Taller 3

1. Desarrolle una función en Matlab que permita rotar una imagen de entrada "I" en escala de grises un ángulo "theta" en grados medido desde el eje vertical de la imagen (eje x de acuerdo con la convención vista en el curso), en el sentido de giro contrario a las manecillas del reloj. La imagen de salida resultante "I2" debe tener las mismas dimensiones que la imagen de entrada. El prototipo de la función es

```
function I2=rotar_imagen(I,theta)
```

Nota: recuerde que las funciones trigonométricas en Matlab asumen que el ángulo ingresa en radianes.

2. Implemente una función en Matlab que permita aplicar el filtrado lineal a una imagen "*I*" usando una máscara "*h*" de dimensiones *M* x *N* (donde usted puede asumir que *M* y *N* son enteros impares). La función debe retornar la imagen de salida "I2" recortada, es decir, debe tener las mismas dimensiones de la imagen de entrada.

```
function I2=aplicar_filtrado_lineal(I,h)
```

3. Desarrolle una **función** en Matlab que permita aplicar los procesamientos que se indican a continuación a una imagen de entrada en escala de grises "I", para generar la imagen de salida "I2", de acuerdo con el valor numérico que le asigne el usuario al argumento "sel" en el siguiente prototipo:

```
function I2=procesamiento_de_imagenes(I,sel)
```

sel:

1. Aplicar el **procesamiento local de ecualización del histograma** sobre una vecindad de 3x3 pixeles.

2. Aplicar un **filtro paso bajo promedio** con una máscara de 15x15 pixeles.

3. Aplicar un **filtro paso bajo no-lineal tipo mediana** usando una vecindad de 9x9 pixeles.

4. **Realzar** la imagen de entrada con el **Laplaciano**. Debe usar la máscara:

[-1 -1 -1; -1 8 -1; -1 -1 -1].

Nota 1: Puede asumir que la imagen de entrada "I" está en escala de grises y que su intensidad ha sido cuantizada con 8 bits. En todos los casos la imagen de salida "I2" debe tener las mismas dimensiones que la imagen de entrada y debe estar en formato entero sin signo de 8 bits (**uint8**). El valor mínimo de intensidad en la imagen de salida "I2" **no puede ser** inferior a 0, y el valor máximo no puede ser superior a 255.

Nota 2: La mediana de un vector en Matlab se calcula con el comando **median**. Para encontrar la mediana de una matriz, primero convierta la matriz a vector, y después aplique el comando median.

4. Diseñe una función en Matlab que permita crear una imagen panorámica "I3", tan grande como sea necesaria, a partir de dos imágenes "I1" e "I2" y una matriz "M" que contiene 4 puntos de control o correspondencias en ambas imágenes de la siguiente manera:

$$M = [x_1\ y_1\ v_1\ w_1$$

$$x_2\ y_2\ v_2\ w_2$$

$$x_3\ y_3\ v_3\ w_3$$

$$x_4\ y_4\ v_4\ w_4];$$

Donde el punto (x_i,y_i) en la imagen I1 corresponde con el punto (v_i,w_i) en la imagen I2 para i=1, 2, 3, 4. El prototipo de la función que debe implementar está dado por:

```
function I3=crear_panoramica(I1,I2,M)
```

Nota: puede usar el siguiente modelo de transformación:

$$x=c1*v+c2*w+c3*v*w+c4$$

$$y=c5*v+c6*w+c7*v*w+c8$$

donde c1, c2, c3, c4, c5, c6, c7, c8 son constantes que dependen de los datos de entrada.

5. Desarrolle una función que permita calcular y graficar tanto la magnitud como la fase del gradiente de una imagen de entrada en escala de grises.

6. Desarrolle una función que permita aplicar el filtrado de desenfoque a una imagen de entrada en escala de grises. Recuerde que los parámetros que caracterizan esta técnica debe ser argumentos de la función para que el usuario pueda modificarlos.

7. Genere un mapa de colores de su autoría con al menos 256 muestras y aplíquelo a una imagen en escala de grises cuantizada con 8 bits. Muestre en una misma ventana la imagen en escala de grises y la imagen resultante en color falso.

8. Realice una función en Matlab que permita convertir una imagen de RGB a HSI usando las expresiones vistas en clase para tal fin.

9. Desarrolle una función en Matlab que calcule la transformada discreta directa de Fourier "F" a partir de una imagen en escala de grises "f" de tamaño arbitrario. La función debe retornar además de "F", el espectro de fase "EF" (en radianes), el espectro de magnitud "EM" (en números), y el espectro de potencia "EP" (en números). Prototipo:

```
function [F,EF,EM,EP]=mi_DFT2D(f)
```

10. Desarrolle una función en Matlab que calcule la transformada discreta inversa de Fourier "f" a partir de una matriz que contiene la transformada de Fourier "F" de tamaño arbitrario. Prototipo:

```
function [f]=mi_IDFT2D(F)
```

11. (10%) Desarrolle una función en Matlab que multiplique cada elemento de una matriz de entrada por un factor de la forma $(-1)^{(x+y)}$, donde “x” y “y” son variables que indican la fila y columna del elemento. Prototipo:

```
function mtx_out=mi_multiplicacion(mtx_in)
```

12. (10%) Desarrolle una función en Matlab que a partir de la distancia de corte D0, le entregue al usuario un filtro paso bajo Gaussiano “GLPF” y un filtro paso alto Gaussiano “GHPF” de dimensiones “P” x “Q”. Prototipo:

```
function [GLPF, GHPF]=calcular_filtros(D0,P,Q)
```

13. (40%) Desarrolle una función en Matlab que realice el proceso de filtrado en frecuencia de una imagen “f” de dimensiones “M” x “N”, con un filtro en frecuencia arbitrario “H” que es real, simétrico, de dimensiones “2M” x “2N” y que se encuentra centrado. La función debe retornar la imagen filtrada “g” de forma tal que tenga las mismas dimensiones de la imagen de entrada “f”. Tenga en cuenta el proceso de relleno de ceros, y recuerde centrar la transformada de Fourier para aplicar el filtro. Puede usar las funciones de los puntos anteriores para tal fin si lo considera necesario. Prototipo:

```
function g=filtrado_en_frecuencia(f,H)
```