Primera versión del informe

Jose Miguel Becerra Casierra-2043246
Univalle,Ingeniería de Sistemas, Abril 2024.

1. INTRODUCCIÓN.

El procesamiento de imágenes es un campo multidisciplinario que abarca técnicas y algoritmos para adquirir, mejorar, analizar y comprender imágenes digitales. Con el avance de la tecnología, el procesamiento de imágenes se ha convertido en un componente crucial en una amplia gama de aplicaciones, desde la medicina y la astronomía hasta la seguridad y el entretenimiento. Este informe contiene dos partes, la primera es la segmentación de una imagen médica de extensión .nii con diferentes algoritmos, y la segunda es sobre diferentes algoritmos de procesamiento de esa imagen médica.

2. SEGMENTACIÓN.

En la segmentación se trataron cuatro algoritmos de segmentación, estos son:

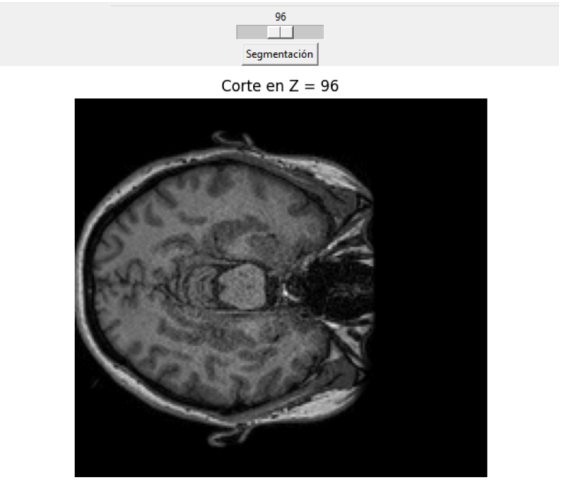
- Umbral
- Isodata
- Crecimiento de regiones
- K-means
- Mean filter

2.1. Umbral

La segmentación por umbralización es un proceso que implica dividir una imagen en regiones o segmentos basados en el valor de intensidad de los píxeles. Se establece un umbral, y los píxeles se asignan a diferentes segmentos según si su valor de intensidad está por encima o por debajo de ese umbral. Este método es útil para separar objetos de interés del fondo en una imagen, facilitando su análisis y procesamiento posterior.

La implementación de este algoritmo en python se realizó de la siguiente manera:

- Toma la capa actual de la visualización.
- Por cada valor de la imagen lo compara con el umbral, si es mayor que el umbral le da el valor de 255, y si es menor que este, le da el valor de 0.

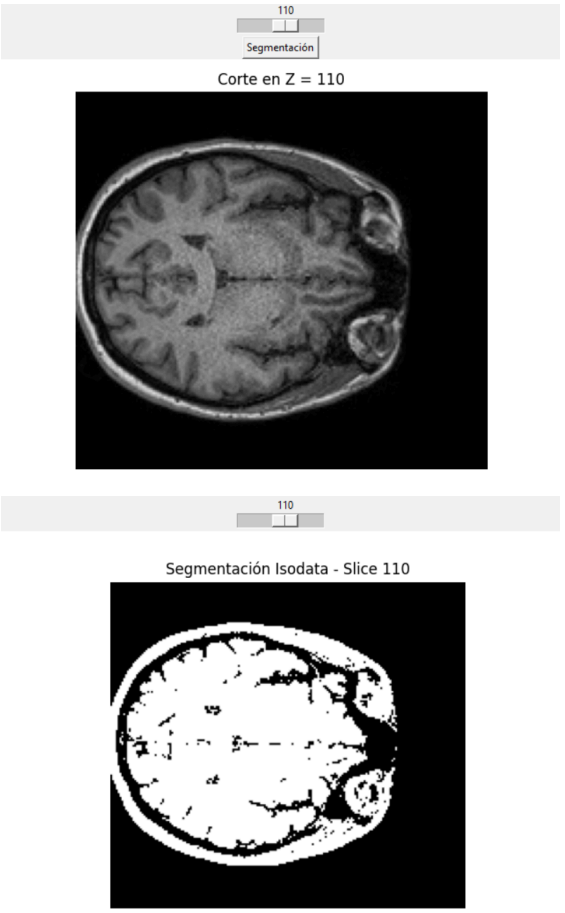


2.2. Isodata

Isodata es un método de segmentación de imágenes que busca dividir una imagen en regiones o clases distintas basadas en el nivel de gris de los píxeles. El algoritmo ajusta iterativamente los umbrales de la imagen según estadísticas como la media y la desviación estándar de los niveles de gris de los píxeles. En cada iteración, se calculan nuevos umbrales y se reasignan los píxeles a las clases correspondientes. Este proceso se repite hasta que los umbrales convergen o hasta que se alcanza un número máximo de iteraciones.

La implementación de este algoritmo en python se realizó de la siguiente manera:

- Se establece el umbral inicial (initial_threshold) y la tolerancia.
- Se inicia la iteración en un ciclo while donde se calcula un nuevo umbral promediando los valores de el el fondo y el primer plano.
- Luego se verifica si la diferencia entre el nuevo umbral y el umbral antiguo es mejor que la tolerancia establecida, en el caso que sea menor, eso significa que el algoritmo converge y rompe el bucle.
- Finalmente teniendo el umbral, este es aplicado a la imagen final.



2.3. Crecimiento de regiones

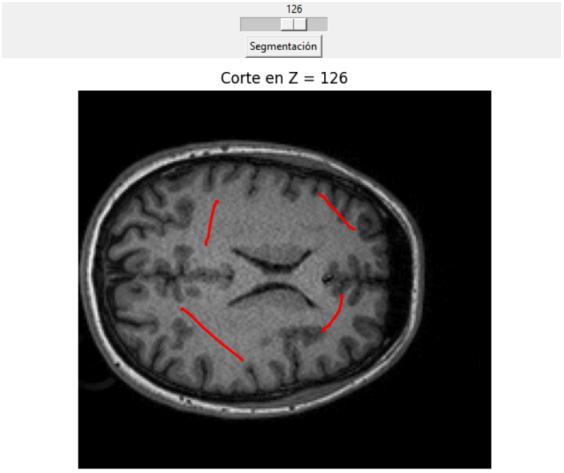
La segmentación de crecimiento de regiones es un método utilizado en el procesamiento de imágenes para dividir una imagen en regiones o áreas significativas basadas en ciertos criterios, como la similitud de píxeles o la conectividad.

La implementación de este algoritmo en python se realizó de la siguiente manera:

- Para la tomar los puntos iniciales, se basa en trazos, se realizan trazos en la imagen y

todos esos trazos se agregan a una cola de ejecución.

- Se establece un umbral de intensidad el cual sirve para la comparación con los vecinos.
- Se itera sobre los valores almacenados a la cola de ejecución, por cada valor se analizan los 8 vecinos cercanos.
- Si la diferencia de intensidad es menor que el umbral, se agrega ese vecino a la cola de ejecución.
- Finalmente todos esos valores que cumplen con la intensidad son marcados de un color distintivo.

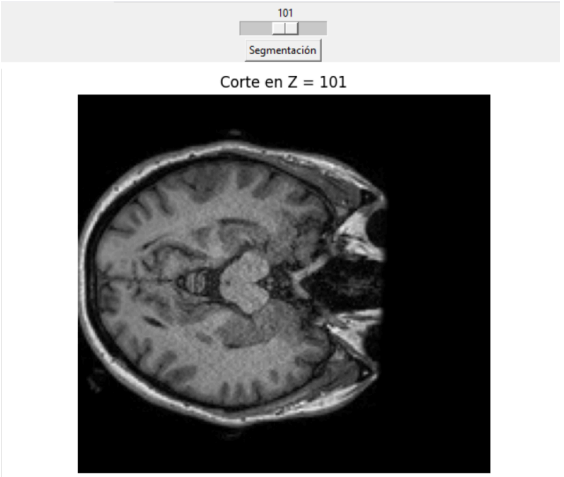


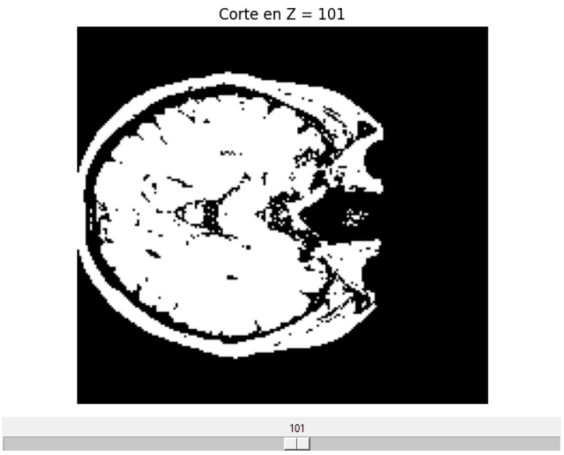
2.3. K-Means

La segmentación K-means es un método de agrupamiento no supervisado que divide una imagen en “k” grupos o segmentos según las características de intensidad de los píxeles. Este algoritmo comienza asignando aleatoriamente los píxeles a los grupos y calculando los centroides de cada uno. Luego, iterativamente, reajusta los centroides y reasigna los píxeles al grupo más cercano hasta que se alcanza un criterio de convergencia.

La implementación de este algoritmo en python se realizó de la siguiente manera:

- Primero se escogen aleatoriamente los centroides.
- Se itera sobre los valores para conocer a qué grupo se encuentra mas cerca y se le asigna.
- Se recalculan los centroides recorriendo cada grupo y sacando la media.
- Si los centroides no cambian, sale del bucle.
- Para cada píxel en la imagen original, se selecciona el color del centroide al que fue asignado.





3. Procesamiento.

En el procesamiento se trataron cuatro algoritmos, estos son:

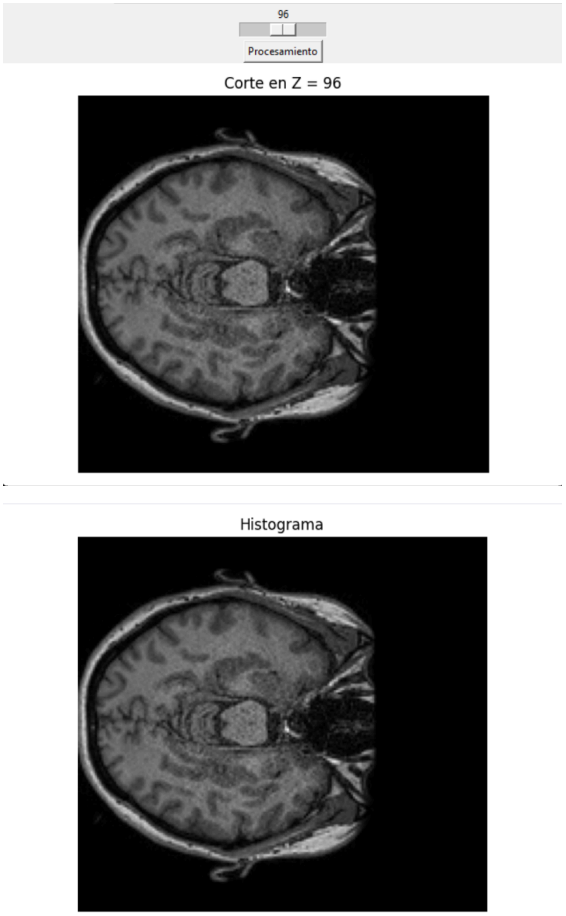
- Histogram matching
- White stripe
- Intensity rescaller
- Z-Score

3.1. Histogram matching

El histogram matching es una técnica común en el procesamiento de imágenes utilizada para mejorar el contraste de una imagen. Consiste en ajustar la distribución de intensidades de píxeles en una imagen para que coincida con una distribución de referencia deseada. Esta técnica tiene dos partes, el *training* el cual es donde se analiza la imagen que se va a tomar de referencia, y el *transforming* el cual es donde se le aplica la lógica en la imagen objetivo.

La implementación de esta técnica en python se realizó de la siguiente manera:

- Primero se inicia con el training. Se toma la imagen de referencia y se definen los percentiles de intensidad que tiene.
- En cada tramo(percentil) se obtiene la función lineal $fx = mx + b$.
- Ya se pasa a el transforming. Se definen los percentiles de intensidad en la imagen objetivo y se itera sobre cada valor de esta.
- En cada valor se obtiene en qué percentil de esta está, con esa información se pasa a los percentiles de la imagen de referencia para obtener la fórmula equivalente.
- Ya con la fórmula, se realiza la operación con el valor que se tiene y ese es el nuevo valor.



El resultado de aplicar esta técnica a la imagen original no se es muy notorio en el resultado ya que la imagen base

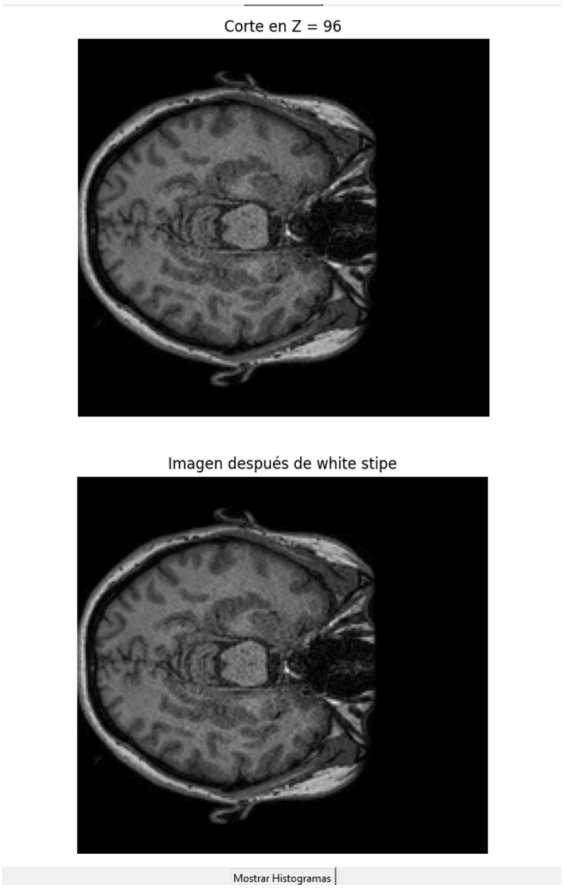
y la imagen de referencia ambas son imágenes clínicas en escala de grises.

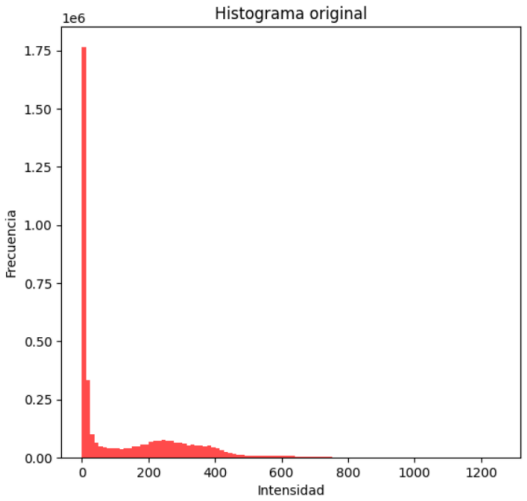
3.2. White stripe

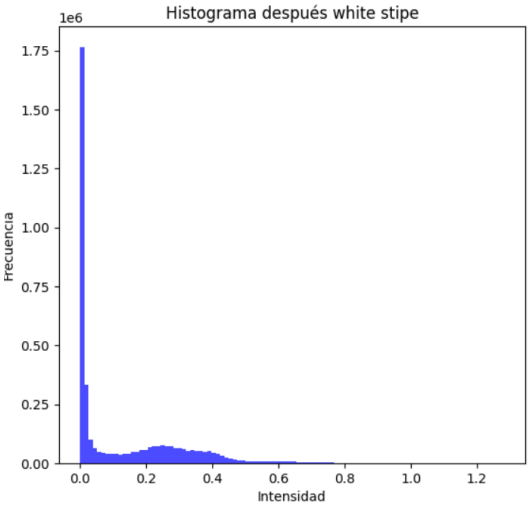
Esta técnica busca el último pico de la distribución y con ese valor estandarizar la imagen.

La implementación de esta técnica en python se realizó de la siguiente manera:

- Se realiza el histograma de la imagen dada.
- Con la función *peaks_f* se encuentran todos los picos que tiene el histograma.
- Se aplica la formula: $I* = \frac{I}{ws(I)}$ con cada valor, en donde cada valor se divide entre el valor del último pico.





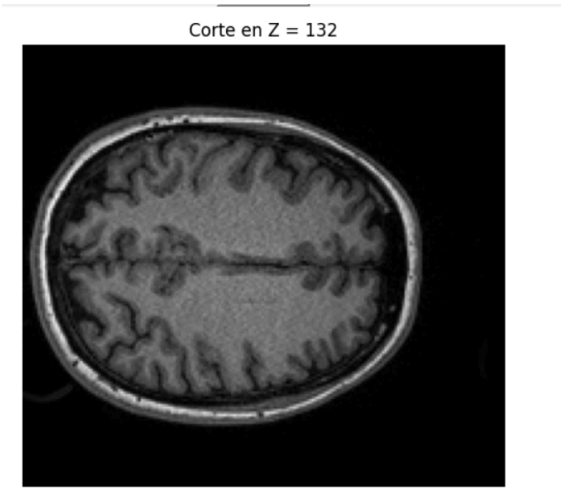


3.3. Intensity rescaller

Este método busca la intensidad mayor y menor, y con estos dos valores se estandariza cada valor de la imagen.

La implementación de este método en python se realizó de la siguiente manera:

- Se toman todos los datos de la imagen para encontrar la intensidad mayor y la intensidad menor.
- A cada valor se le aplica la fórmula $I* = \frac{I - min(I)}{max(I) - min(I)}$ para conocer el valor final.

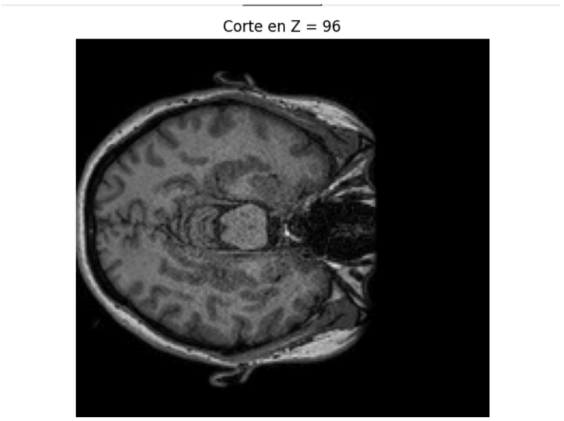


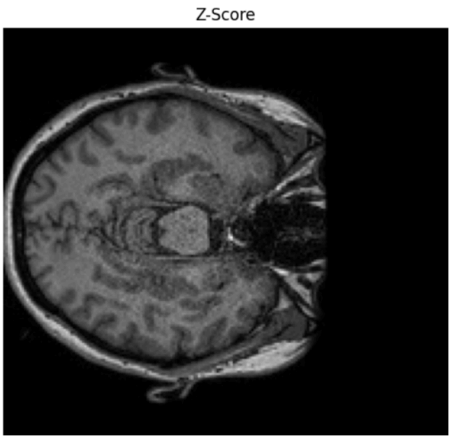
3.4. Z-Score

El Z-Score es una medida estadística utilizada en procesamiento de imágenes para la normalización de intensidades. Se calcula restando la media de intensidad de la imagen y dividiendo por la desviación estándar de la intensidad en toda la imagen.

La implementación de este método en python se realizó de la siguiente manera:

- Se obtiene la media y la desviación estándar de la imagen.
- A casa valor se la imagen se la aplica la fórmula $I^* = \frac{I - \mu(I)}{\sigma}$





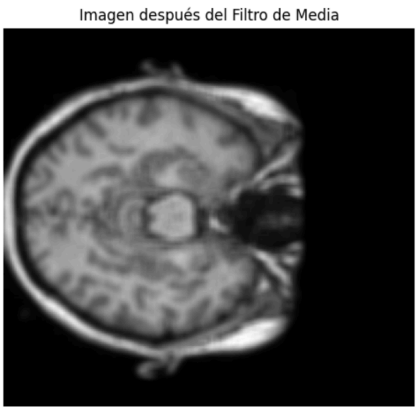
3.5. Mean filter

El algoritmo mean filter es usado para suavizar imágenes y reducir el ruido. Usa la media de todos sus vecinos.

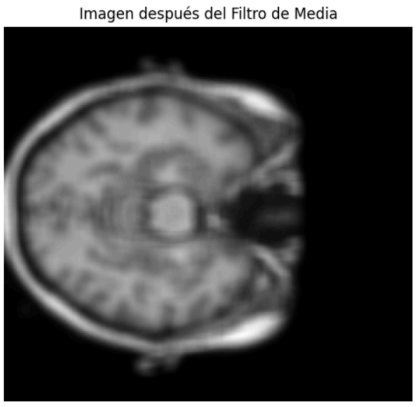
La implementación de este método en python se realizó de la siguiente manera:

- Se recorre la imagen y cada valor.
- Dependiendo del kernel escogido se recorren los vecinos en un cuadrado, este puede ser 3x3, 5x5, 10x10
- Se toma cada valor, se suman, se calcula la media y lo asigna.

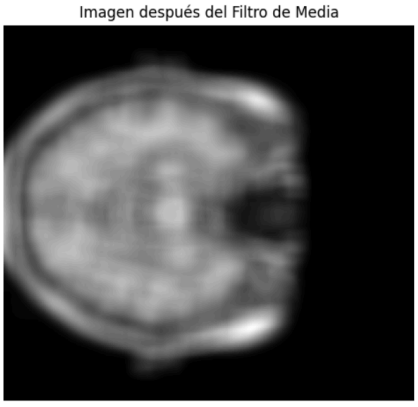
3x3



5x5



10x10



3.5. Median filter

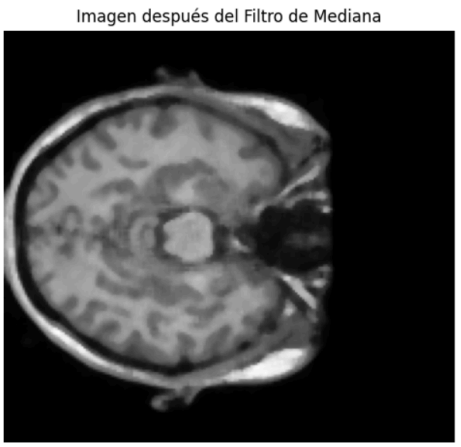
El algoritmo mean filter es usado para reducir el ruido y preservar los bordes. Usa la mediana de todos sus vecinos.

La implementación de este método en python se realizó de la siguiente manera:

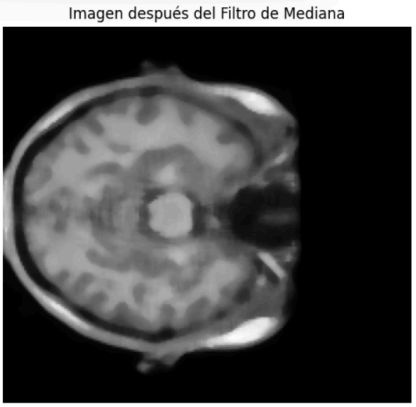
- Se recorre la imagen y cada valor.
- Dependiendo del kernel escogido se recorren los vecinos en un cuadrado, este puede ser 3x3, 5x5, 10x10
- Se toman los valores, se calcula la media y lo asigna.

Quitamos el promedio, dividimos entre la desviación estándar.

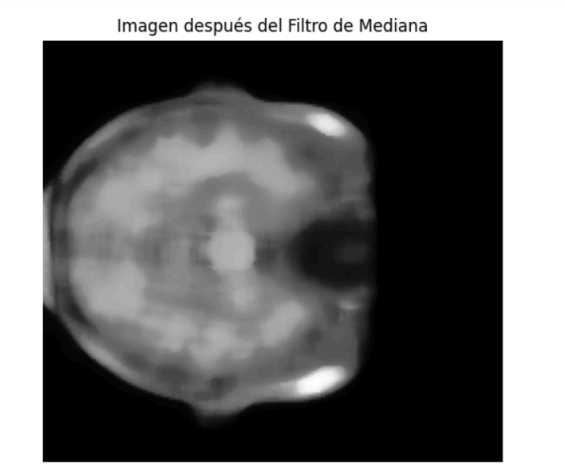
3x3



5x5



10x10



Los métodos como histogram matching, mean filter y median filter, toma más tiempo en su ejecución ya que su implementación está realizada con for anidados.