Universidad Nacional de San Agustín de Arequipa

Escuela Profesional:
Ciencias de la Computación

Tema:
Thresholding

Alumnos:
Renzo Vicente Castro
Andy Ñaca Rodríguez
Eduardo Sánchez Hincho
Luis Villanueva Flores

Curso:
Computación Gráfica

Docente:
Vicente Machaca Arceda

2020

Thresholding

1. Introducción:

En el presente documento se hablará sobre el Thresholding o método de valor de umbral, se explicarán algunos ejercicios en el leguaje Python para dar a conocer su funcionamiento.

2. ¿Qué es Thresholding:

La umbralización(Thresholding) es uno de los más importantes métodos de segmentación. El objetivo es convertir una imagen en escala de grises a una nueva con sólo dos niveles, de manera que los objetos queden separados del fondo.[1]

Figura 1: Imagen Original.

Figura 2: Imagen Binarizada con Thresholding.

3. Propiedades:

Normalmente los métodos del valor umbral "binarizan"la imagen de partida, es decir se construyen dos segmentos: el fondo de la imagen y los objetos buscados. La asignación de un pixel a uno de los dos segmentos (0 y 1) se consigue comparando su nivel de gris g con un cierto valor umbral preestablecido t (en inglés threshold). La imagen final es muy sencilla de calcular ya que para cada pixel sólo hay que realizar una comparación numérica. La regla de cálculo correspondiente es:[2]

$$T_{global}(g) = \begin{cases} 0 & \text{si } g < t \\ 1 & \text{si } g \geq t \end{cases}$$

Figura 3: Regla de cálculo.

Los métodos del valor umbral son métodos de segmentación completos, es decir cada pixel pertenece obligatoriamente a un segmento y sólo uno. Otros métodos de segmentación permiten que los segmentos se solapen. Si en la imagen existen varios objetos con una luminosidad similar, con un mismo tono de gris, todos los píxeles que los componen pertenecerán al mismo segmento. En la práctica siempre hay algún píxel que queda fuera del segmento aunque pertenezca al objeto, normalmente debido a ruidos en la imagen original. En función del valor umbral que se escoja el tamaño de los objetos irá oscilando.[2]

4. Elección del valor umbral:

El punto clave es la elección del valor umbral más adecuado. Esto se puede realizar bastante bien de manera manual, como en el ejemplo anterior, pero dado que con las variantes local y dinámica del método se deben establecer muchos valores umbral, se necesita un método que permita calcular el mejor valor umbral automáticamente. Hay un gran número de métodos para la elección del valor. Tanto si optamos por calcular el umbral manualmente o mediante un programa, el histograma sigue siendo el elemento más importante. Los máximos locales se corresponden con los objetos de la imagen. En el mejor de los casos el histograma será "bimodal", es decir, en el histograma se podrán reconocer dos picos claramente. Una técnica sencilla, pero también propensa a errores es elegir como valor umbral la media entre los dos picos del histograma. Otra técnica sencilla es elegir como umbral el valor más bajo entre los dos picos. Con este método seguramente se conseguiría una segmentación algo mejor.[2]

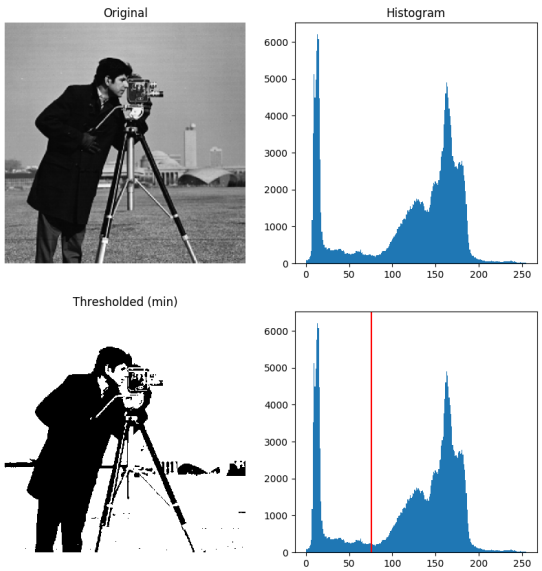

Figura 4: Elección del valor umbral.

5. Variantes:

Las más conocidas son 2:

5.1. Método global del valor umbral:

Con el método global del valor umbral se elige un valor umbral para toda la imagen. Este método es el más fácil de calcular, pero también muy sensible a las pequeñas variaciones que puedan existir en la luminosidad de la imagen. El método global, por lo tanto, sólo se utiliza para segmentar imágenes con mucho contraste. Estas imágenes puede provenir de documentos mecanografiados o de fotografías realizadas a contraluz. Si establecemos varios valores umbral se puede modificar el método de forma que tengamos más de dos segmentos. Para n segmentos se establecen (n-1) valores umbral :[2]

$$T_{global_n}(g) = \begin{cases} 0 & \text{si } g < t_1 \\ 1 & \text{si } t_1 \leq g < t_2 \\ \vdots & \vdots \\ n & \text{si } g \geq t_{n-1} \end{cases}$$

Figura 5: Método global del valor umbral.

5.2. Método local del valor umbral:

Con el método local del valor umbral se divide la imagen original en regiones y se establece un valor umbral para cada una de ellas. Es decir en cada región de la imagen Ri se establece un valor umbral ti, sin que esto afecte a la calidad de la segmentación de las otras regiones. El cálculo para cada pixel (x,y) es:[2]

$$T_{local}(x,y) = \begin{cases} 0 & \text{si } g(x,y) < t_i \\ 1 & \text{si } g(x,y) \geq t_i \end{cases} \quad \forall (x,y) \in \text{Region } R_i$$

Figura 6: Método local del valor umbral.

6. Aplicaciones:

Algunas de las aplicaciones del Thresholding son:

- Medicina:Para la detección temprana de tumores cerebrales.[3]
- Militar: En aeronaves para mejorar el reconocimiento de objetivos.[4]
- Reconocimiento óptico de texto.[2]

7. Ejercicios:

7.1. Celulas de ratones

7.1.1. Codigo:

Nuestra funcion "thresholdingrecibira el nombre de la imagen a procesar, el intervalo que sera 0 para cada pixel que este dentro y un booleano para indicar si haremos uso del suavizado gaussiano o no, esto como se vera posteriormente nos ayuda a eliminar el ruido de nuestros resultados.En las imagenes siguientes podremos notar la diferencia,la imagen de la izquierda sin suavizado gaussiano y el de la derecha si.

Listing 1: Celulas vivas eliminadas

```
import cv2
import numpy as np
from matplotlib import pyplot as plt


def thresholding(name,MinT,MaxT,Gauss):
    #Cargar Imagen
    img = cv2.imread(name,cv2.IMREAD_GRAYSCALE)
    #Aplicar filtro Gaussiano para eliminar ruido
    if(Gauss):
        img = cv2.GaussianBlur(img,(5,5),0)
    #Creacion de subplots
    fig,axs=plt.subplots(1,3)
    #Imagen Original
    axs[0].imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    #Histograma
    axs[1].hist(img.ravel(),256,[0,256])
    #Thresholding
    for y in range(0,img.shape[0]):
        for x in range(0,img.shape[1]):
            img[y,x]=255 if(MaxT>=img[y,x]>=MinT) else 0
    #Imagen resultante
    axs[2].imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))

# ELIMINANDO CELULAS VIVAS
thresholding("celulas_raton.png",145,180,False)
thresholding("celulas_raton.png",145,180,True)
thresholding("celulas_raton2.png",100,180,False)
thresholding("celulas_raton2.png",100,180,True)

# ELIMINANDO CELULAS MUERTAS
thresholding("celulas_raton.png",193,195,False)
thresholding("celulas_raton.png",193,195,True)
thresholding("celulas_raton2.png",190,191,False)
thresholding("celulas_raton2.png",190,191,True)
plt.show()
```

7.1.2. Eliminar celulas vivas

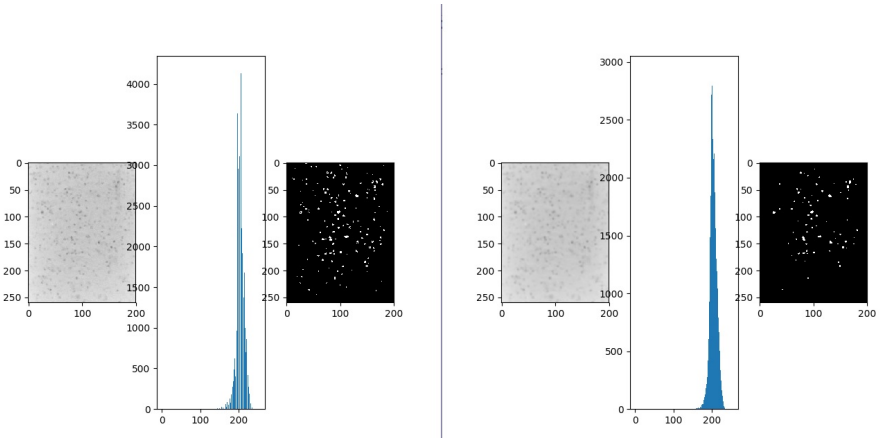

Figura 7: Resultados de la imagen 1

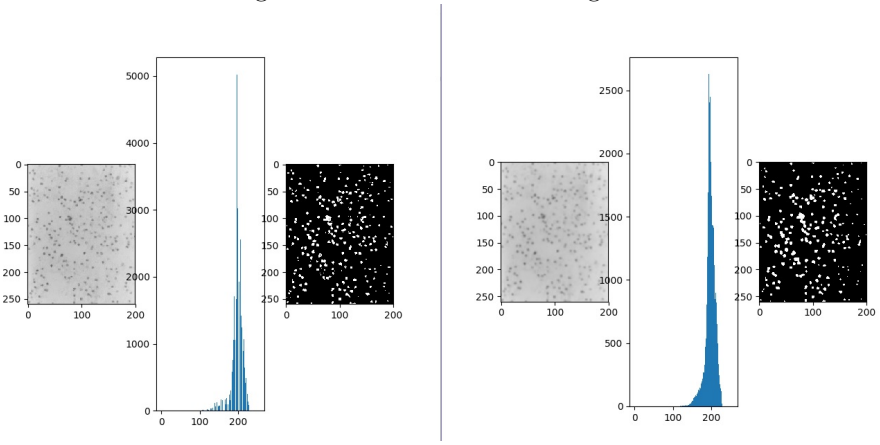

Figura 8: Resultados de la imagen 2

7.1.3. Eliminar celulas muertas

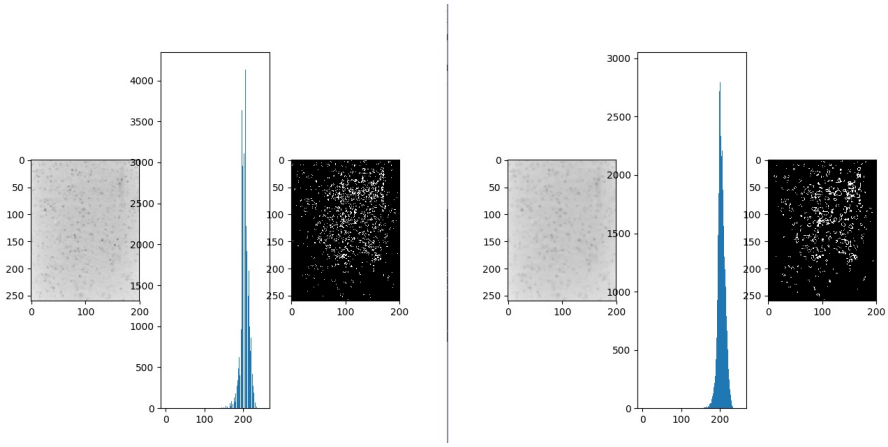

Figura 9: Resultados de la imagen 1

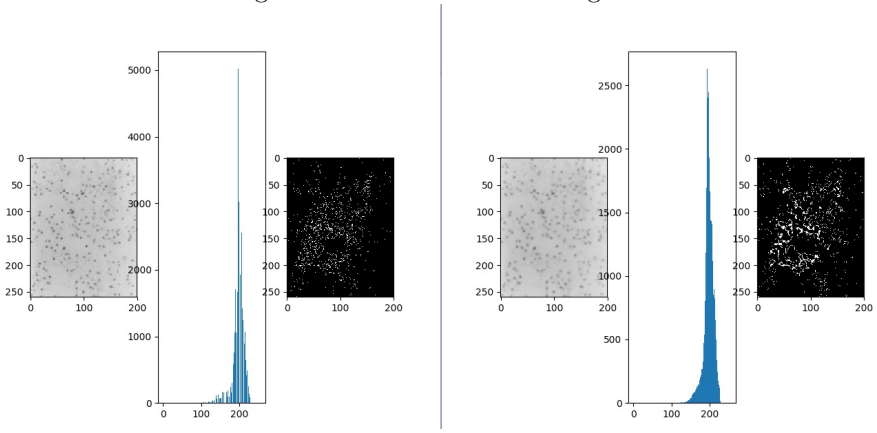

Figura 10: Resultados de la imagen 2

7.2. Campos de maiz

7.2.1. Codigo:

Lo que hacemos en esta sección de código es generar nuesto histograma a través de un for para que no los genere defrente sin tener que declarar uno por uno, una vez visto el histograma lo que hacemos es analizar nuestra imagen que es una matriz y cuando la combinación "b, g , r" salga de los límites, en este caso del amarillo, haremos que eso se pinte de color negro, así marcando los campos de trigo solicitados.

Listing 2: Campos de trigo

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
fig,axs=plt.subplots(1,3)
img = cv2.imread('trigo.png')
axs[0].imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
color = ('b','g','r')
for i, c in enumerate(color):
    hist = cv2.calcHist([img], [i], None, [256], [0, 256])
    axs[1].plot(hist, color = c)
height, width , canal= img.shape
for i in range(height):
    for j in range(width):
            if img[i][j][0]<=121 or img[i][j][1]<=144 or img[i][
                j][2]<=184:
                img[i][j][0]=0
                img[i][j][1]=0
                img[i][j][2]=0
axs[2].imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.show()
```

7.2.2. Segmentación de Campos de Trigo

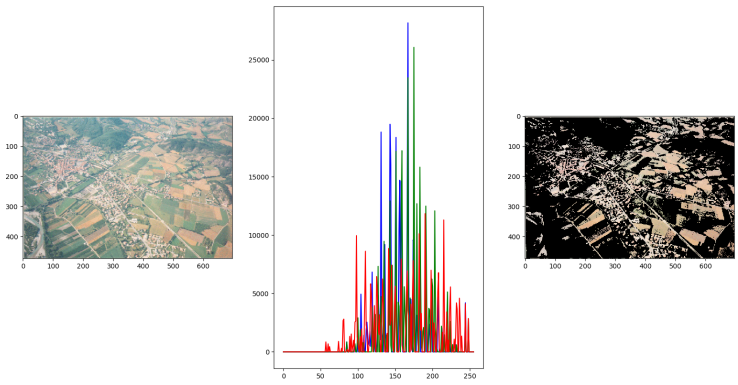

Figura 11: Resultados de la segmentación de campos de trigo

8. Conclusiones:

- El Thresholding es uno de los métodos más conocidos para la binarización de imágenes.
- El método local no es tan sensible a la luminosidad.
- El Thresholding es un método que nos permite separar imagenes para quedarnos con segmentos que si son de nuestro interés.
- Podemos usar este algoritmo para tener un preprocesamiento de imagenes,luego podemos usarlo para distintas cosas como etiquetar o contar elementos.

9. Referencias:

1. La umbralización. (2020). Retrieved 29 April 2020, from https://www.lpi.tel.uva.es/ nacho/docencia/ing_ond_1/trabajos_03_04/sonificacion/cabroa_archivos/umbralizacion.html

2. MÉTODO DEL VALOR UMBRAL In-text: ("Método del valor umbral", 2020) Your Bibliography: Método del valor umbral. (2020). Retrieved 29 April 2020, from https://es.wikipedia.org/wiki/M%C3%A9todo_del_valor_umbral

3. Mustaqeem, A., Javed, A., & Fatima, T. (2012). An Efficient Brain Tumor Detection Algorithm Using Watershed & Thresholding Based Segmentation. MECS, 34-39.

4. Region Image Segmentation Method Based on Improved Adaptive Application Thresholding in Aircraft Target Recognition Computer Programming Skills & Maintenance2009. (2020). Retrieved 29 April 2020, from http://en.cnki.com.cn/Article_en/CJFDTotal-DNBC2009S1062.htm