Ejercicios del Tema 7

Gonzalo Galiano Casas

Departamento de Matemáticas
Universidad de Oviedo
galiano@orion.ciencias.uniovi.es

Nombre:

Instrucciones

Se trata de realizar los ejercicios adjuntos programando en Matlab o Python. No es necesario que me enviéis ningún archivo de imagen.

Para enviarme los resultados tenéis las siguientes opciones:

- Si usáis Linux, podréis abrir este documento con un lector de pdf's como Okular u otros y acceder a las cajas de **Respuesta** de los ejercicios, donde debéis pegar directamente vuestros códigos de modo que, copiando y pegando, yo pueda ser capaz de reproducir vuestros resultados en Matlab o Python. Al finalizar, guardar el archivo pdf dándole el nombre: *VuestrosApellidos.pdf*.

- Acrobat Reader no permite realizar la operación anterior. De modo que si no disponéis de otro lector de pdf's que sí os permita escribir en las cajas de **Respuesta**, lo más cómodo será que me enviéis directamente los scripts generados en Matlab o Python. Como hay cinco ejercicios, habrá otros tantos scripts. Por favor, comprimirlos en un único archivo y dadle el nombre *VuestrosApellidos.zip*.

Para hacer los ejercicios se necesitan los archivos:

- cameraman_512.tif y cameraman_256.tif, a las que nos referimos como *cameraman*,

- lena_gray_512.tif, a la que nos referimos como *Lena*,

- boat_256.png, a la que nos referimos como *boat*

Comentarios

Los ejercicios correspondientes al Laboratorio 1 son fáciles y están ideados para entrenarse con el manejo de matrices y la representación de imágenes.

En el correspondiente al Laboratorio 2, entramos en materia. Su programación con Matlab es especialmente simple, debido a las funciones `gradient` y `divergence`, que nos evitan preocuparnos de la discretización espacial de la imagen. Con Python, es un poco más enrevesado, pero en el Laboratorio se dan indicaciones claras de cómo hacerlo. Como veréis, los resultados de un código tan simple son razonablemente buenos. En esta sección, se propone también un ejercicio de teoría.

El ejercicio correspondiente al Laboratorio 3 exige algo más de trabajo, aunque si se ordenan bien los cálculos, el código no debería superar las veinte líneas. Lo más conveniente, tanto en Matlab como en Python, es usar la vectorización siempre que se pueda, pues ahorra mucho tiempo de ejecución. Con esto me refiero a que si tenéis que multiplicar dos matrices, elemento a elemento, es mejor usar el operador $.*$ de Matlab, o $*$ de Python, que iterar a lo largo de dos bucles.

Ejercicios del Laboratorio 1

Ejercicio 1.1. Escribir una función cuya entrada (input) sea

- una imagen de cualquier tipo,
- unos rangos para sus píxeles, por ejemplo $23 \leq i \leq 53$ y $12 \leq j \leq 65$,

y cuya salida (output) sea la submatriz (tipo double) que corresponda a los índices dados y una figura que contenga la imagen. Usar la función para seleccionar la cabeza del *cameraman* en cameraman_512.tif.

Respuesta

Ejercicio 1.2. Las máscaras son filtros geométricos de una imagen. Por ejemplo, si queremos seleccionar una región de una imagen, podemos hacerlo multiplicando la matriz de la imagen original por una matriz de igual tamaño que contenga unos en la región que queremos conservar y cero en el resto. En este ejercicio seleccionaremos una región circular de la imagen *Lena* de radio 150. Estos son los pasos:

- Leer la imagen y convertirla a escala de grises y doble precisión.
- Crear una matriz de las mismas dimensiones rellena de ceros.
- Modificar esta matriz de forma que contenga unos en un círculo de radio 150, es decir, si $(i-c_x)^2+(j-c_y)^2 < 150^2$, donde $(c_x,c_y) = (m/2,n/2)$ es el centro de la imagen.
- Multiplicar la imagen por la máscara (recordar que son matrices).
- Mostrar el resultado.

Cuando se multiplica por cero, se convierten a negro los píxeles de fuera del círculo. Modifica el programa para hacer visible esos píxeles con la mitad de su intensidad.

Respuesta

Ejercicio 1.3. El degradado lineal es un efecto en el que se oscurece una imagen vertical u horizontalmente. Podemos hacer esto con una máscara que sea constante por columnas pero que tome un valor decreciente por filas, desde uno en la primera fila hasta cero en la última. Construir dicha matriz y aplicarla a la imagen *Lena*.

Indicación: Un modo de plantear el código es usando bucles y condicionales. Sin embargo, vectorizar ahorra tiempo de ejecución: con el comando `linspace` puede efectuarse la degradación, y mediante `repmat` (en Matlab) o `tile` (en Python) puede construirse, a partir del vector obtenido con `linspace`, la matriz máscara.

Respuesta

Ejercicios del Laboratorio 2

Observación. Al leer una imagen y transformarla a doble precisión el resultado puede estar escalado en uno de los intervalos $I_1 = [0,1]$ (como en la versión Matlab) o en $I_{255} = [0,255]$ (como en la versión Python). Esto hace que algunnos parámetros varíen dependiendo de la escala. Por ejemplo, añadir un ruido gaussiano del 5 % a una imagen escalada en I_1 llevará un coeficiente distinto que si está escalada en I_{255}. Y lo mismo ocurre con otros parámetros. Avisaremos estos cambios cuando corresponda.

Ejercicio 2.1. Siguiendo el esquema visto para el caso del laplaciano, escribir un código que ejecute la minimización de la variación total. Aplicarlo a la imagen boat_256.png corrompida con ruido gaussiano del 5 %, usando los valores de los parámetros

- $\lambda = 10,\ \tau = 0{,}01,\ \varepsilon = 1.e-6$, num_iter=20, si la escala es I_1,
- $\lambda = 0{,}1,\ \tau = 0{,}75,\ \varepsilon = 1.e-6$, num_iter=20, si la escala es I_{255}.

Mostrar el resultado en una figura.

Respuesta

Ejercicio 2.2. (Teoría) Calcular la derivada direccional del funcional de variación total,

$$J(v) = \int_\Omega |\nabla v(x)|\, dx.$$

Indicación: Usar que $|s| = \sqrt{|s|^2}$.

Respuesta

Ejercicios del Laboratorio 3

La observación de la sección anterior aplica a este ejercicio.

Ejercicio 3.1. En el Ejercicio 2.1 hemos realizado un código para la eliminación de ruido mediante la minimización de la variación total. En este, comenzaremos por escribir un código que implemente el filtro de Yaroslavsky. Una vez compuestos los dos códigos, vamos a comparar los resultados aplicados a la imagen `boat_256.png`, siguiendo lo visto en el Laboratorio 3. Haremos lo siguiente:

- Leer la imagen `boat_256.png` y transformarla a doble precisión, escala de grises.
- Crear una nueva imagen añadiendo a la anterior un ruido gaussiano del 5 %.
- Usar el código de variación total con los parámetros
  - $\lambda = 10,\ \tau = 0{,}01$, $\varepsilon = 1.e-6$, num_iter=20, si la escala es I_1,
  - $\lambda = 0{,}05,\ \tau = 0{,}5$, $\varepsilon = 1.e-6$, num_iter=20, si la escala es I_{255}.
- Usar el filtro Yaroslavsky con los parámetros
  - $h = 0{,}15$ y $\rho = 30$, si la escala es I_1,
  - $h = \rho = 30$, si la escala es I_{255}.

  De este modo, el cuadrado en el que se realiza la integración es de tamaño 61×61.
- Visualizar las imágenes resultantes y sus curvas de nivel.
- Calcular los PSNR's de las imágenes restauradas respecto la imagen original sin ruido. Obsérvese que *PSNR no* es una función predefinida, hay que escribirla.

Indicación: Para la integración del filtro de Yaroslavsky utilizamos la aproximación siguiente

$$\int_{B(x,\rho)} f(x)dx = \sum_{x \in B(x,\rho)} f(x).$$

Nota: El filtro de Yaroslavsky es algo más sencillo de programar que el bilateral, dado que la integración se limita directamente al cuadrado $B(x,\rho)$. En el filtro bilateral, en principio, la integración ha de realizarse sobre toda la imagen. Sin embargo, puesto que la función gaussiana decae a cero muy rápidamente, normalmente se limita el recinto de integración a un cuadrado de radio 2ρ.

Respuesta