**Preprocessing of brain tumor images using thresholding can be performed in several steps as follows:**

1. Load the brain tumor image: The first step is to load the brain tumor image into the programming environment. This can be done using a variety of libraries depending on the programming language used.
2. Convert the image to grayscale: Since thresholding is typically performed on grayscale images, it is necessary to convert the brain tumor image to grayscale if it is not already in grayscale format.
3. Apply Gaussian smoothing: Before applying thresholding, it is often helpful to apply a Gaussian smoothing filter to the image to reduce noise and improve the overall quality of the image.
4. Apply thresholding: Thresholding can be applied to the image using a variety of thresholding methods, such as Otsu's method, adaptive thresholding, or manual thresholding. The threshold value can be determined based on the intensity distribution of the image.
5. Perform morphological operations: After thresholding, morphological operations such as erosion and dilation can be performed to remove small artifacts and smooth the edges of the tumor.
6. Perform feature extraction: Once the preprocessing steps are complete, features can be extracted from the image, such as the size and shape of the tumor, to aid in diagnosis and treatment planning.

**1. Load the brain tumor image:**

The process of loading a brain tumor image into a programming environment depends on the programming language and libraries being used. We used openCV.

**Note: What is openCV library and why we use it?**

OpenCV (Open Source Computer Vision Library) is an open-source computer vision and machine learning software library that provides a collection of algorithms and tools to help developers create applications that can interpret and understand visual data from images and videos.

**There are several reasons why OpenCV is a popular choice for computer vision applications:**

1. OpenCV has a vast library of pre-built algorithms and functions that can be used for image and video processing, object detection, face recognition, and more.
2. OpenCV supports a wide range of programming languages, including Python, C++, and Java.
3. OpenCV can be used with different hardware, including CPUs and GPUs.
4. OpenCV is open source, which means that it is free to use and modify.
5. OpenCV has a large community of developers who contribute to the library, making it a constantly evolving and improving tool for computer vision applications.

**2. Convert the image to grayscale (in case it is a 3-channel colored image):**

The number of channels in an image represents the number of color components per pixel. In OpenCV, color images are represented as 3-channel images in the BGR (Blue-Green-Red) format, where each pixel has three values that represent the intensity of blue, green, and red color components. Grayscale images, on the other hand, have only one channel that represents the intensity of the pixel.

* To convert the brain tumor image to grayscale format, we can use OpenCV's cvtColor() function.
* The cvtColor() function to convert the image from the default (3-channel images) BGR (blue-green-red) color format to grayscale (one channel images).

**Note: Why we resize the image to 256 px?**

Resizing the image to 256 pixels is a common practice in image processing and computer vision tasks to standardize the size of the input image.

**This helps us to:**

1. Simplify the computational requirements and reduce the memory usage.
2. Resizing the image to a smaller size can help to reduce the noise and smooth out the image, making it easier to process and analyze.
3. Helps to standardize the input size for different patients and different MRI machines.

**3. Apply Gaussian smoothing:**

* Gaussian smoothing filter is often applied to an image prior to thresholding to reduce the effect of noise and small variations in the image.
* The filter works by convolving the image with a Gaussian kernel, which has a bell-shaped curve that gives more weight to the central pixel and less weight to the surrounding pixels.
* We used openCV.
* This process helps to:

1. Smooth out small variations in the image and reduce the noise level, which can improve the accuracy of the thresholding process.
2. Remove small details in the image that may not be relevant to the analysis, which can further simplify the image and make the thresholding process more efficient.

**Note: What is the central pixels and the less central pixels?**

In the context of the Gaussian filter, the central pixels refer to the pixel being processed by the filter, which is at the center of the kernel. The less central pixels refer to the surrounding pixels, which are weighted less heavily than the central pixel.

In the Gaussian kernel, the weights decrease as the distance from the central pixel increases. The central pixel has the highest weight, and the weight decreases as the distance from the center increases. This means that the central pixel has the most influence on the filtered value, while the surrounding pixels have less influence. This weighting scheme helps to smooth out the image and reduce noise while preserving the edges and other important features in the image.

**why we used kernel\_size=5?**

In Gaussian smoothing, the kernel size determines the size of the neighborhood around each pixel that is considered in the smoothing operation. The kernel size determines how much smoothing is applied to the image, and larger kernel sizes result in more smoothing.

The kernel size should be an odd number, so that the center of the kernel can be placed on the pixel being smoothed. A larger kernel size will increase the computational cost of the smoothing operation.

We used a kernel size of 5, which is a common choice for Gaussian smoothing. This kernel size is large enough to smooth out some of the noise in the image, while still preserving some of the fine details in the image.

**What is sigma and why it is =0?**

Sigma (σ) is a parameter that determines the standard deviation of the Gaussian function used for blurring the image. The larger the sigma value, the more the image will be blurred. A sigma value of 0 is not a valid input because it would result in a zero-sized Gaussian kernel.

In the provided code snippet, sigma is set to 0 because it is not desired to apply any additional smoothing on top of the Gaussian filter, since the purpose is to enhance the edges and features in the image, rather than blur them. However, the function can still work properly if a small non-zero value is used for sigma.

**4. Apply thresholding: using OpenCV:**

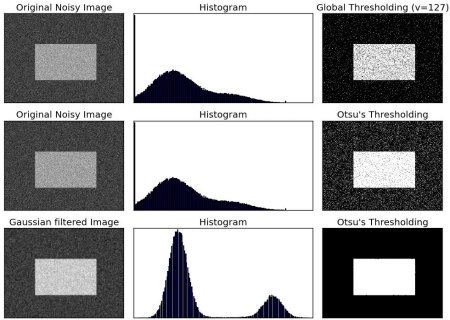
Before segmenting a brain tumor in an MRI, we can use a variety of thresholding techniques, including:

1. Otsu's method: This is a global thresholding technique that automatically computes an optimal threshold value based on the intensity distribution of the image.
2. Adaptive thresholding: This method is useful when the illumination in the image is not consistent. It divides the image into small regions and applies a thresholding operation on each region based on its local illumination.
3. Binary thresholding: This method converts the grayscale image to a binary image, where pixels with intensity values above the threshold are set to 1 (white) and pixels below the threshold are set to 0 (black).
4. Simple thresholding: This method sets all pixels with intensity values above a threshold to a maximum value and all other pixels to 0.

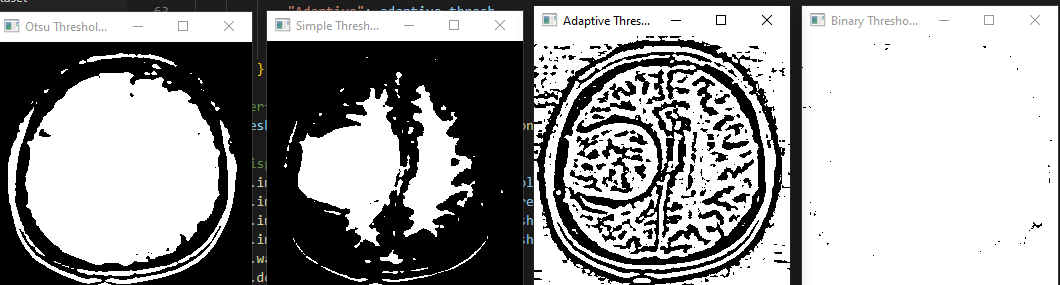
The choice of thresholding technique depends on the characteristics of the image and the specific requirements of the segmentation task.

The intensity threshold to use depends on the specific image and the desired segmentation result. It is typically chosen based on the intensity distribution of the image, where the threshold value separates the pixels that belong to the object of interest (in this case, the tumor) from the background.

A good starting point for choosing an intensity threshold is to visually inspect the image and identify the intensity range that corresponds to the tumor region. Alternatively, thresholding methods like Otsu's method or adaptive thresholding can be used to automatically select the threshold value based on the image histogram.



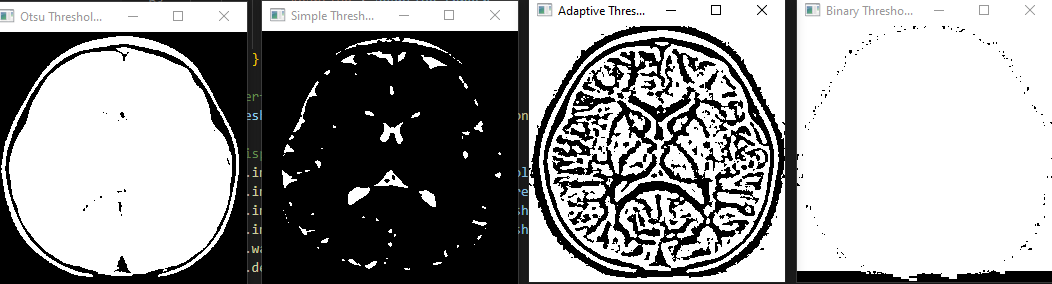
Experimentation and evaluation are usually necessary to determine the best thresholding technique for a particular application. So, we did a simple program to see what we should choose between them:



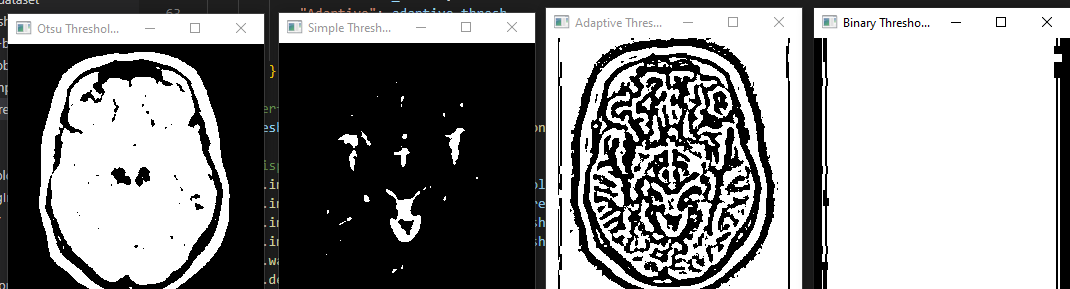
A brain with a tumor, under Ots (global), simple, adaptive, and binary thresholding methods.



A brain with a tumor, under Ots (global), simple, adaptive, and binary thresholding methods.



A brain without a tumor, under Ots (global), simple, adaptive, and binary thresholding methods.

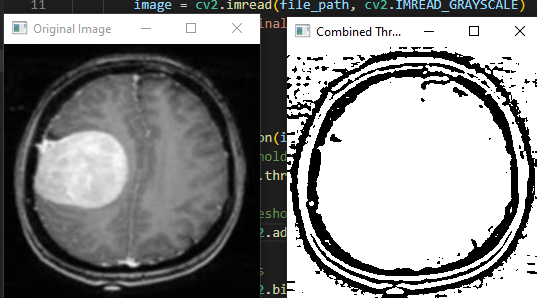


A brain without a tumor, under Ots (global), simple, adaptive, and binary thresholding methods.

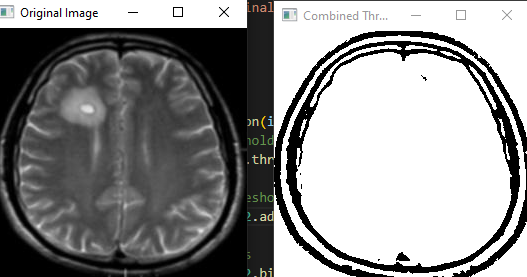
It looks like the adaptive thresholding method is the best one. Then, the adaptive threshold is the chosen one.

It is not necessary to use multiple thresholding techniques before segmentation, but it can be helpful in certain cases. But ChatGPT recommended: For brain tumor MRI segmentation, I would recommend using a combination of Otsu's thresholding and adaptive thresholding. Otsu's thresholding is a good choice for images with bimodal intensity distributions, which is often the case for MRI images, while adaptive thresholding can handle local variations in image intensity. Together, these techniques can help to accurately segment the tumor region while minimizing false positives and false negatives.

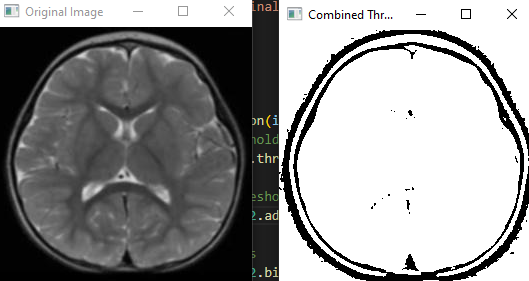
So we did a program to test the product of their combination:



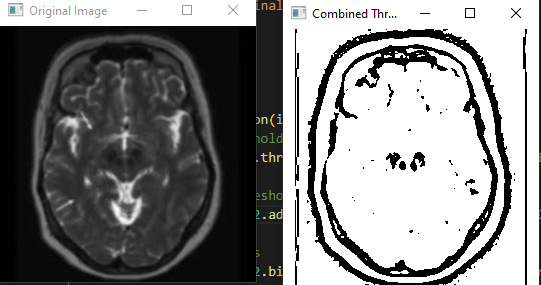
A brain with a tumor, under the combination of Ots (global) and adaptive thresholding methods.



A brain with a tumor, under the combination of Ots (global) and adaptive thresholding methods.



A brain without a tumor, under the combination of Ots (global) and adaptive thresholding methods.



A brain without a tumor, under the combination of Ots (global) and adaptive thresholding methods.

The result is similar to the result of the Ots method alone, and it is clearly not efficient, so we will use the adaptive method alone.

**5. Perform morphological operations:**

Using NumPy and openCV: Morphological operations can be used to further refine the segmentation results obtained after thresholding. Morphological operations like erosion and dilation can be applied to remove small artifacts or noise from the segmented image and to smooth the edges of the tumor.

Erosion can be used to remove small bright regions or isolated pixels in the background of the segmented image. This is done by convolving the image with a small structuring element and taking the minimum value of all pixels under the structuring element. Dilation, on the other hand, can be used to fill in small holes or gaps in the segmented image. This is done by convolving the image with a small structuring element and taking the maximum value of all pixels under the structuring element.

We created a structuring element (kernel) of size 5x5 with an elliptical shape. This kernel is then used in cv2.morphologyEx() function to perform morphological opening on the image, after thresholding, to remove small artifacts or noise.

**Note: What is numpy and why we use it?**

NumPy (Numerical Python) is a Python library used for working with arrays and matrices. It provides a powerful N-dimensional array object, as well as functions for performing mathematical operations on arrays.

**There are several reasons why NumPy is a popular library in scientific computing and data analysis:**

1. NumPy arrays are much faster and more memory-efficient than Python lists, especially for large data sets. This is because NumPy arrays are stored in contiguous blocks of memory, whereas Python lists are stored as a collection of pointers.
2. NumPy provides a wide range of mathematical functions that can be applied to arrays, such as sin(), cos(), exp(), log(), and many others.
3. NumPy allows for broadcasting, which means that operations can be performed on arrays with different shapes and sizes. This makes it easier to perform operations on large datasets.
4. NumPy provides linear algebra functions, such as dot product, matrix multiplication, and eigenvalues and eigenvectors.
5. NumPy can interface with other libraries, such as SciPy (Scientific Python) and Pandas (Python Data Analysis Library), which makes it a powerful tool for scientific computing and data analysis.

**Why did we choose 300dpi as the resolution?**

A resolution of 300 dpi (dots per inch) is often preferred for MRI images because it provides a high level of detail, which is important for accurately identifying and separating different structures within the image. Higher-resolution images can provide even more detail, but they may also require more processing power and storage space.

**6. Perform feature extraction:**

To perform feature extraction on the preprocessed brain tumor image, various techniques can be applied, including:

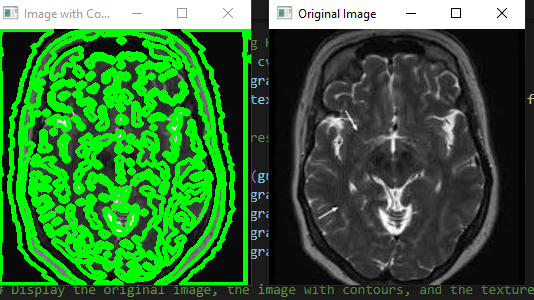
1. Contour detection: Contour detection can be used to extract the outer boundary of the tumor. The OpenCV function findContours() can be used to detect the contours of objects in the image. Once the contours are detected, their properties, such as area, perimeter, and shape, can be calculated to extract meaningful features.
2. Texture analysis: Texture analysis techniques can be used to extract features based on the texture of the image, such as entropy, contrast, and homogeneity. This can be done using various image analysis libraries, such as scikit-image or PyRadiomics.
3. Intensity-based features: Intensity-based features, such as mean, variance, and skewness, can be calculated for each segmented region, which can be used to distinguish between different types of tissue or lesions based on their intensity characteristics.
4. Machine learning: Machine learning techniques can be used to extract features automatically from the preprocessed image. This involves training a model on a dataset of preprocessed brain tumor images and their corresponding labels, such as tumor type, grade, or location. The model can then be used to extract features from new images.

In our program it performs the following steps:

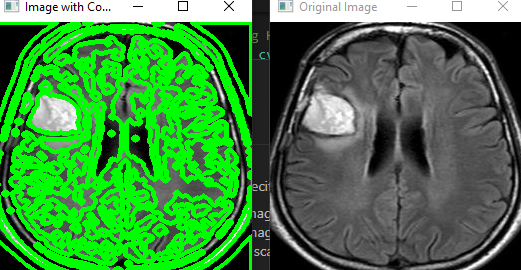
1. Performs contour detection on the opened image to identify the boundaries of objects in the image
2. Draws the contours on the original image to show the location of the objects
3. Calculates texture features and intensity-based features using Haralick texture analysis on the image with contours

Texture analysis and intensity-based features are useful for MRI brain tumor segmentation. MRI images contain a lot of texture and intensity variations, and these features can help differentiate between different types of tissue or lesions based on their texture and intensity characteristics. They can also help to identify the location and extent of a tumor within the brain. In fact, texture analysis and intensity-based features are commonly used in the development of computer-aided diagnosis (CAD) systems for brain tumor segmentation.

**Here is a comparison between the original image and the image after applying our preprocessing of brain tumor images using thresholding:**



Brain with no tumors.



Brain with tumor.