Practicum_Fondations_CV_Lab3 Report

Ivan Yahir Gómez Mancilla

Maxence Martin Camille André

December 11, 2023

1. Theoretical understanding

In this section, the theoretical knowledge contained in each of the methods implemented during the laboratory will be presented. The topics will be addressed in the order of appearance, and they will be listed to facilitate understanding and reading.

Threshold-Based Segmentation

Threshold-based segmentation is a foundational technique that plays a pivotal role in image processing. This method hinges on the establishment of a threshold to categorize image pixels into two distinct groups: object and background. Here, we delve into three key techniques within this category, each addressing various aspects of segmentation challenges.

a) *Global Thresholding*: Global thresholding is a straightforward approach where a single threshold value is applied uniformly across the entire image. This simplicity is advantageous for its ease of implementation; however, it comes with a caveat. Global thresholding can be particularly sensitive to variations in lighting conditions. Consequently, in scenarios with changing illumination, the accuracy of segmentation may be compromised. This technique is a foundational concept, laying the groundwork for more advanced segmentation methods.
b) *Adaptive Mean Thresholding*: In response to the challenges posed by global thresholding, the adaptive mean thresholding algorithm introduces a more nuanced strategy. Instead of applying a single threshold to the entire image, this algorithm calculates local thresholds based on the mean values of pixels in different regions of the image. By doing so, the algorithm dynamically adjusts the threshold based on the local characteristics of the image. This adaptive approach enhances segmentation robustness, especially in situations where lighting conditions vary across various parts of the image.
c) *Otsu's Thresholding*: Otsu's Thresholding represents an automated and sophisticated approach to threshold determination. This method seeks to find an optimal threshold by minimizing intra-class variance or, equivalently, maximizing inter-class variance. The result is a precise segmentation that adapts to the inherent properties of the image. Otsu's Thresholding is particularly valuable in scenarios where a clear demarcation between object and background intensities exists. The automated nature of this technique makes it well-suited for a wide range of image processing applications, ensuring adaptability to diverse image characteristics.

In essence, these threshold-based segmentation techniques showcase a spectrum of approaches, from the simplicity of global thresholding to the adaptability of algorithms like Adaptive Mean Thresholding and the precision of automated methods like Otsu's Thresholding. Each technique serves as a building block in the understanding and application of segmentation strategies, catering to different requirements and challenges encountered in image analysis.

Edge-Based Segmentation

Edge-based segmentation refers to the process of identifying and isolating boundaries or edges within an image, where significant changes in intensity occur. This approach enables the extraction of essential features, making it a cornerstone for various computer vision applications. In this section, we will explore two prominent methods for edge-based segmentation: the Canny edge detector and the Laplacian of Gaussian (LoG).

a) *Sobel operator*: The Sobel Operator is a convolution-based edge detection method commonly employed in image processing and computer vision. Its primary objective is to capture the rate of change of pixel intensities in an image, providing insights into the spatial variations and, consequently, highlighting potential edges or boundaries. This operator is specifically designed to be sensitive to vertical and horizontal changes in intensity.
The Sobel Operator utilizes convolution with two 3x3 kernels: one for detecting changes in the horizontal direction and the other for changes in the vertical direction. These kernels emphasize the gradient components along the respective axes. The convolution process involves overlaying the kernels on the image and computing the weighted sum of pixel values within the kernel window. The resulting gradient approximations are then combined to yield the overall gradient magnitude and direction.
Mathematically, let I represent the image intensity function, and G_x and G_y denote the horizontal and vertical Sobel kernels, respectively. The convolutions can be expressed as follows:

$$Gradient\ in\ the\ horizontal\ direction\ (G_x): I_x = I * G_x$$
$$Gradient\ in\ the\ vertical\ direction\ (G_y): I_y = I * G_y$$

The overall gradient magnitude ($|\nabla I|$) and direction (θ) can then be calculated using the individual gradient components:

$$|\nabla I| = \sqrt{I_x^2 + I_y^2}$$
$$\theta = \arctan\left(\frac{I_y}{I_x}\right)$$

b) *Canny Edge-Based segmentation*: used in computer vision to identify and extract edges from an image, Canny edge-based segmentation is a technique which goal is to find the boundaries of objects within an image by detecting significant changes in intensity, which often correspond to the presence of an edge. It is usually described as a robust method, having a vast number of subphases where the images are transformed sequentially taking the main advantages of a diverse group of techniques. The classical implementation works with the next workflow:
   1. Gaussian Blur: The original image, denoted as I, undergoes Gaussian Blur with a predefined Gaussian kernel size and sigma value. This step eliminates significant noise in the image, smoothing it and preparing it for subsequent region identification processes.
   2. Sobel Edge Detection: The Sobel algorithm is employed to calculate the image gradient. By applying horizontal and vertical Sobel kernels, both the gradient matrix and Theta values are obtained, essential for the next stage.
   3. Non-Maximum Suppression: This stage retains only maximum intensity values, considering the direction and magnitude of the gradient vector obtained from the Sobel Edge Detector. Non-Maximum Suppression aims to eliminate non-maximum intensities across the entire image, preserving only the most prominent intensity changes, typically representing image contours.
   4. Double Threshold: Using two threshold values, this stage categorizes points in the image based on their intensities. The categories include:

a. Strong: Points with intensity such that their likelihood of contributing to a contour is nearly certain.
b. Null: Points with intensity such that their likelihood of contributing to a contour is nearly non-existent.
c. Weak: Points with intensity between strong and null, not strong enough to be marked as strong but not weak enough to be null. The algorithm reduces all intensities in the image to three values, with strong receiving maximum intensity, null the minimum, and weak an intermediate value determined by the developer. Threshold values are typically derived as a percentage of the minimum and maximum intensity values in the original image.

5. Hysteresis: In the final phase, hysteresis utilizes the double threshold transformation to determine if weak points belong to a contour. The criterion is as follows: any weak point adjacent to a strong point is considered strong, while others are considered null.

Canny edge detection algorithm is a sophisticated and effective method, its five-stage process reduces noise, computes gradients, retains significant intensity changes, categorizes points based on thresholds, and establishes strong or null points through hysteresis. The output is a binary image that ideally captures the contours present in the original image.

c) *Laplacian of Gaussian (LoG):* The Laplacian of Gaussian (LoG) serves as a robust contour detection method, employing a multi-stage process that involves Gaussian smoothing, convolution with Laplace kernels, and a final step known as Zero Crossing to pinpoint contours within an image. While it is feasible to compute the LoG in a single operation, the conventional approach divides the process into distinct stages to enhance precision and control over the results.

The initial stage entails Gaussian smoothing, wherein the image undergoes blurring using a Gaussian kernel. This smoothing process incorporates key parameters such as the kernel size and the sigma value, influencing the computation of function values. Following the smoothing phase, the resultant image is then subjected to convolution with Laplace kernels to accentuate the contours. This sequential approach allows for a nuanced treatment of the image, optimizing the detection of edges and ensuring a more refined output.

The final processing step, Zero Crossing, draws from signal processing theory. It involves identifying points or instances where a signal crosses a designated value boundary, often referred to as zero value. In the context of image processing, a point is identified when there is a discernible change in intensity within the neighboring directions, causing this change to cross the zero value. This meticulous identification is crucial because such intensity variations frequently correspond to contours in the image. By identifying and amplifying these changes to their maximum intensity, the effectiveness of contour detection is significantly enhanced.

Laplacian of Gaussian, with its staged approach emerges as a sophisticated and effective method for precise contour detection in computer vision applications.

Watershed-Based Segmentation

Watershed-based segmentation, commonly referred to as the Watershed algorithm, is a sophisticated technique inspired by the topographic concept of watersheds. This innovative method leverages the analogy of a landscape's watersheds to delineate boundaries between different objects within an image. The crux of

this approach lies in treating the image gradient as a topographic surface, with watershed lines serving as the boundaries that separate distinct regions.

The Watershed algorithm operates on the premise of the image gradient, which reflects the intensity changes across the image. By considering gradient values as analogous to topographic heights, the algorithm interprets the image as a three-dimensional landscape. In this landscape, pixels with high gradient values correspond to elevated regions, while low gradient values represent depressions.

The algorithm simulates a flooding process, where each pixel is treated as a point in the landscape. As the flooding progresses, it identifies watershed lines—the points where flooding from various sources converges. These watershed lines become the natural boundaries between different objects or regions in the image.

This approach proves particularly effective in handling complex images characterized by intricate object structures, texture variations, and overlapping features. The Watershed algorithm excels in scenarios where other segmentation methods might struggle, offering a powerful solution for applications requiring detailed and accurate delineation of object boundaries.

The Watershed algorithm's strength lies in its ability to adapt to the local characteristics of an image. It discerns subtle changes in intensity and texture, allowing it to precisely identify boundaries between objects. While the Watershed method may require careful preprocessing to address challenges such as noise or unwanted local minima, its performance in segmenting complex images makes it a valuable tool in the image processing toolkit.

Region-Based Segmentation

Region-based segmentation is a computer vision technique that involves partitioning an image into meaningful and homogeneous regions based on certain characteristics or criteria. This method aims to group pixels that share similar attributes, such as color, texture, or intensity, into distinct regions. Unlike edge-based segmentation, which focuses on identifying boundaries, region-based segmentation emphasizes the coherent grouping of pixels to form perceptually meaningful regions.

Among the diversity of possibilities in this group of algorithms, we chose Split-and-Merge using Region Adjacency Graphs (RAG), which represents a sophisticated approach to region-based segmentation in computer vision. This method is particularly powerful in partitioning an image into homogeneous regions by recursively splitting and merging, driven by the analysis of region adjacency relationships. It works as follows:

The initiation of the Split and Merge algorithm involves a holistic perspective on the original image, treating it as a unified and cohesive entity. This initial phase aims to create a structured segmentation by adhering to specific criteria. The algorithm meticulously divides the image into smaller regions, each regarded as an independent and freshly generated segment. These resultant regions are not merely isolated entities but are systematically documented and represented through the construction of a Region Adjacency Graph (RAG) which serves as a pivotal element in the algorithm's methodology.

The RAG transforms the segmented image into a graph structure where nodes represent individual regions, and connections signify the adjacency relationships between these regions. Importantly, the strength of these connections is determined by a weighted criterion, reflecting the dissimilarity in characteristics

between the connected regions. This graph-based representation becomes instrumental in orchestrating subsequent iterations of the algorithm.

The heart of the algorithm lies in its iterative exploration of the Region Adjacency Graph. This iterative process systematically evaluates the connections and dissimilarities between regions with the overarching goal of identifying nodes (regions) that satisfy a predefined criterion for similarity. The algorithm seeks to recognize adjacency connections that align with this criterion, compelling the merging of these regions into a unified and cohesive group.

Upon the conclusion of a round of RAG iteration, the algorithm dynamically refines its understanding of the image segmentation. It doesn't stop there; the algorithm undertakes a subsequent iteration by once again partitioning the image into smaller sectors. This phase involves a careful analysis of regions within these sectors, identifying those that exhibit a potential for amalgamation based on the established criteria. The algorithm then seamlessly merges these identified regions, fostering a continuous and dynamic refinement of the image segmentation.

This cyclical process of RAG exploration, criteria-based amalgamation, and image partitioning exemplifies the adaptability and dynamism of the Split and Merge algorithm. It ensures that the segmentation evolves iteratively, progressively enhancing the cohesion of regions and refining the overall representation of the original image. The algorithm's ability to balance global and local considerations through these iterations contributes to its effectiveness in capturing diverse structures and patterns within images.

Clustering-Based Segmentation

Technique that involves grouping pixels or regions in an image into distinct clusters or segments based on their similarities in certain feature spaces. The fundamental idea is to identify and group together elements that share common characteristics, such as color, intensity, texture, or spatial proximity. Clustering algorithms play a central role in this segmentation approach, as they aim to partition the image into homogeneous groups, or clusters, while maximizing the similarity within each cluster and minimizing the similarity between different clusters. Two methods were used in this laboratory:

a) *K-Means*: The primary objective of k-Means is to partition a dataset into k clusters, where each cluster is characterized by a centroid that represents the mean of the data points belonging to that cluster. In the context of image segmentation, k-Means is applied to group pixels based on similarity in feature space, leading to the creation of distinct regions or segments within an image. It involves:
    1. Initialization: The process begins by selecting k initial centroids, which can be chosen randomly or through a more sophisticated initialization method. Each data point is then assigned to the cluster represented by the nearest centroid.
    2. Assignment: In the assignment step, each pixel in the image is assigned to the cluster whose centroid is closest in feature space. Common features include color values, intensity, and spatial coordinates.
    3. Centroid Update: After the assignment step, the centroids of the clusters are recalculated by taking the mean of the feature values of all the pixels assigned to each cluster.
    4. Reassignment: The assignment and centroid update steps are repeated iteratively until convergence. Pixels are reassigned to clusters based on the updated centroids, and the centroids are recalculated accordingly.

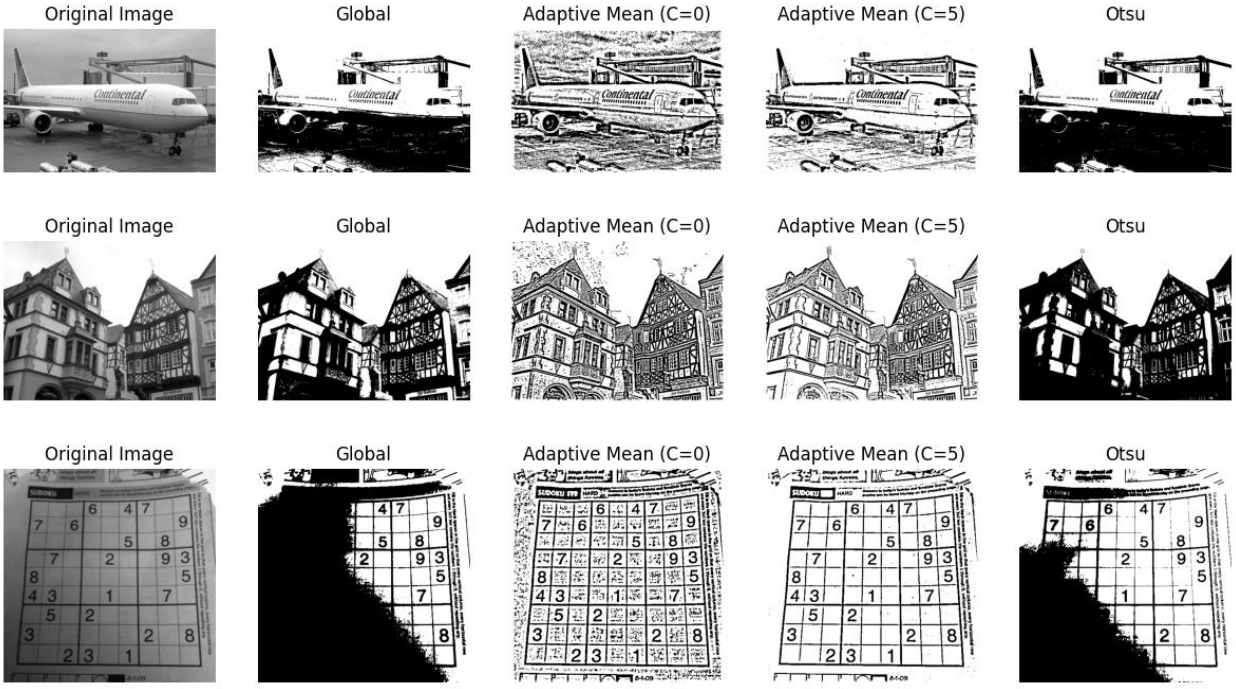
5. Convergence: Convergence occurs when there is minimal change in the assignments or centroids between iterations. At this point, the algorithm has successfully grouped pixels into k clusters, and the process concludes.

b) *Gaussian Mixture Models (GMM):* This method represents a probabilistic approach to clustering and density estimation, frequently employed in image segmentation. Unlike k-Means, which assumes spherical clusters in the feature space, GMMs allow for more flexible and ellipsoidal clusters, making them powerful for capturing complex distributions within an image. It follows the steps:

1. Initialization: GMM initialization involves setting initial values for the mean, covariance, and weight of each Gaussian component. Commonly used techniques include k-Means clustering for initial means and identity matrices for covariances.
2. Expectation-Maximization (EM): GMM parameters are iteratively refined using the Expectation-Maximization algorithm. The E-step computes the likelihood of each data point belonging to each cluster, while the M-step updates the parameters based on these likelihoods.
3. Convergence: The algorithm iterates until convergence, where the parameters stabilize and do not change significantly between iterations.

2. Results and discussion

Threshold-Based Segmentation

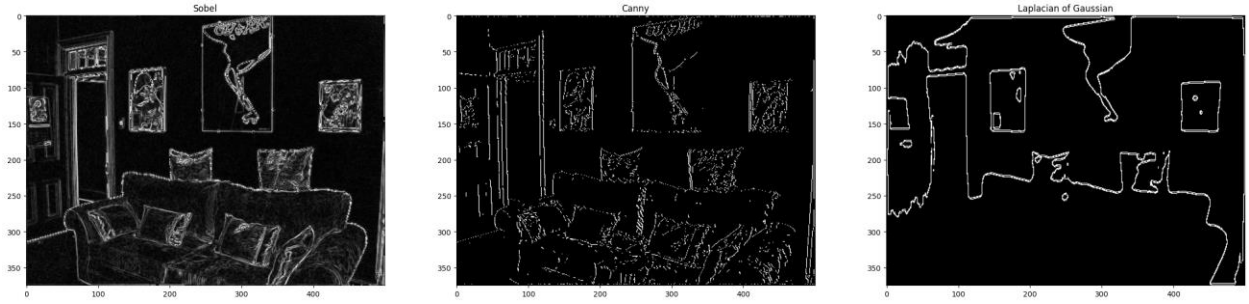


We can see that the various algorithms have different respective effects on image processing. As expected, global thresholding yields poor results when there is a high variation in brightness within the same image, as observed in the processing of the sudoku image. Regarding Continental's plane, the algorithm struggles to correctly distinguish the bottom of the fuselage from the tarmac. The facade, on the other hand, produces results that are not bad, although the details could be more precise.

Next, the adaptive mean thresholding algorithm provides excellent results. It is necessary to play with its threshold hyperparameter, C, to find the best results. It is noticeable that a smaller C makes it more sensitive to noise, as can be seen, especially in the photos of the plane and the facade. After filtering, clouds that were nearly invisible to the naked eye become much clearer. The parameter C=5 produces particularly impressive results with a high level of detail and very little noise.

Finally, the Otsu thresholding algorithm presents mixed results, comparable to global thresholding. There may be a programming error present, as the expected results for this algorithm were significantly higher than the obtained results.

Edge-Based Segmentation



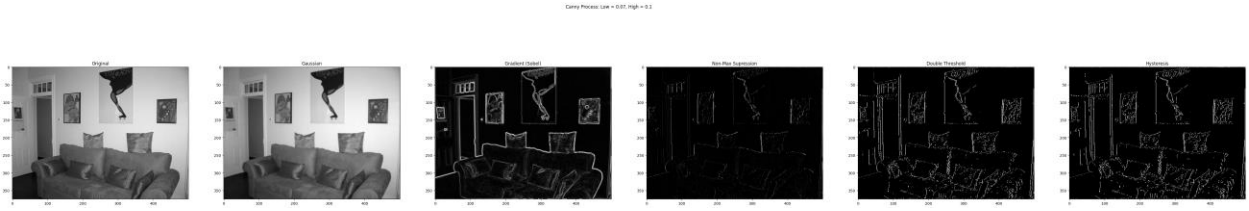
From this we highlight that the Sobel operator effectively enhances the contrast between image contours and flat backgrounds. This is evident in the room image, particularly on the walls and floor, where areas lacking intricate details exhibit a reduced intensity, forming distinct contours around the painting, door, and sofa boundaries. However, the Sobel operator's drawback lies in its sensitivity to image noise, noticeable in the sofa and cushion areas, where undesired shadows are introduced due to the texture, adversely affecting the result.

On the other hand, the Laplacian of Gaussian (LoG) serves as an alternative with high dependency on defined parameters. While it excels in defining contours with clarity and continuity, variations in parameters significantly impact results. Striking a balance becomes challenging, often leading to compromises, such as emphasizing contours with more pronounced intensity changes. This trade-off is observable in the image, where certain parts are vividly highlighted, but details in other regions, such as the floor demarcation or bottom of the sofa, are lost.

Lastly, the Canny edge detector positions itself as a middle ground between the Sobel and LoG filters. It binary highlights points with more noticeable intensity changes in the image, addressing noise sensitivity without overlooking crucial details. Unlike LoG, Canny yields contours that are not as continuous, yet it provides a balanced image. It allows for the definition of figures and a clear representation of structural details, striking an acceptable equilibrium between noise perception and contour precision.

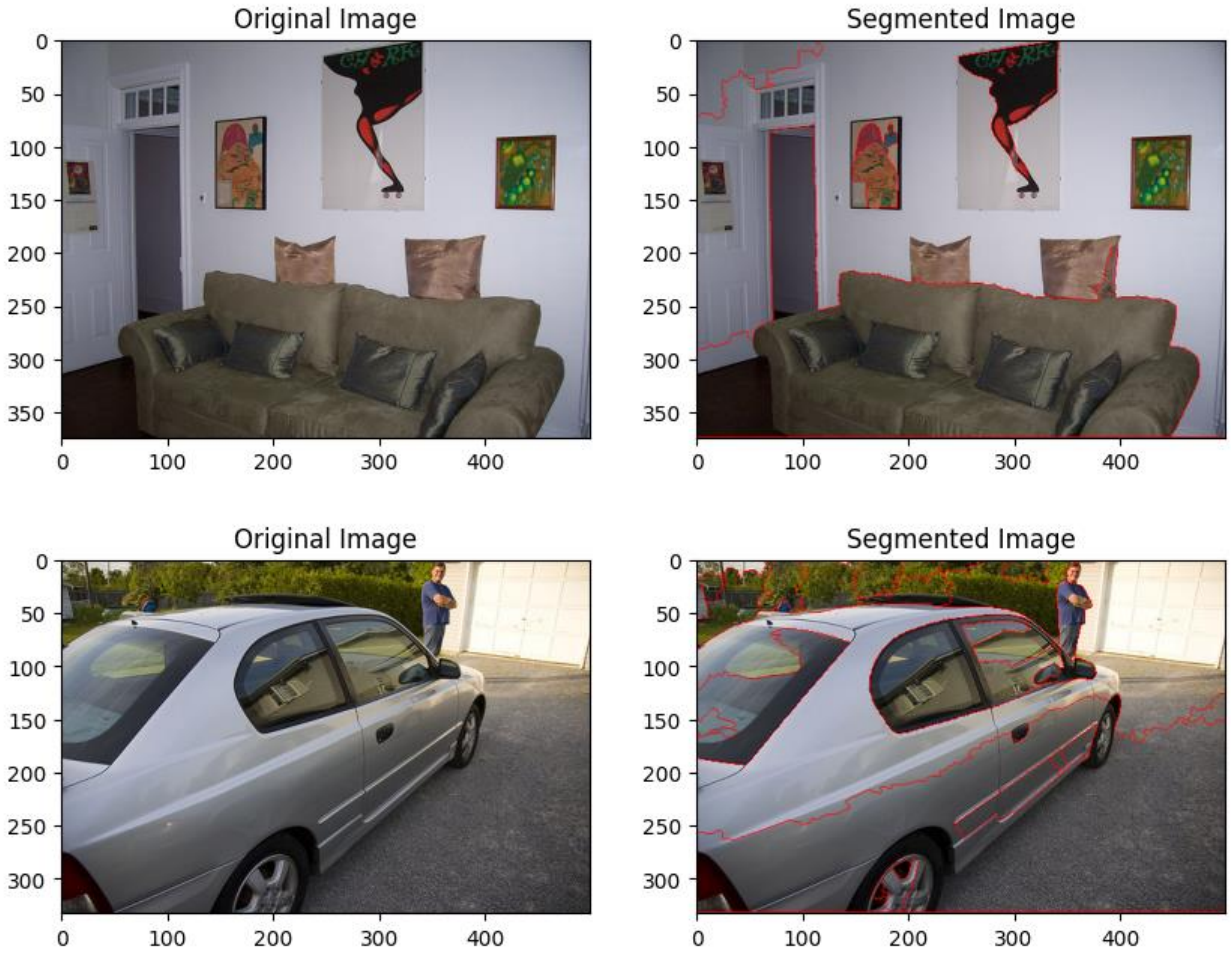
For us it was important to have control on each phase of Canny's and LoG's methods, that's why we also get the following visuals:

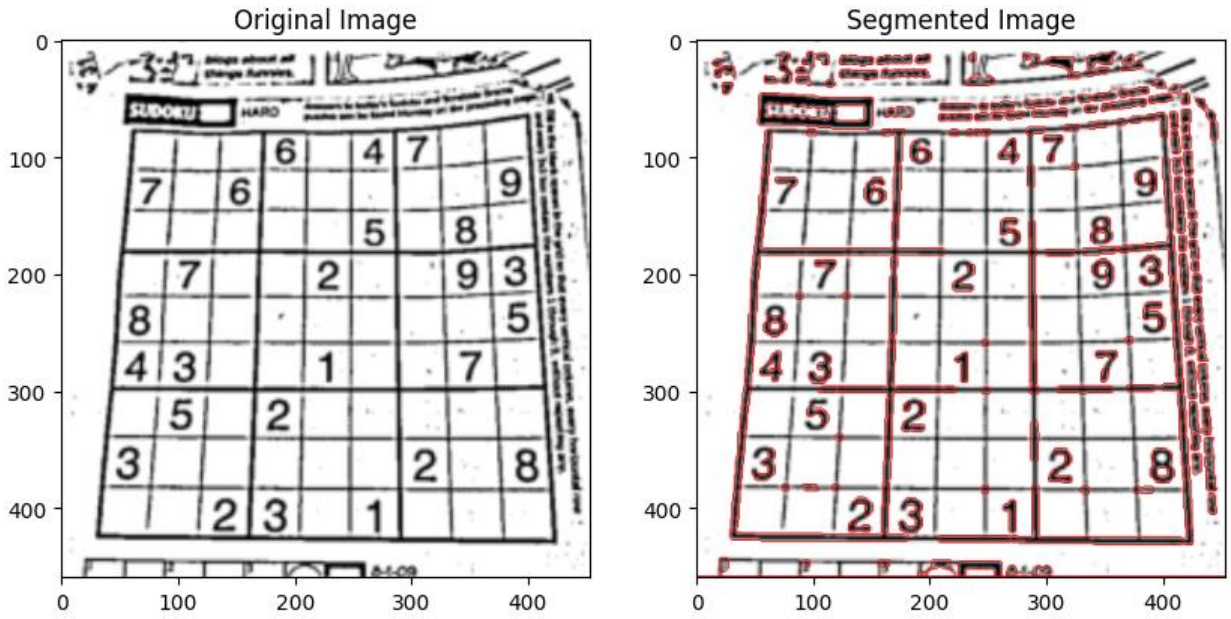
1. Canny:



2. LoG:

Watershed-Based Segmentation





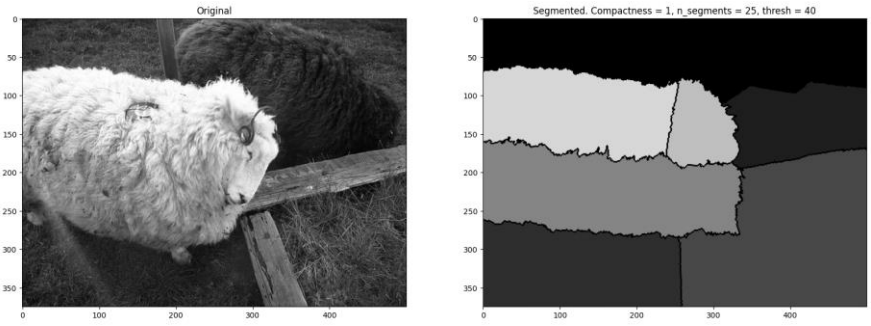
As you can see, our implementation of the Watershed algorithm yields mixed results. The segmentation provided by the algorithm on these complex images is inconsistent. Regarding the room with the sofa, the results are not bad. Indeed, it clearly distinguishes the sofa from the wall; however, it fails to separate the sofa from the floor. Similarly, it correctly identifies several elements in the paintings. However, it segments the top-left part of the image very poorly, where the delineation it defines does not represent anything real.

Concerning the car, we observe that the results are catastrophically poor. This is due to the complexity of the image composition, with very sharp changes in brightness due to external light, reflections from the body, and the transparency of the windows—all these elements completely confuse the algorithm in its segmentation.

Regarding the sudoku image, after processing, the results seem significantly better as it distinguishes the different black/white areas more clearly. However, one could have expected better performance from such a simple input image.

Region Based Segmentation

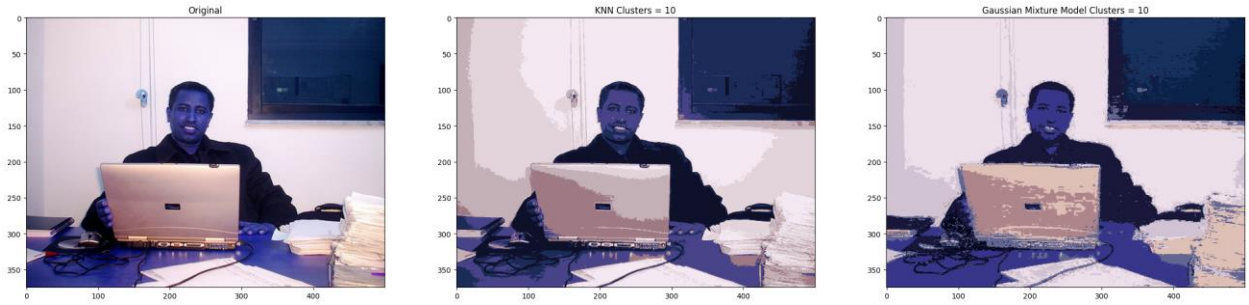
As the result of our work with Split-and-Merge algorithm using RAG, we obtain the following visual:



Were we highlight that, while the algorithm adeptly handles substantial intensity changes, as evidenced in the image featuring high-intensity values of the sheep, defining even its sections effectively, it encounters challenges in discerning subtle intensity variations. This limitation becomes evident in regions with low-intensity changes, such as the sheep's areas with lower intensity values and the trough section, where the entire food zone merges with the wood boundary. Although the algorithm's performance varies with different input values, potentially allowing for better configurations, the identification of regions remains acceptable, albeit not optimal. Despite accurately outlining expected shapes, the result may deviate in terms of both position and size from the anticipated outcome.

Clustering-Based Segmentation

For the analysis of this method, we apply both algorithms (K-Means and GMM) to an image, obtaining the visual:



From it we discuss that considering the inherent characteristics of the K-Nearest Neighbors (KNN) model, which determines similarity based on proximity to clusters around a given point, it becomes apparent that the segmentation it performs on an image can be influenced by the similarity in intensities of two objects. This effect is noticeable in the segmentation of the laptop and background, where the algorithm divides the laptop into sections based on its brightness intensity, incorporating them into the cluster under which the wall was segmented. It's worth highlighting the circular shape of these clusters, emphasizing the tonal changes caused by frontal illumination in the image. The sensitivity to illumination also impacts the face of the subject, particularly in the chin area, segmented differently from the rest of the face. Notably, the algorithm excels in accurately defining the differences between the table and the objects placed on it.

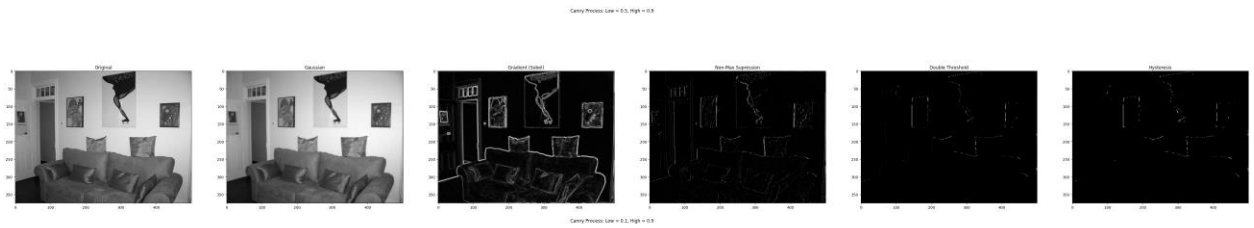
In contrast, the Gaussian Mixture Model (GMM) provides a more regular segmentation, exhibiting less noise in the result caused by intensity changes due to illumination. This uniformity is observable in the background, showcasing non-circular groupings. The face and body of the subject also appear less affected by noise. However, the segmentation of the laptop, while an improvement over KNN, maintains the division into several clusters, preventing the individual segmentation expected for this object. The segmentation of the table is generally satisfactory, although there is a noticeable challenge in segmenting the stack of sheets on the right compared to KNN.

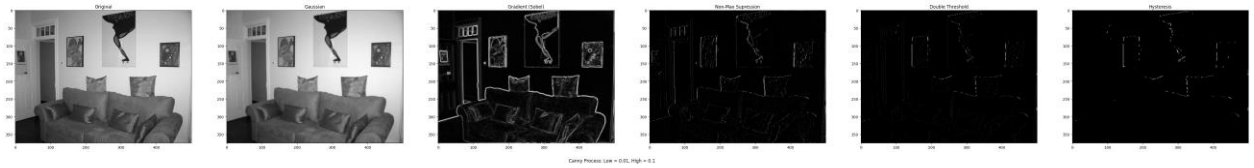
It's crucial to note the longer execution times associated with the GMM model. This can be interpreted as a necessary tradeoff for achieving uniformity in segmentation. Consequently, we contend that KNN might be a preferable option when dealing with images exhibiting minimal intensity variation due to luminosity issues and containing few objects. On the other hand, GMM could be more effective in handling images of greater complexity, despite the higher execution costs involved.

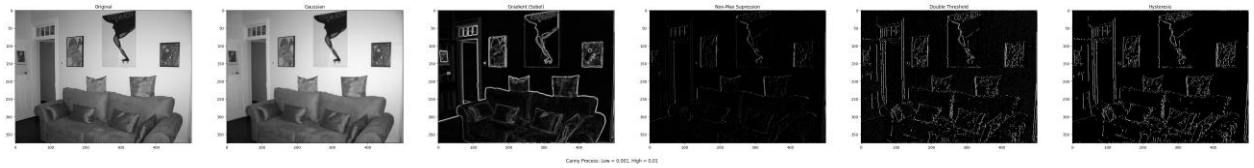
3. Experiments

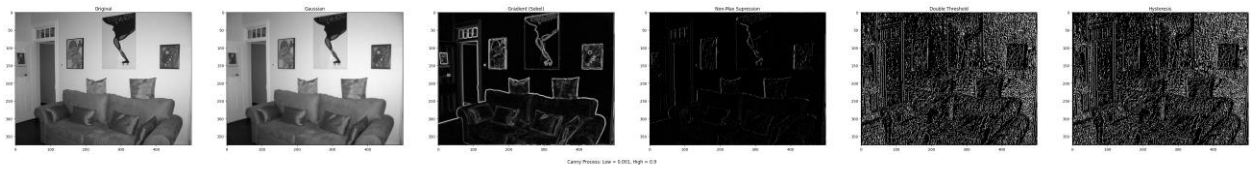
Edge-Based segmentation

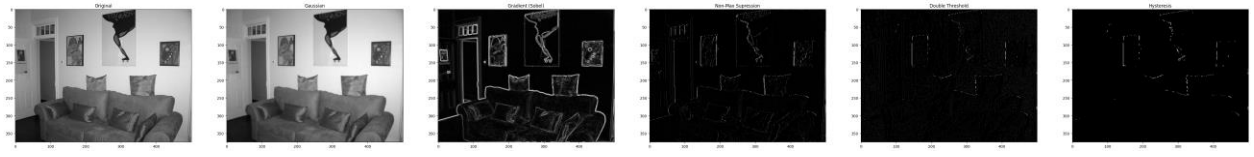
Tunning the parameters of Canny algorithm, we got a wide variety of results before getting to such a satisfactory configuration as the one presented on results section. A list of some of those results is presented below:

1. 

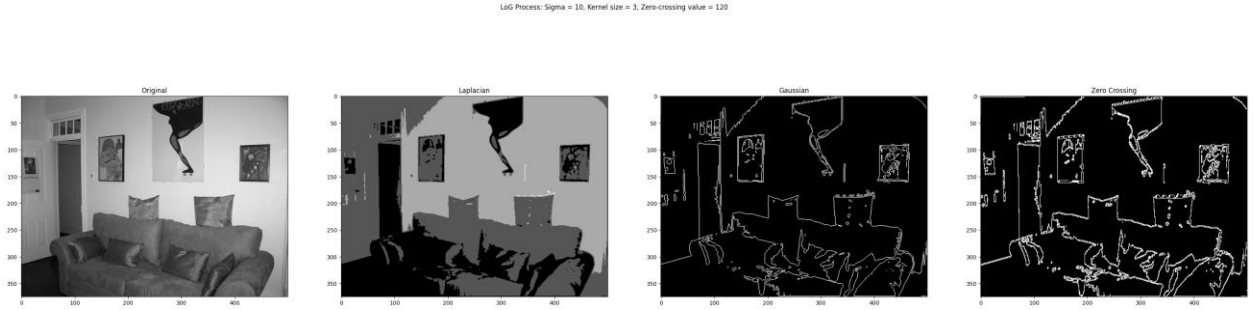
2. 

3. 

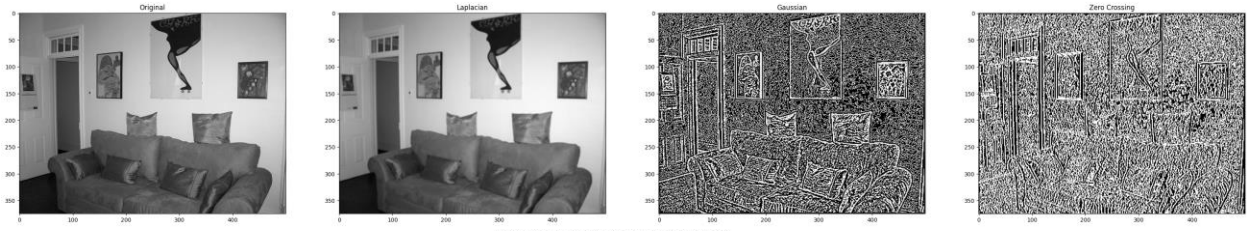
4. 

5. 

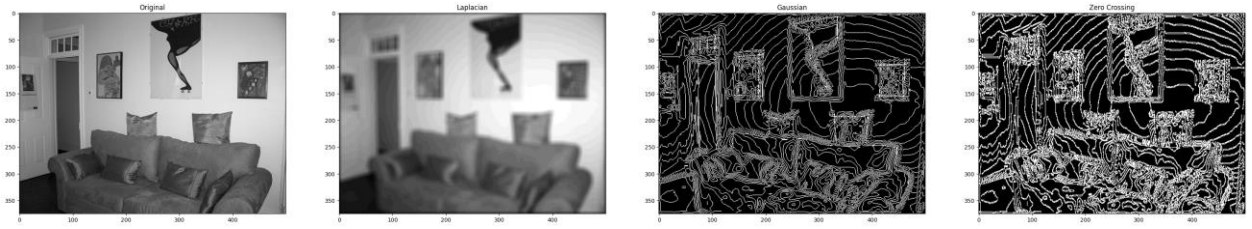
The same as before is presented for LoG algorithm:

1. 

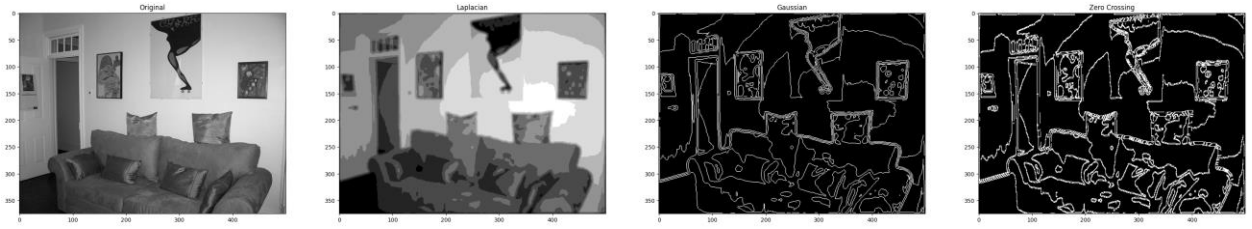
LoG Process: Sigma = 1, Kernel size = 3, Zero-crossing value = 120

2. 

LoG Process: Sigma = 10, Kernel size = 11, Zero-crossing value = 120

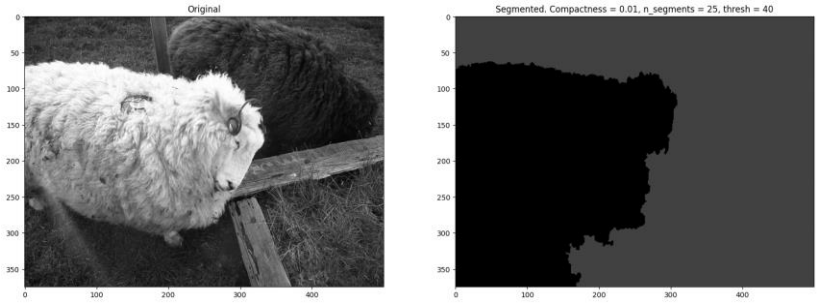
3. 

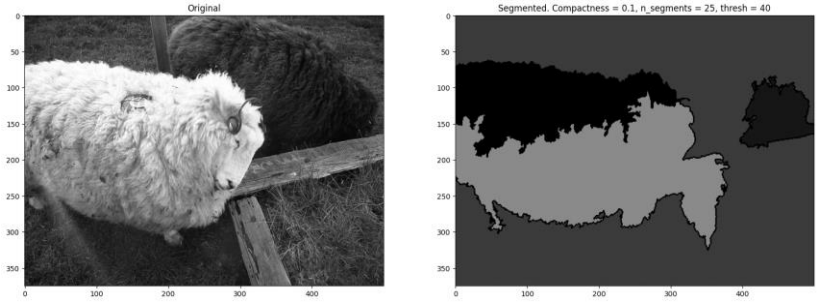
LoG Process: Sigma = 15, Kernel size = 7, Zero-crossing value = 80

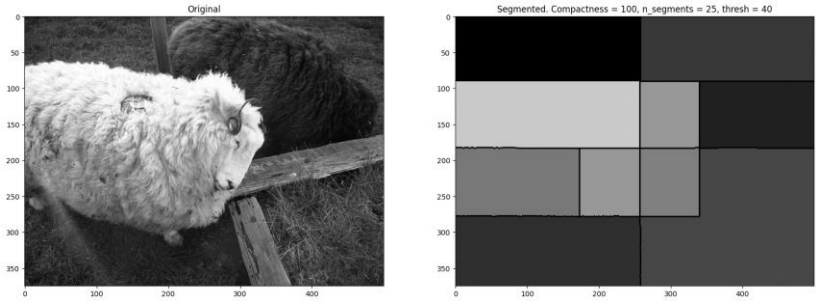
4. 

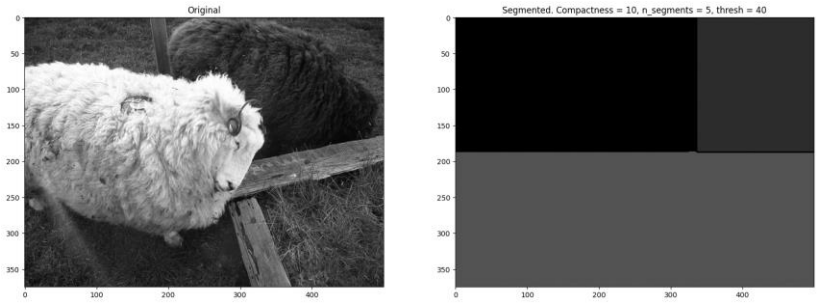
Region-Based Segmentation

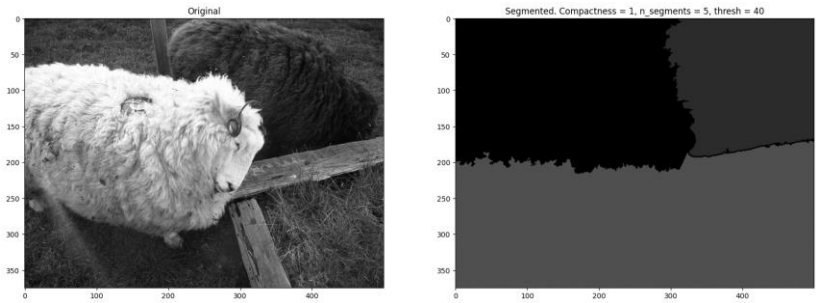
In Split-and-Merge algorithm, our main playground recalls on the method's parameters, those are the compactness of the segments, the total number of splitted segments and the threshold for deciding to merge or not a section. A list of variations and the obtained results is shown below:

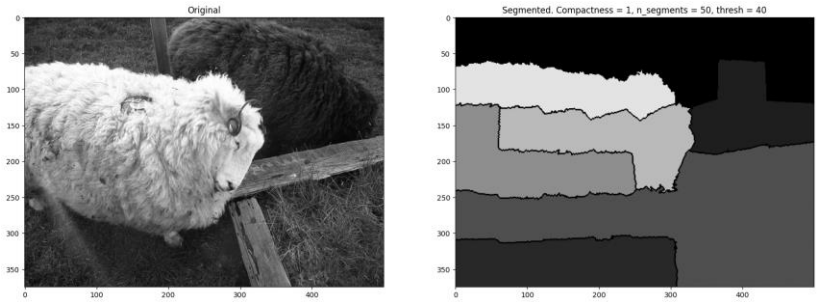
1. 

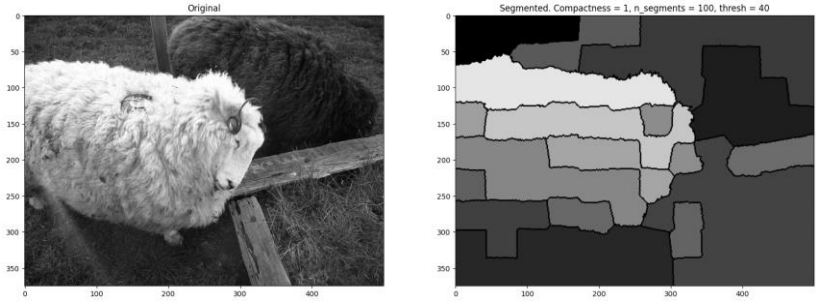
2. 
Original
Segmented. Compactness = 0.1, n_segments = 25, thresh = 40

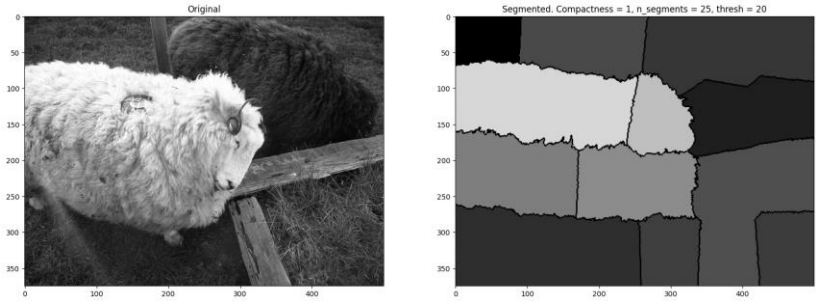
3. 
Original
Segmented. Compactness = 100, n_segments = 25, thresh = 40

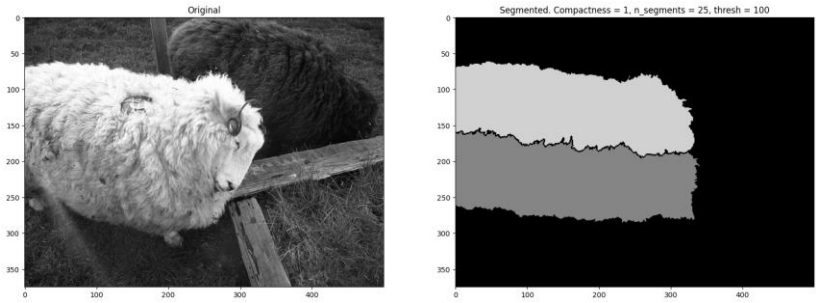
4. 
Original
Segmented. Compactness = 10, n_segments = 5, thresh = 40

5. 
Original
Segmented. Compactness = 1, n_segments = 5, thresh = 40

6. 
Original
Segmented. Compactness = 1, n_segments = 50, thresh = 40

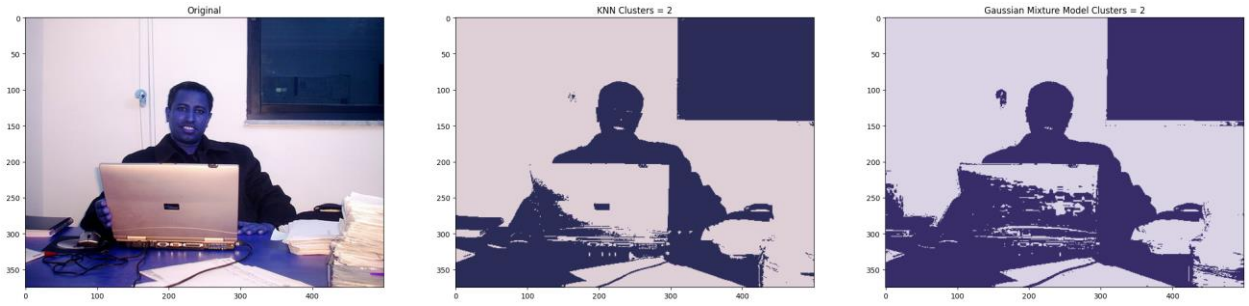
7. 

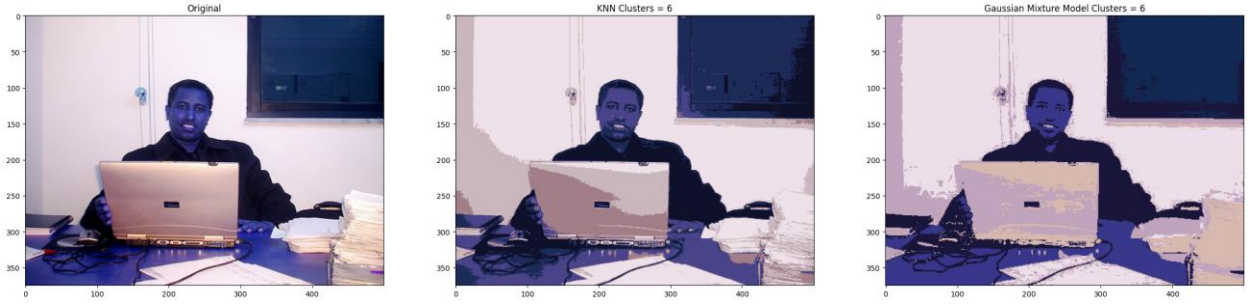
8. 

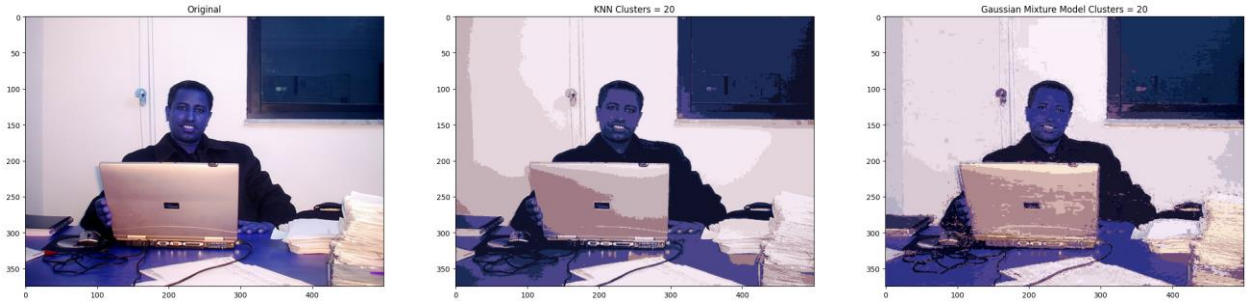
9. 

Cluster-Based segmentation

For each of the algorithms implemented, there is one principal parameter to tune: the number of clusters created. Below we can observe some variations on these parameters.

1. 

2. 

3. 

4. Conclusion

In this comprehensive exploration of various segmentation techniques, we have gained valuable insights into the strengths and limitations of different methods applied to diverse aspects of image processing. The threshold-based segmentation methods, including Global Thresholding, Adaptive Mean Thresholding, and Otsu's Thresholding, showcased their efficacy in separating objects from backgrounds under varying conditions. These techniques provide fundamental tools for basic image segmentation needs, each addressing specific challenges associated with global and adaptive thresholding.

Edge-based segmentation, employing the Sobel Operator, Canny Edge Detector, and Laplacian of Gaussian, demonstrated their prowess in detecting edges and contours, catering to the nuanced requirements of different applications. The multi-stage nature of the Canny Edge Detector highlighted its adaptability to various scenarios, from noise reduction to edge tracking. The Laplacian of Gaussian method, while computationally heavier, exhibited its capability to capture both blobs and edges in a refined manner.

The watershed-based segmentation approach, leveraging the Watershed Algorithm, provided a unique perspective by treating the gradient magnitude as a topographic surface. This technique, implemented through OpenCV, demonstrated its efficiency in segmenting images based on catchment basins, particularly useful in scenarios where regions are defined by gradient variations.

In the realm of region-based segmentation, we explored both Split-and-Merge and Flood Fill techniques. The Split-and-Merge method exhibited a recursive approach to region partitioning, emphasizing homogeneity criteria, while Flood Fill, a seed-based method, demonstrated its ability to grow regions from a seed point. Both techniques underscored the importance of homogeneity in defining and merging regions.

The analysis of clustering-based segmentation methods, K-means Clustering and Gaussian Mixture Models (GMM), offered valuable insights into their applications. K-means, with its predefined cluster count, proved effective in partitioning images based on pixel color or intensity. In contrast, GMM, with its probabilistic model, provided a soft-segmentation approach, offering a nuanced understanding of normally distributed subpopulations within the overall image.