Práctica 3: Umbralización de imágenes utilizando conjuntos difusos y conjuntos intervalo-valorados difusos

La segmentación de imágenes consiste en dividir una imagen en regiones o clases disjuntas que tengan propiedades o atributos similares. Uno de los métodos más utilizados para segmentar una imagen globalmente es el proceso de umbralización. La característica de estos métodos es suponer que cada región está caracterizada por los niveles de intensidad de grises de la imagen. En esta práctica vamos a centrarnos en la división de imágenes en dos clases: píxeles que pertenecen al fondo de la imagen y píxeles que corresponden al objeto de la imagen.
*El proceso de umbralización consiste en encontrar un nivel de gris **t** tal que los píxeles de mayor intensidad que **t** pertenecen al fondo (u objeto) y los menores pertenecen al objeto (o fondo).*

Algoritmo 1: umbralización de imágenes maximizando medidas de similaridad.

La idea principal de este algoritmo es el siguiente: para cada nivel de intensidad **t**, creamos un conjunto difuso Qt. Este conjunto difuso representa la pertenencia de cada nivel de intensidad al fondo o al objeto al establecer que **t** es el umbral de la imagen. La pertenencia de cada elemento al conjunto viene dado por lo parecido que es cada nivel de intensidad con la media de intensidades del fondo o el objeto. Este parecido va a ser medido utilizando funciones de equivalencia restringidas. A continuación se presenta el esquema de Algoritmo 1:

1. **FOR t = 0 TO L-1 DO**
   1.1. Calcular la media de intensidades del fondo y del objeto

$$m_b(t) = \frac{\sum_{q=0}^{t} q \cdot h(q)}{\sum_{q=0}^{t} h(q)} \qquad m_o(t) = \frac{\sum_{q=t+1}^{L-1} q \cdot h(q)}{\sum_{q=t+1}^{L-1} h(q)}$$

   1.2. Calcular el conjunto difuso

$$Q_t = \left\{ (q, \mu_{Q_t}(q)) \mid q \in \{0,1,\ldots,L-1\} \right\}$$

$$\mu_{Q_t} = \begin{cases} F\left(\frac{q}{L-1}, \frac{m_b(t)}{L-1}\right) = \varphi\left(REF\left(\frac{q}{L-1}, \frac{m_b(t)}{L-1}\right)\right) & if\ q \le t \\ F\left(\frac{q}{L-1}, \frac{m_o(t)}{L-1}\right) = \varphi\left(REF\left(\frac{q}{L-1}, \frac{m_o(t)}{L-1}\right)\right) & if\ q > t \end{cases}$$

1.3. Calcular la similaridad del conjunto Qt

$$SM(\tilde{1}, Q_t) = M_{q=0}^{L-1} h(q) \cdot REF_2(1, \mu_{Q_t}(q))$$

2. Tomar como umbral el valor de t asociado al conjunto de mayor similaridad.

Algoritmo 2: Selección del umbral óptimo.

Una vez implementado Algoritmo 1, obtendrás diferentes umbrales para la misma imagen simplemente utilizando diferentes funciones de equivalencia restringidas, automorfismos u operadores de agregación. A continuación encontrarás el esquema de un algoritmo que nos permite encontrar el umbral óptimo entre todos los umbrales de una misma imagen.

1. Para cada uno de los umbrales t (y sus correspondientes Qt) obtenidos en Algoritmo1

1.1. Calcular el conjunto H dado por

$$H(Q_t) = \left\{ (q, \mu_{H(Q_t)}(q)) \mid q = 0, \ldots, L-1 \right\} \quad dado\, por$$

$$\mu_{H(Q_t)}(q) = \begin{cases} \dfrac{m_b(t)}{L-1} & if\ q \le t \\ \dfrac{m_o(t)}{L-1} & if\ q > t \end{cases}$$

1.2. Calcular la similaridad

$$SM(Q_t, H(Q_t)) = M_{q=0}^{L-1} h(q) \cdot REF_3(\mu_{Q_t}(q), \mu_{H(Q_t)}(q))$$

2. Tomar como umbral óptimo el t* de mayor similaridad.

Algoritmo 3: Utilización de conjuntos intervalo-valorados difusos para mejorar el proceso de umbralización.

La utilización de extensiones de los conjuntos difusos (conjuntos intervalo-valorados difusos, conjuntos intuicionistas de Atanassov, …) nos permite medir y cuantificar el desconocimiento o ignorancia que tenemos a la hora de construir la función de pertenencia. El esquema del algoritmo de umbralización basado en conjuntos intervalo-valorados difusos es el siguiente:

1. FOR t = 0 TO 255 DO

1.1. Calcular L conjuntos difusos $\tilde{Q}_t$ como los construidos en Algoritmo 1

1.2. Asociar a cada conjunto $\tilde{Q}_t$ un conjunto intervalo-valorado difuso Qt dado por

$$Q_t = \left\{ (q, M_{Q_t}(q)) \mid q = 0, \ldots, L-1 \right\}$$

$$M_{Q_t}(q) = \left[\mu^{\alpha}_{\tilde{Q}}(q), \mu^{\frac{1}{\alpha}}_{\tilde{Q}}(q) \right]$$

1.3. Calcular la entropía de cada conjunto intervalo-valorado como

$$\varepsilon_F(Q_t) = \frac{1}{\sum_{q=0}^{L-1} h(q)} \sum_{q=0}^{L-1} h(q) \cdot \left(\mu^{\frac{1}{\alpha}}_{\tilde{Q}_t}(q) - \mu^{\alpha}_{\tilde{Q}_t}(q) \right)$$

2. Tomar como mejor umbral t el asociado al conjunto Qt de menor entropía intervalo-valorada.

Algoritmo 4: Construcción de conjuntos intervalo-valorados difusos a partir de funciones de ignorancia

Las principales diferencias entre Algoritmo 3 y Algoritmo 4 son las siguientes:

- Construcción de dos conjuntos difusos para cada nivel de **t** (en lugar de uno solo)
- Construcción del conjunto intervalo-valorado difuso a partir de funciones de ignorancia.

El concepto de función de ignorancia nos permite medir la ignorancia o desconocimiento que tiene el experto en la asignación de la pertenencia de un píxel a dos conjuntos difusos (uno asociado al fondo y otro asociado al objeto). Una posible expresión de función de ignorancia es $G_i(x, y) = 2 \cdot \min(1-x, 1-y)$

El esquema de Algoritmo 4 es el siguiente:

1. FOR t = 0 to L-1 DO

1.1. Calcular la media al fondo y al objeto

1.2. Calcular los conjuntos Q_{Bt} y Qot de la siguiente forma

$$Q_{Bt} = \left\{ (q, \mu_{Q_{Bt}}(q)) \mid q = 0, \ldots L-1 \right\}$$

$$\mu_{Q_{Bt}}(q) = REF\left(\frac{q}{L-1}, \frac{m_B(t)}{L-1} \right)$$

$$Q_{Ot} = \left\{ (q, \mu_{Q_{Ot}}(q)) \mid q = 0, \ldots L-1 \right\}$$

$$\mu_{Q_{Ot}}(q) = REF\left(\frac{q}{L-1}, \frac{m_O(t)}{L-1} \right)$$

1.3. Calcular el conjunto intervalo-valorado difuso dado por

$$Q_{Gt} = \left\{ (q, M_{Q_{Gt}}(q)) \mid q = 0, \ldots, L-1 \right\}$$

$$M_{Q_{Gt}}(q) = \left[G_i(0.5, 0.5) - G_i(\mu_{Q_{Bt}}(q), \mu_{Q_{Ot}}(q)), G_i(0.5, 0.5) \right]$$

1.4. Calcular la entropía del conjunto intervalo-valorado difuso

$$\varepsilon_F(Q_{Gt}) = M_{q=0}^{L-1} G_i(\mu_{Q_{Bt}}(q), \mu_{Q_{Ot}}(q))$$

2. Tomar como umbral el valor de t asociado al conjunto de menor entropía.

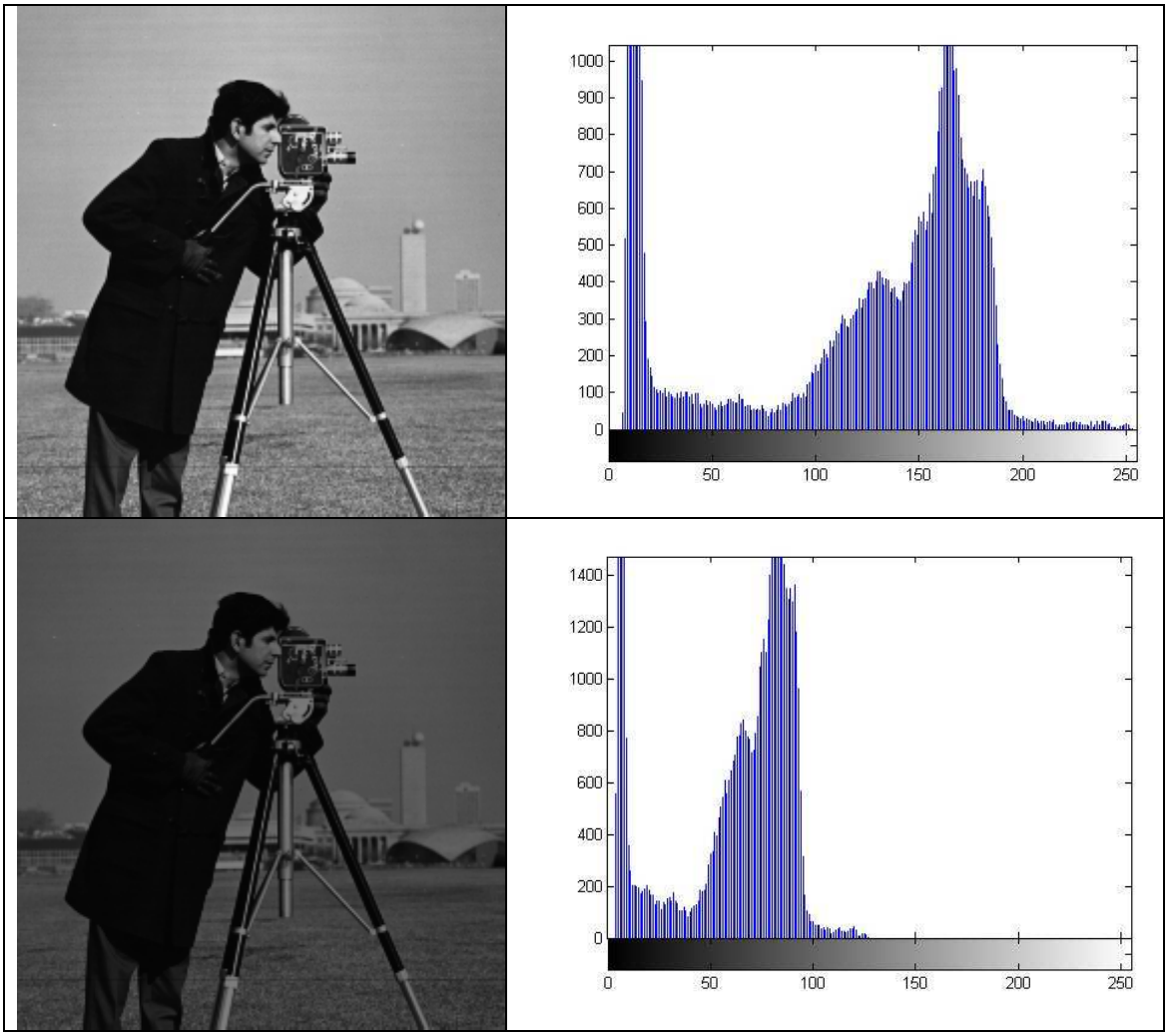
Implementa en Matlab los 4 algoritmos de umbralización propuestos en esta práctica. En MiAulario encontrarás un conjunto de imágenes con las que podrás probar tus algoritmos. Además, cada una de las imágenes viene acompaña por su segmentación ideal (realizada por un experto humano). Podrás medir la eficacia de tus algoritmos con las imágenes ideales. Una posible medida es la del porcentaje de píxeles bien clasificados.

Deberás entregar, además de los archivos Matlab con el código implementado, un documento en el que analices los algoritmos utilizados y las soluciones obtenidas. Se valorará la profundización de los comentarios, pruebas y conclusiones. El plazo límite es el Lunes 4 de abril a las 23.55h.

Algunos recordatorios que pueden ayudarte:

- Una imagen en escala de grises de dimensión NxM puede ser vista como una matriz de NxM elementos en la que cada elemento representa el nivel de intensidad de gris. Las escalas de niveles de grises suelen tomar valores entre 0 y L-1 (es decir, son escalas de L niveles de intensidad). Las imágenes con las que trabajaremos son de L = 256 niveles de gris.

- El histograma de una imagen es una gráfica en la que, para cada nivel de intensidad, se muestra el número de píxeles de la imagen que tienen esa intensidad.



- Para obtener el histograma de una imagen puedes utilizar el comando en Matlab *imhist*.

    - *Imhist(imagen)* muestra el histograma de la imagen. Si la imagen es en escala de grises, *imhist* toma por defecto 256 niveles de gris. Si la imagen es binaria, entonces tomará únicamente 2 niveles de gris.

    - *[counts, x]=imhist(imagen)* devuelve dos vectores. X es un vector que contiene cada uno de los niveles de gris (si L = 256, entonces x(1)=0, x(2)=1, …, x(256)=255. Counts devuelve el número el conteo de píxeles de la imagen. Así, counts(1) contiene el número de píxeles de intensidad 0, counts(2) el número de píxeles de intensidad 1, …, counts(256) el número de píxeles de intensidad 255.

- Para evitar errores, cuando llames a *imhist*, los elementos de tu imagen deberá ser valores entre 0 y 255 de tipo *uint8*. Si después crees necesario modificar los píxeles de la imagen, recuerda que puedes pasar los elementos a tipo *double*.