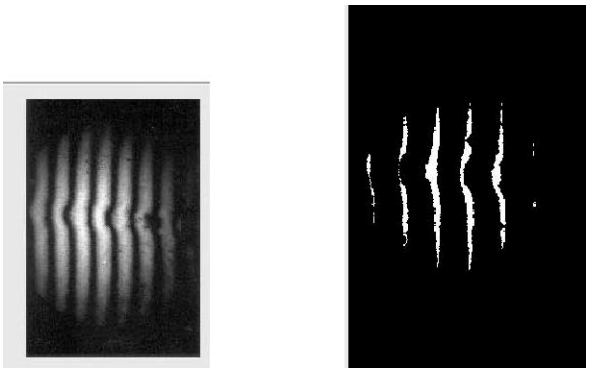
Fig. 15. Ejemplos de erosión. a) Ejemplo de la poción Erosión, con tipo de mascara Disk a 5 coeficientes. b) Ejemplo de la poción Erosión, con tipo de mascara Disk a 10 coeficientes.

3.3.2 Dilatación

En la dilatación dichos objetos sufren la operación opuesta a la erosión, es decir, aumentan en área, Figura 16.



a)



b)

Fig. 16. Ejemplos de Dilatación. a) Ejemplo de la poción Dilatación, con tipo de mascara Diamond a 5 coeficientes. b) Ejemplo de la poción Dilatación, con tipo de mascara Diamond a 10 coeficientes.

4. CONCLUSIONES

Se ha desarrollado una aplicación que sirve como herramienta de apoyo para la impartición de cursos de Procesamiento Digital de Imágenes, la cual esta basada en MatLab, pero es independiente de esa plataforma.
La aplicación permite utilizar de una forma clara y sencilla algoritmos complejos que son utilizados en el procesamiento de imagen, a su vez tiene opciones que pueden guardar resultados, permitiendo así no solo su uso para enseñanza, sino también como procesador de imagen.

REFERENCIAS

1. R. C. González, R. E. Woods, Tratamiento Digital de Imágenes, Addison-Wesley/Diaz Santos.

2. S. L. Eddins R. Gonzalez, R. E. Woods. Digital Image Processing using MATLAB.

3. Mathworks Inc., http://www.mathworks.es/

4. Toolbox de MatLab