Maestría en Inteligencia Artificial Aplicada

TC 4033: Visión computacional para imágenes y video

Tecnológico de Monterrey

Dr. Gilberto Ochoa Ruiz

6. Otsu Thresholding

Equipo # 16

Edwin David Hernández Alejandre A01794692

Miguel Guillermo Galindo Orozco A01793695

Jorge Pedroza Rivera A01319553

Juan Carlos Alvarado Carricarte A01793486

Gerardo Aaron Castañeda Jaramillo A01137646

Table of Contents

1. Libraries
2. Single Thresholding
3. Multi Thresholding
4. Ejercicio
5. Conclusiones
6. Referencias

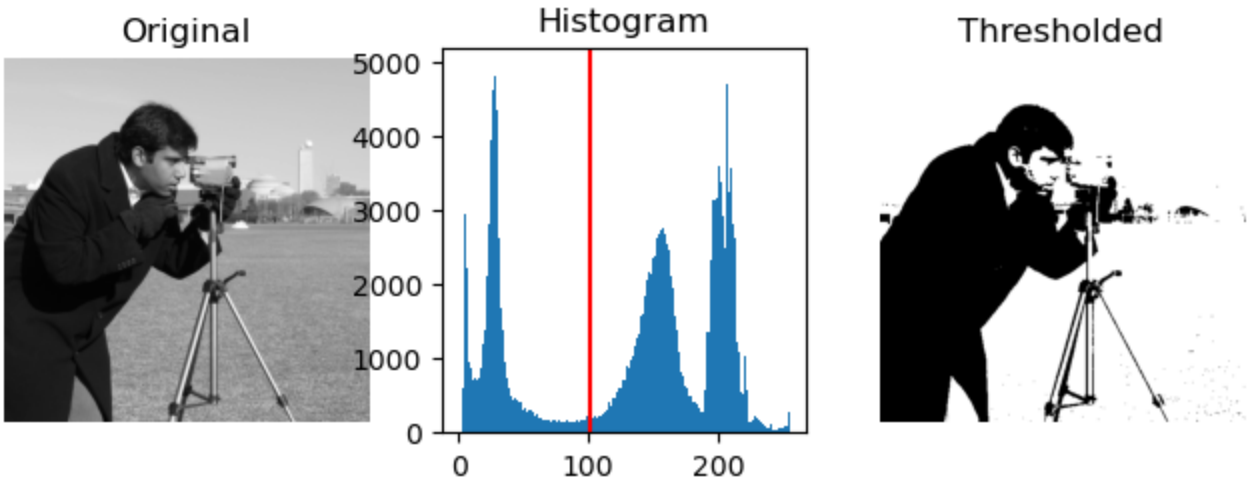
Thresholding is used to create a binary image from a grayscale image

Importing Libraries

Single Thresholding

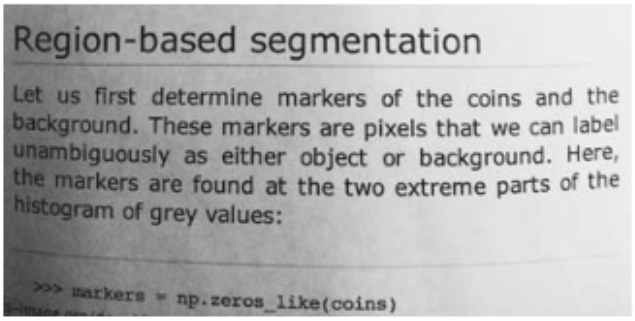
We illustrate how to apply one of these thresholding algorithms. Otsu's method [2]_ calculates an "optimal" threshold (marked by a red line in the histogram below) by maximizing the variance between two classes of pixels, which are separated by the threshold. Equivalently, this threshold minimizes the intra-class variance.

```
C:\Users\G\AppData\Local\Temp\ipykernel_26240\1528479257.py:9: MatplotlibDeprecationWarning: Auto-removal of overlapping axes is deprecated since 3.6 and will be removed two minor releases later; explicitly call ax.remove() as needed.
  ax[2] = plt.subplot(1, 3, 3, sharex=ax[0], sharey=ax[0])
```

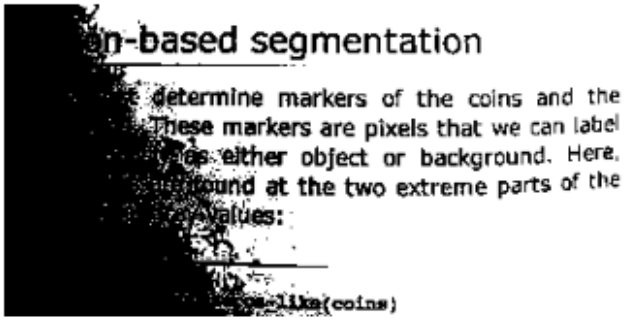


If you are not familiar with the details of the different algorithms and the underlying assumptions, it is often difficult to know which algorithm will give the best results. Therefore, Scikit-image includes a function to evaluate thresholding algorithms provided by the library. At a glance, you can select the best algorithm for your data without a deep understanding of their mechanisms.

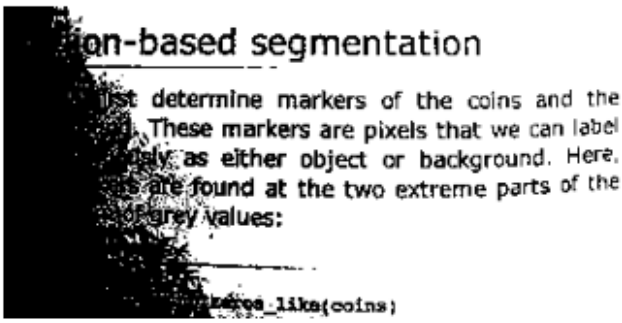
Original



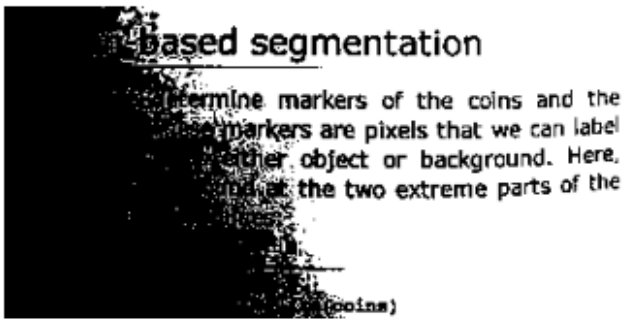
Isodata



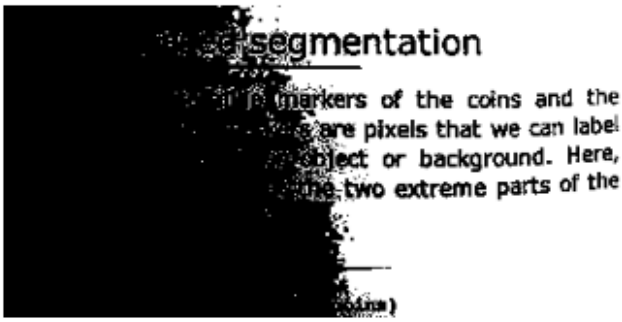
Li



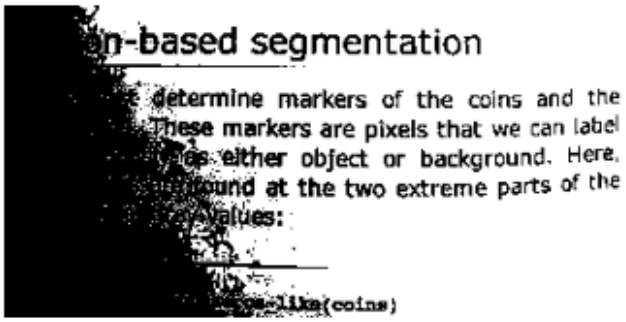
Mean



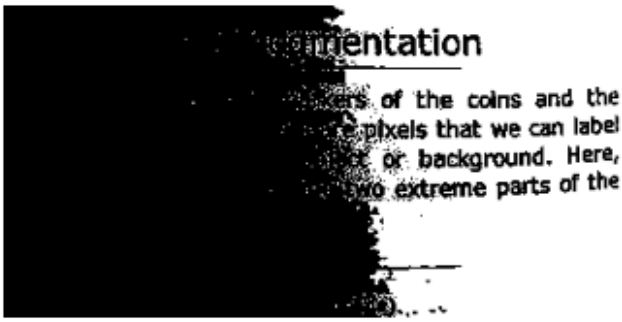
Minimum



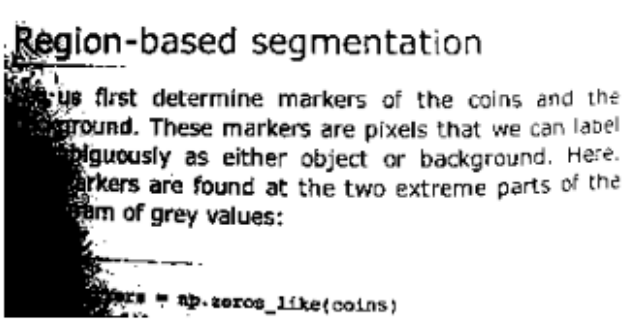
Otsu



Triangle



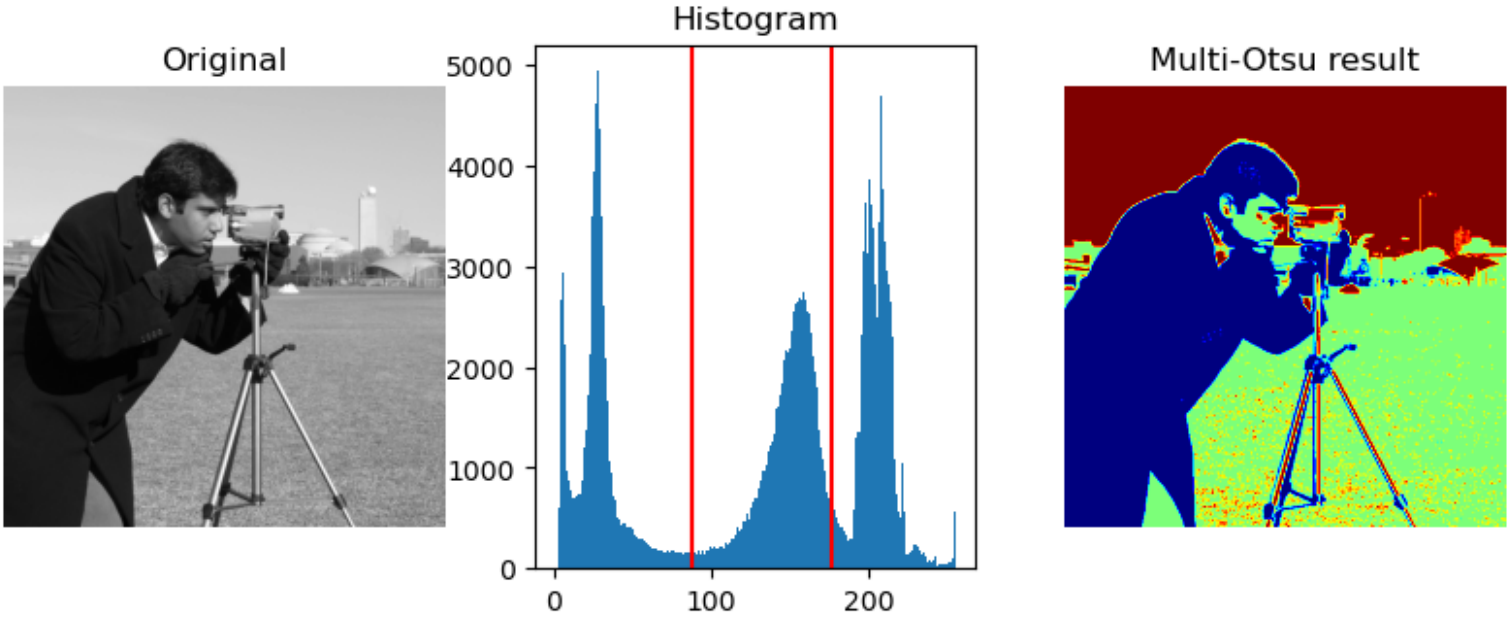
Yen



Multi Thresholding

The multi-Otsu threshold is a thresholding algorithm that is used to separate the pixels of an input image into several different classes, each one obtained according to the intensity of the gray levels within the image.

Multi-Otsu calculates several thresholds, determined by the number of desired classes. The default number of classes is 3: for obtaining three classes, the algorithm returns two threshold values. They are represented by a red line in the histogram below.



Ejercicio

Experimenta con diferentes imagenes ademas de las provistas en en Colab, identifca imagenes con diferentes backgrounds y estilos, cuales son las limitaciones de single thresholding contra el algoritmo de Otsu

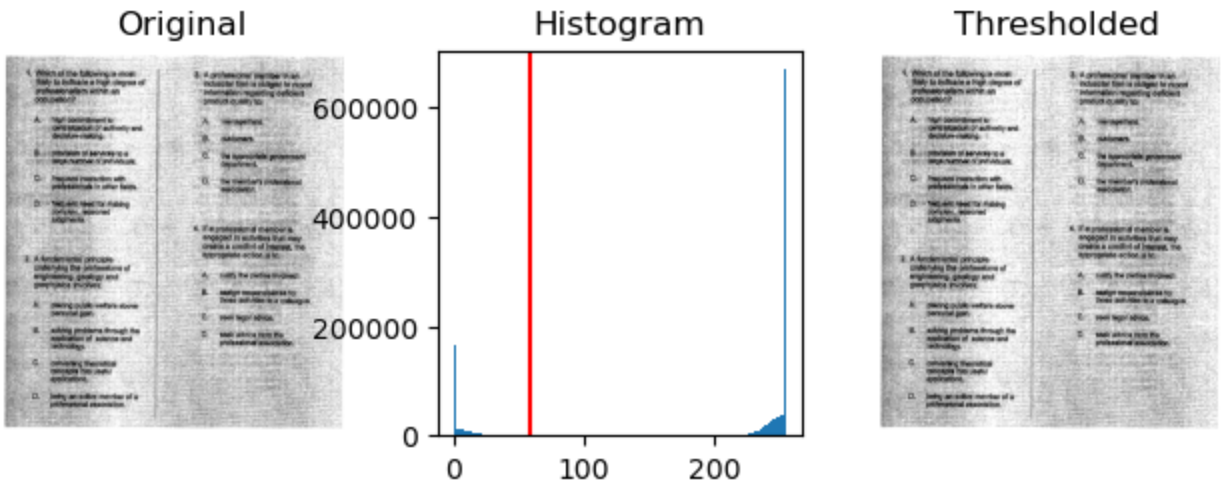
Comenzamos creando una función para aplicar el método de otsu para 1 solo threshold:

Cargamos 6 imágenes utilizadas en actividades previas:

Aplicamos median filter a un par de ellas para mejor funcionamiento:

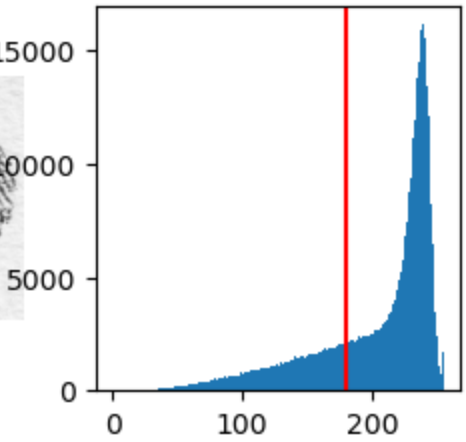
Aplicamos el método a cada una:

```
C:\Users\G\AppData\Local\Temp\ipykernel_26240\3126854094.py:9: MatplotlibDeprecationWarning: Auto-removal of overlapping axes is deprecated since 3.6 and will be removed two minor releases later; explicitly call ax.remove() as needed.
  ax[2] = plt.subplot(1, 3, 3, sharex=ax[0], sharey=ax[0])
```



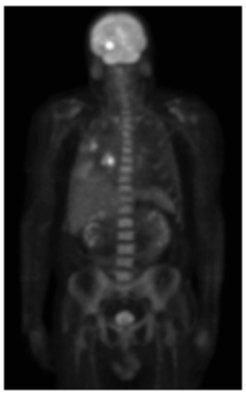
Original

Histogram

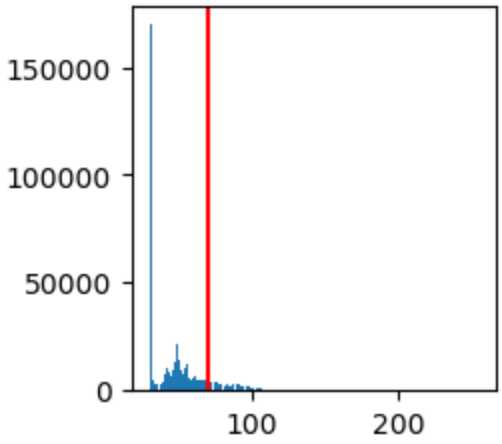


Thresholded

Original



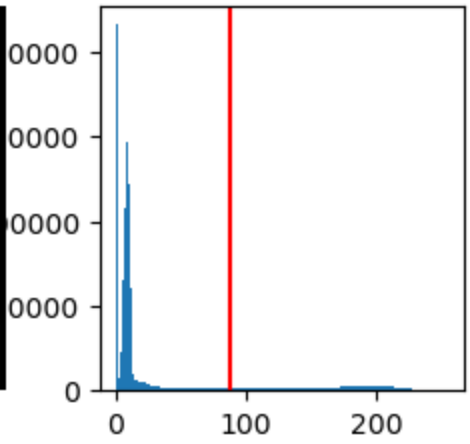
Histogram



Thresholded

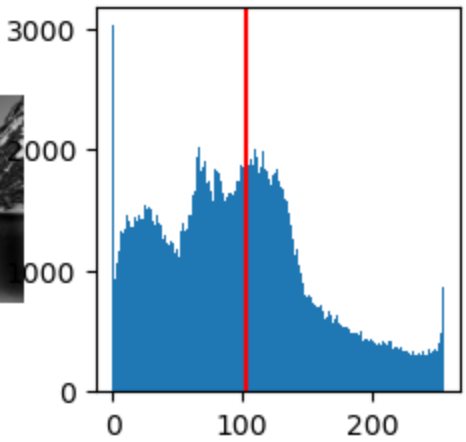
Original

Histogram



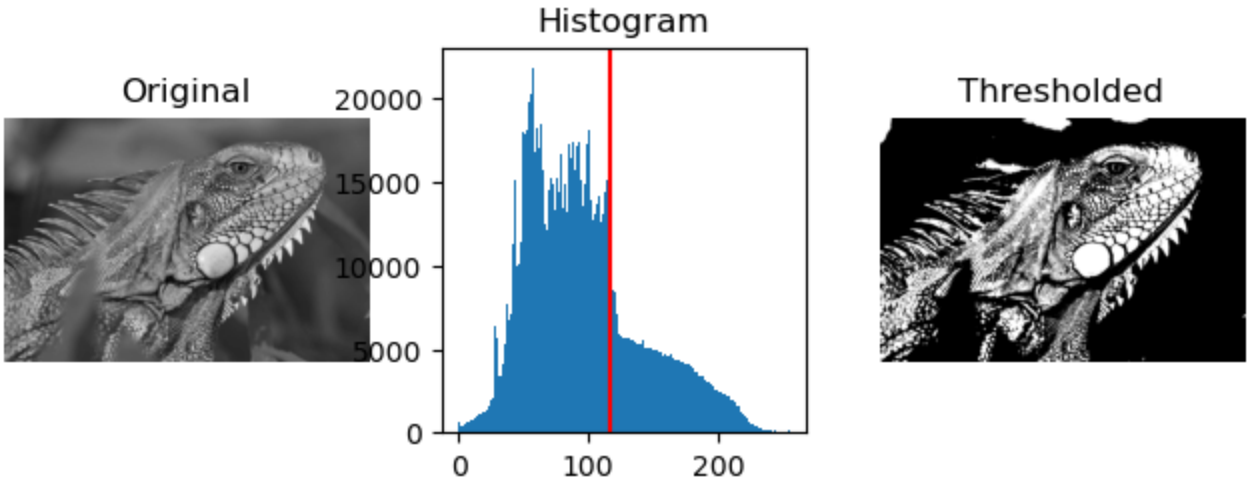
Thresholded

Histogram



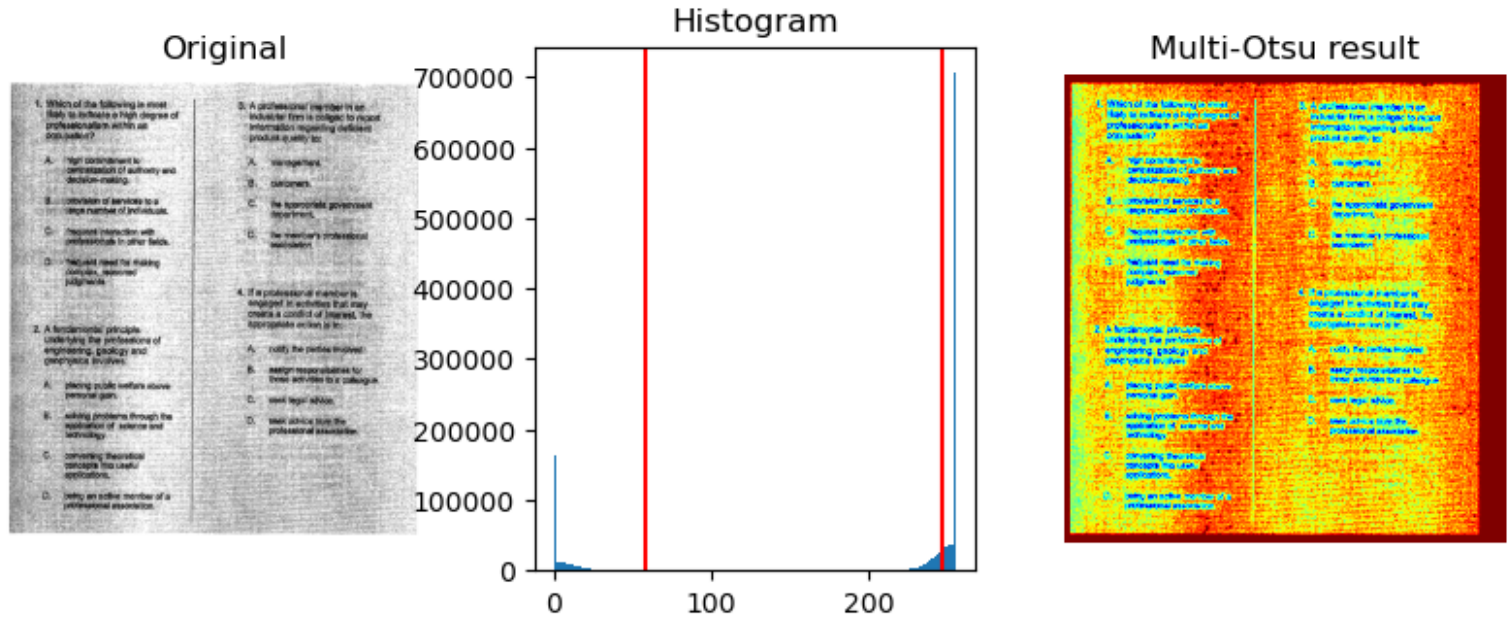
Original

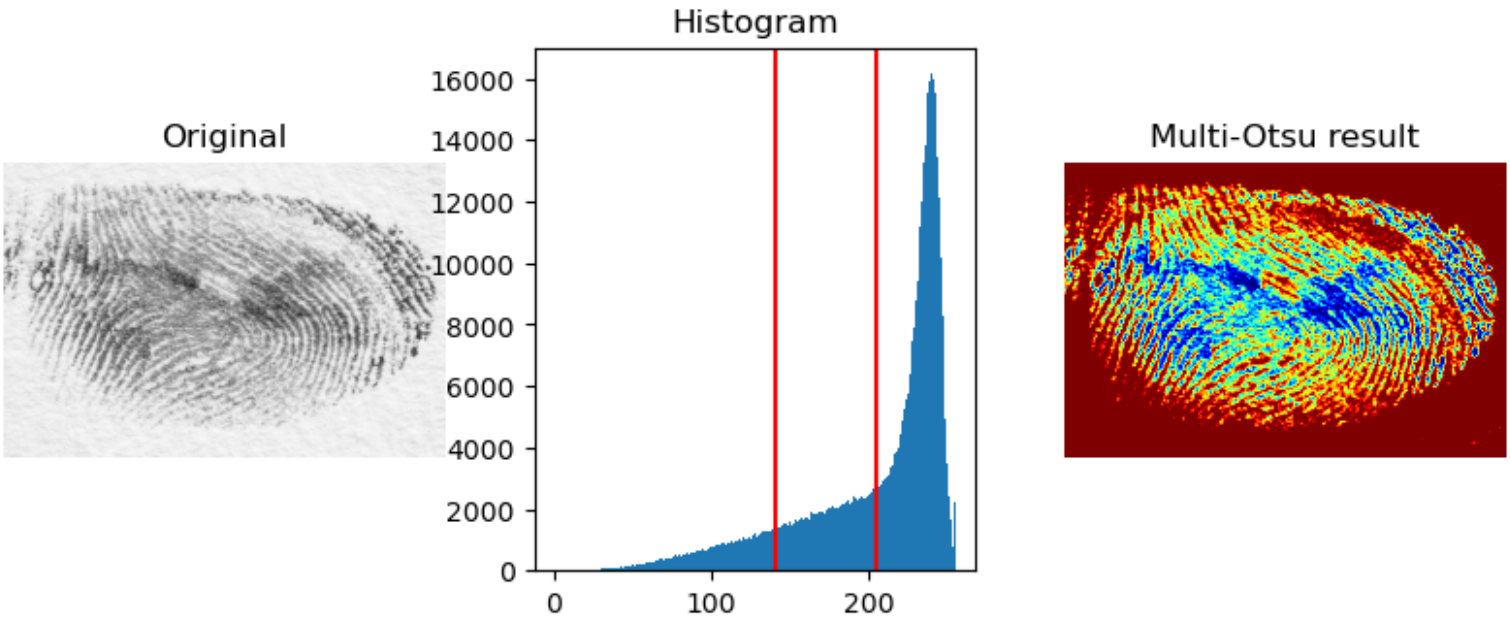
Thresholded



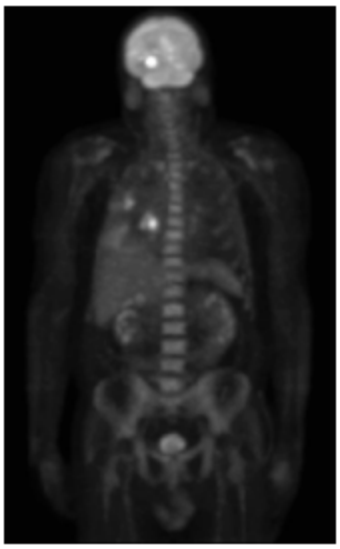
Ahora generamos una función para aplicar le método de threshold múltiple:

Lo aplicamos a las mismas imágenes:

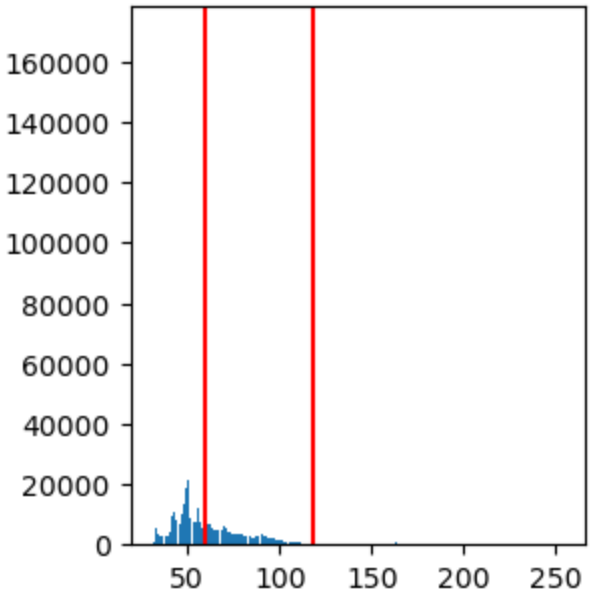




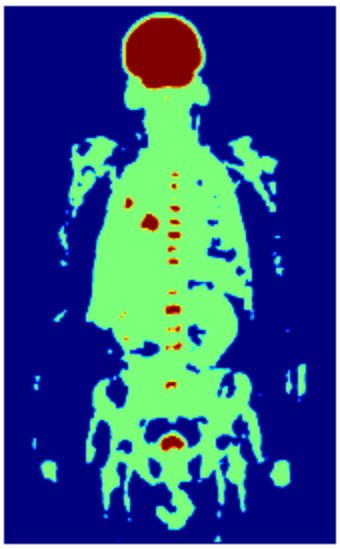
Original



Histogram

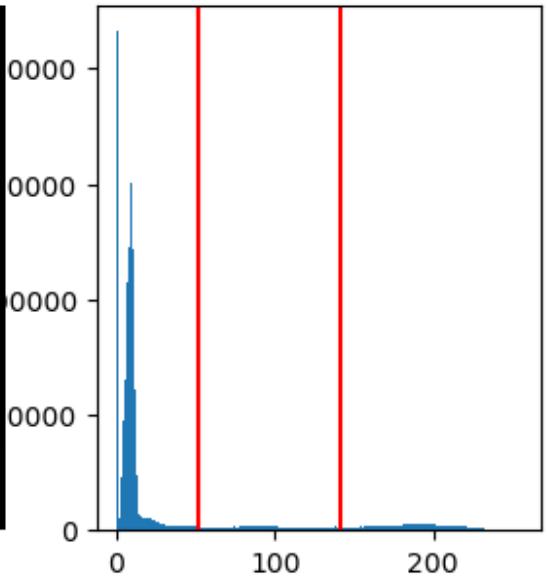


Multi-Otsu result



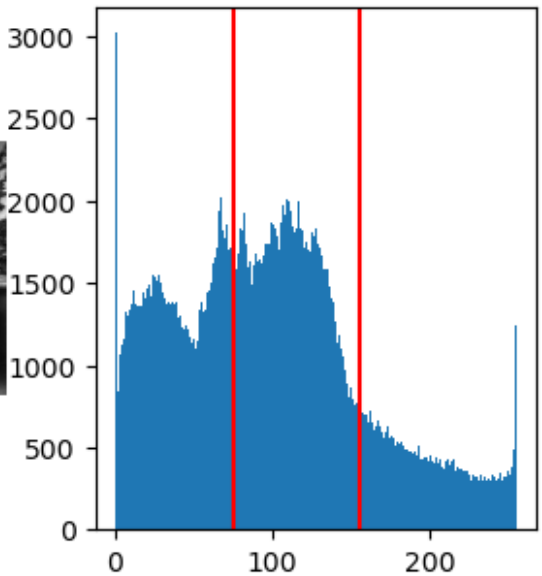
Original

Histogram



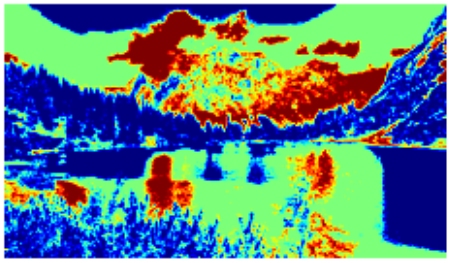
Multi-Otsu result

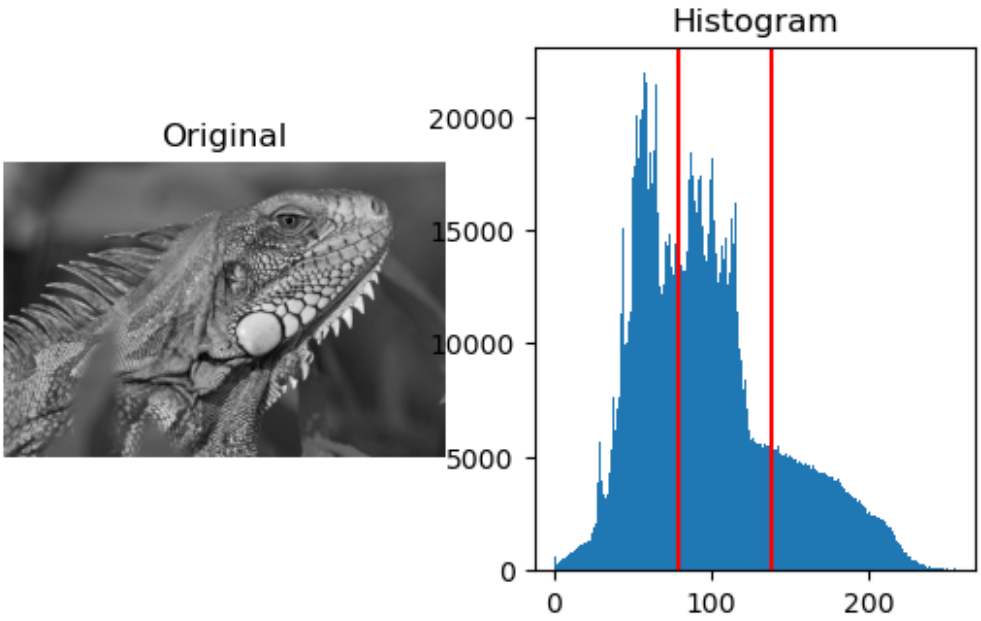
Histogram

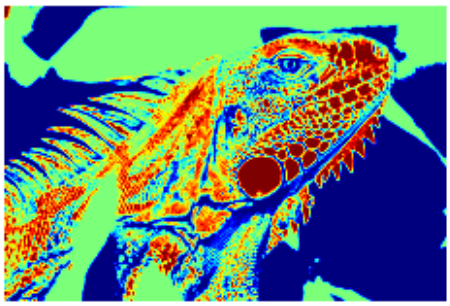


Original

Multi-Otsu result







Conclusiones

El método de otsu es una técnica de thresholding automática que se utiliza para determinar uno o varios umbrales optimo al separar los pixeles de una imagen en clases distintas. Este método es bueno en su eficiencia al hacer el cálculo, y ayuda al no tener que establecerlo anualmente. Además de que tiene buena adaptabilidad para la distribución de intensidades en iluminación y contraste. Generalmente es utilizado en aplicaciones de segmentación de imágenes, donde se requiere separar objetos de interés de un fondo.

Una nota particular con la primera imagen y con 1 solo umbral, fue que el ruido que tenia la imagen no lo permitía funcionar bien. Por lo se soluciono aplicando un median filter. Seguido, fue posible convertir la imagen a una imagen binaria donde la identificación del texto fuera mucho mas sencilla.

Al observar la diferencia entre las imágenes con 1 solo umbral, y con 3 clases, fue posible como variaba la distribución de intensidades entre las imágenes. Sin embargo, es claro notar que, dependiendo de la imagen, estas pueden requerir cierto pre-procesamiento de imagen para adaptarlas al método. Ya que es sensible al ruido.

Referencias

- baeldung. (2023). Understanding Otsu's Method for Image Segmentation. Retrieved from Baeldung: https://www.baeldung.com/cs/otsu-segmentation
- Geeks For Geeks. (2021). Mahotas – Otsu's method. Retrieved from Geeks For Geeks: https://www.geeksforgeeks.org/mahotas-otsus-method/
- Gonzalez, R., & Woods, R. (2018). Digital Image Processing. Pearson.
- Murzova, A. (2020). Otsu's Thresholding with OpenCV. Retrieved from LearnOpenCV: https://learnopencv.com/otsu-thresholding-with-opencv/
- Muthukrishnan. (2020). Otsu's method for image thresholding explained and implemented. Retrieved from MUTHU: https://muthu.co/otsus-method-for-image-thresholding-explained-and-implemented/
- OpenCV. (n.d.). Image Thresholding. Retrieved from OpenCV: https://docs.opencv.org/4.x/d7/d4d/tutorial_py_thresholding.html