Tecnológico de Monterrey

Actividad de Semana i

Taller de Procesamiento de Imágenes Utilizando Matlab y Visita a CICATA-IPN Unidad Qro

Dra. Araceli Soto Hernández

25 de Septiembre del 2018

Contenido

3. Procesamiento de Imágenes en Matlab

3.1. Gráficos en 3D.

3.2. Detectar bordes en una imagen

3.3. Filtros en Matlab

3.1. Gráficos en 3D

MATLAB proporciona funciones para visualizar matrices en 2D, escalares en 3D y vectores de datos 3D. Puede usar estas funciones para visualizar y entender datos grandes, complejos y multidimensionales. Puede especificar las características de la gráfica como la vista de la cámara, perspectiva, efecto de iluminación y transparencias. Las funciones de graficación en 3D incluyen:

- ✓ Superficies, contornos y mallas
- ✓ Graficar imágenes
- ✓ Cono, sección, cascadas e isosuperficies

3.1. Gráficos en 3D

Importar y Exportar archivos gráficos

MATLAB permite leer y escribir en diversos formatos de archivos gráficos, como GIF, JPEG, BMP, EPS, TIFF, PNG, HDF, AVI y PCX. Como resultado, puede exportar las gráficas de MATLAB a otras aplicaciones como Microsoft Word y Microsoft PowerPoint, o publicar software. Antes de exportar, puede crear y aplicar templates de estilo, cubrir características como el diseño, fondo y escala, para cumplir con las especificaciones.

3.1. Gráficos en 3D

Los gráficos de malla y superficie son gráficos tridimensionales utilizados para representar funciones que tienen la forma:

$$z = f(x, y)$$

Donde:
x e y son variables independientes.
z es la variable dependiente.

Los gráficos de malla y de superficie se generan en tres pasos:
1.- Crear una malla o rejilla en el plano x-y que cubra el dominio de la función.
2.- Calcular el valor de z en cada punto de la rejilla.
3.- Representar el gráfico.

3.1. Gráficos en 3D

Funciones en Matlab para Graficar Función en 3D

Función	Para que sirve?
meshgrid	Genera una matriz a partir de vectores. También coloca líneas de cuadricula definidas por el usuario en gráficos bidimensionales y tridimensionales. [X,Y]=meshgrid(x,y)
mesh	Genera gráficos de malla (o de mallado). mesh(X,Y,Z)
surf	Genera un gráfico de superficie. surf(X,Y, Z)
meshz	Genera gráfico de malla con cortina (dibuja una cortina alrededor de la malla). meshz(X,Y, Z)
meshc	Genera gráfico de malla con contorno. meshc(X,Y,Z)

3.1. Gráficos en 3D

Funciones en Matlab para Graficar Función en 3D

Función	Para que sirve?
surfc	Genera gráfico de superficie con contorno. surfc(X,Y, Z)
Surfl	Genera gráfico de superficie con alumbrado. Surfl (X,Y,Z)
waterfall	Genera gráfico de cascada (dibuja malla unidimensional). waterfall(X,Y,Z)
contour3	Genera gráfico de contorno 3-D. Contour3(X,Y,Z,n)
contour	Genera gráfico de contorno 2-D. Contour(X,Y,Z,n)

3.1. Gráficos en 3D

Programa 1: Hacer un programa de Matlab para graficar la siguiente expresión:

$$z = x^2 y - 2x$$

Para el dominio: $-5 \leq x \leq 5$ $-5 \leq y \leq 5$

Representar con grafico tipo malla y superficie. Observe la diferencia entre ambos.

3.1. Gráficos en 3D

Programa 2: Hacer un programa de Matlab para graficar la siguiente expresión:

$$z = \sqrt{x^2 + y^2}$$

Para el dominio: $-3 \leq x \leq 3$ $-3 \leq y \leq 3$

Representar con grafico tipo malla y superficie.

3.2. Detectar bordes en una imagen

Existen muchas maneras en las que podemos manipular imágenes. En general, buscamos “mejorar” la imagen con el fin de facilitar su interpretación y extraer información de ella. Algunos ejemplos podrían ser:

- ✓ Me interesa que la imagen tenga bordes bien definidos para hacer segmentaciones.
- ✓ Me interesa el reconocimiento de algún patrón.
- ✓ Me interesa hacer un registro entre imágenes.

3.2. Detectar bordes en una imagen

La idea en la mayoría de las técnicas de detección de bordes, es calcular un operador local de derivación y decir que un pixel pertenece a un borde si se produce un cambio brusco de los niveles de gris de sus vecinos.

3.2. Detectar bordes en una imagen

[g,t] = edge(f, ‘método´, parámetro)

Esta función encuentra los bordes de una imagen de distintos niveles de intensidad.

El resultado es una imagen binaria del mismo tamaño que la imagen original en la cual, “1” significa que ha detectado un borde y “0” es que no lo ha detectado.

3.2. Detectar bordes en una imagen

[g,t] = edge(f, ‘método´, parámetro)

Donde:

- f es la imagen de entrada
- método: Sobel, Prewitt, Roberts, Laplaciano, Cruce por cero, Canny.
- g es un arreglo lógico con 1s donde se detectó un borde y 0 en otro lugar.
- parámetro t es opcional, usado por el gradiente para saber que punto pertenece al borde (umbral).

3.2. Detectar bordes en una imagen

Métodos:

- ✓ Sobel.- El filtro Sobel detecta los bordes horizontales y verticales separadamente sobre una imagen en escala de grises.

- ✓ Canny.- El operador de detección de bordes de Canny fue desarrollado por un catedrático de la universidad de Berkeley (EEUU) en 1986 y se basa en un algoritmo de múltiples fases para detectar un amplio rango de bordes.

Sobel Filter

Canny Filter

3.2. Detectar bordes en una imagen

1.- Leer imagen

```
Iori=imread('c:\ara\celula.jpg');
```

2.- Convertir RGB a Gris

```
I=rgb2gray(Iori);
```

3.- Detecta bordes de una imagen en escala de gris.

```
borde=edge(I, ‘canny');
```

3.2. Detectar bordes en una imagen

Ejercicio 1 en Clase

a)Buscar en Internet 5 imágenes que tengan distintos tipos de bordes.

b)Hacer un programa en Matlab para leer las 5 imágenes y hacer detección de bordes utilizando el método de sobel y canny.

c)Observar los resultados de ambos métodos

3.3. Filtros en Matlab

FILTROS ESPACIALES

Estos filtros tienen como objetivo la contribución de determinados rangos de frecuencia de una imagen. El termino espacial se refiere a que el filtro se aplica directo a la imagen y no a una transformada de la misma.

Existen muchos tipos de filtros que podemos utilizar para manipular una imagen, aquí veremos uno de los más comúnmente utilizados que es un filtro Gaussiano.

3.3. Filtros en Matlab

FILTROS ESPACIALES

Generalmente utilizamos el filtro gaussiano para reducir el ruido en una imagen.

El ruido son variaciones en los valores de los pixeles de la imagen debidas a errores que pudieron darse durante la adquisición de la imagen, durante el almacenamiento, durante la transmisión o durante la compresión de la imagen.

3.3. Filtros en Matlab

FILTROS

El filtrado espacial de imágenes en MATLAB se puede realizar con la instrucción **imfilter()** que tiene el formato:

Imagen_b = imfilter (imagen a procesar, w, opción);

Donde:

imagen_b =imagen destino
imagen a procesar =imagen que se va a filtrar
w es la máscara para el filtrado
opción es el método a utilizar, ver tabla 3.3.1.

3.3. Filtros en Matlab

Tabla 3.3.1. Opciones para **imfilter**

Opciones	Descripción
Método	
'symmetric'	La imagen se extiende reflejándola más allá de sus orillas. Se hace un efecto de espejo en la imagen.
'replicate'	La imagen se extiendo replicando sus valores más allá de sus orillas. Es decir, los bordes de la imagen se extienden con pixeles que replican el valor de los pixeles que son borde
'circular'	La imagen se extiendo como si fuera una señal de una dimensión.
Dirección	
'pre'	Hacerlo antes del primer elemento de cada dimensión.
'post'	Hacerlo después del último elemento de cada dimensión.
'both'	Emplear los dos métodos 'pre' y 'post'.

3.3. Filtros en Matlab

El tipo de filtro se obtiene usando la instrucción fspecial que tiene el siguiente formato:

w = fspecial (‘tipo’, parámetros)

Donde
‘tipo’ es alguno de los tipos de filtros, ver tabla 3.3.2.
‘parámetros’ especifica los detalles del filtro.

3.3. Filtros en Matlab

Tabla 3.3.2. Tipos de Filtros

w = fspecial (‘tipo’, parámetros)

Tipo	Sintaxis	Sintaxis y parametro
‘average’	fspecial(‘average’, [r,c])	Filtro de promedio rectangular tamaño rxc. **w = fspecial (‘average’, [r,c])**
‘disk’	fspecial(‘disk’, r)	Filtro de media, circular de promedio de radio r. **w = fspecial (‘disk’, [r,c])**
‘gaussian’	fspecial(‘gaussian’, [r,c], sig)	Filtro gausiano de tamaño r x c con desviación estándar sig. **w = fspecial (‘gaussian’, [r,c], sig)**

3.3. Filtros en Matlab

Tabla 3.3.2. Tipos de Filtros

w = fspecial (‘tipo’, parámetros)

Tipo	Sintaxis	Sintaxis y parametro
‘laplacian’	fspecial(‘laplacian’, alfa)	Filtro laplaciano de 3 x 3 con alfa entre 0 y 1. **w = fspecial (‘laplacian’, alpha)**
‘log’	fspecial(‘log’, [r,c], sig)	Laplaciano de una gausiana (log). **w = fspecial (‘log’, [r,c], sig)**
‘motion’	fspecial(‘motion’, len, teta)	El resultado aproxima un filtro en movimiento por len pixeles cuando se convoluciona con una imagen. **w = fspecial (‘motion’, len,theta)**

3.3. Filtros en Matlab

Tabla 3.3.2. Tipos de Filtros

w = fspecial (‘tipo’, parámetros)

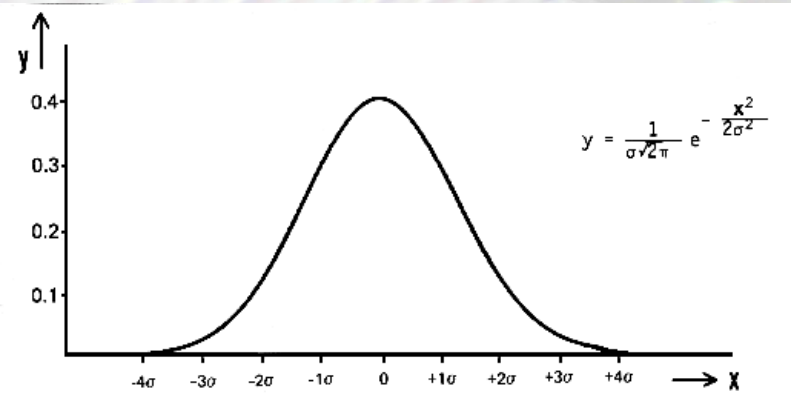
Tipo	Sintaxis	Sintaxis y parametro
‘prewitt’	fspecial(‘prewitt’)	Crea un filtro de prewitt por defecto detecta gradientes verticales. **w = fspecial (‘prewitt)**
‘sobel’	fspecial(‘sobel’)	Crea un filtro 3 x 3 de sobel por defecto detecta gradientes verticales. **w = fspecial (‘sobel’)**
‘unsharp’	fspecial(‘unsharp’,alpha)	Crea un filtro 3x3 tipo sombra. Alpha controla la forma, este debe ser mayor que 0 y menor o igual a 1. **w = fspecial (‘unsharp’, alpha)**

3.3. Filtros en Matlab

También podemos agregar ruido a una imagen, utilizando función de Matlab.

Ir = imnoise(Im, ‘tipo’)

Donde Im es la imagen antes del ruido.



3.3. Filtros en Matlab

EJERCICIO 1: Aplicar filtro gaussiano con diferentes sigmas a la imagen cameraman.tif

EJERCICIO 2: Aplicar ruido sal y pimienta a la imagen eight.tif y después “limpiar” el ruido con un filtro de mediana.

EJERCICIO 3: Buscar una imagen en Internet y aplicar algún filtro.

3.3. Filtros en Matlab

Programa 1: Hacer un programa en Matlab para edición de imágenes, aplicar filtro gaussiano. seguir los siguientes pasos:

1.- Leer imagen original.

2.- Agregar ruido a la imagen.

3.- Aplicar mascara gaussian.

4.- Aplicar la opción de filtrado gaussian a la imagen con ruido.

5.- Representación de las 3 imágenes en ventanas pequeñas

3.3. Filtros en Matlab

Programa 2: Hacer un programa en Matlab para edición de imágenes, aplicar filtro gaussiano. seguir los siguientes pasos:

1.- Leer imagen original.

2.- Agregar ruido a la imagen.

3.- Aplicar mascara average.

4.- Aplicar la opción de filtrado average a la imagen con ruido.

5.- Representación de las 3 imágenes en ventanas pequeñas

3.3. Filtros en Matlab

Programa 3: Hacer un programa en Matlab para edición de imágenes. seguir los siguientes pasos:

1.- Leer imagen original.

2.- Aplicar mascara motion.

3.- Aplicar la opción de filtrado replicate Efecto Movido.

4.- Representación de las 2 imágenes en ventanas pequeñas

3.3. Filtros en Matlab

Programa 4: Hacer un programa en Matlab para edición de imágenes. seguir los siguientes pasos:

1.- Leer imagen original.

2.- Aplicar mascara disk.

3.- Aplicar la opción de filtrado replicate Efecto Borroso.

4.- Representación de las 2 imágenes en ventanas pequeñas

3.3. Filtros en Matlab

Programa 5: Hacer un programa en Matlab para edición de imágenes. seguir los siguientes pasos:

1.- Leer imagen original.

2.- Aplicar mascara unsharp.

3.- Aplicar la opción de filtrado replicate Mejorar contraste.

4.- Representación de las 2 imágenes en ventanas pequeñas

Bibliografía

1.- Etter M. Delores. **Solución de Problemas de Ingeniería con Matlab**. 2ª Ed. Prentice Hall. 1997.

2.- Javier García de Jalón *et al.* **Aprenda Matlab 7.0 Como si estuviera en primero**. Universidad Politécnica de Madrid. 2005.

3.-http://www.utm.mx/~vero0304/HCPM/11-movies.pdf