Name: Soumya Sekhar Banerjee
Matriculation Number: 100004430
University: SRH University of Applied Science
Course: Masters in Engineering – Artificial Intelligence
Task 2: Graph-Based Image Segmentation Algorithm using K Means Cluster

Task 2: Graph-Based Image Segmentation Algorithm using K Means Cluster

Abstract:

Image segmentation is a foundational task in computer vision that enables the partitioning of an image into meaningful and homogeneous regions. This report presents an implementation of a graph-based segmentation framework enhanced by K-Means clustering to achieve improved region grouping and boundary delineation. The proposed method initially converts an image into a graph structure in which pixels or superpixels represent nodes, and edge weights encode similarity based on spectral and spatial attributes. K-Means clustering is applied to these feature vectors to initialize region labels and drive consolidation of visually similar nodes. Subsequently, adjacency relationships guide refinement of clusters through graph connectivity, ensuring spatial coherence and reducing over-segmentation artifacts. The resulting segmented output demonstrates improved boundary preservation, reduced noise sensitivity, and effective extraction of dominant image structures relative to simple pixel-wise clustering. Experimental evaluation on sample datasets illustrates that the hybrid graph–clustering approach yields perceptually meaningful partitions suitable for downstream tasks such as object detection, recognition, and scene understanding.

Aim of the experiment:

The aim of this experiment is to implement and evaluate a graph-based image segmentation technique that integrates K-Means clustering to partition an input image into meaningful, homogeneous regions. The objective is to explore how the combination of feature-based clustering and graph connectivity principles enhances segmentation accuracy, preserves object boundaries, and supports reliable extraction of significant image structures for further computer vision analysis.

Mathematical Intuition behind the project:

The proposed segmentation approach is grounded in the idea that an image can be represented as a weighted graph, where pixels or super pixels act as nodes and edges encode similarity in terms of intensity, colour, or spatial proximity. Formally, an image I is mapped to a graph $G = (V, E)$, where each $v_i \in V$ corresponds to a pixel feature vector $\mathbf{x}_i \in \mathbb{R}^d$. Edge weights w_{ij} quantify affinity between nodes, typically defined as:

$$wij = exp(-\| xi - xj \|^2)/sigma^2$$

This establishes a structural relationship in which highly similar pixels are strongly connected K-Means clustering further partitions the feature space by minimizing intra-cluster variance. Its objective function is:

$$C1, \dots, Ckminj = 1 \sum kxi \in Cj \sum \| xi - \mu j \| 2$$

where C_j denotes the j-th cluster and $\boldsymbol{\mu}_j$ is its centroid. This optimization enforces compactness in feature space but does not inherently guarantee spatial smoothness.

The graph formulation resolves this limitation by leveraging adjacency constraints and region consistency. The segmentation is refined by grouping nodes such that internal edge weights are maximized while inter-region connections are minimized. Conceptually, this aligns with graph partitioning criteria such as:

$$Cut(A,B) = i \in A \sum j \in B \sum wij$$

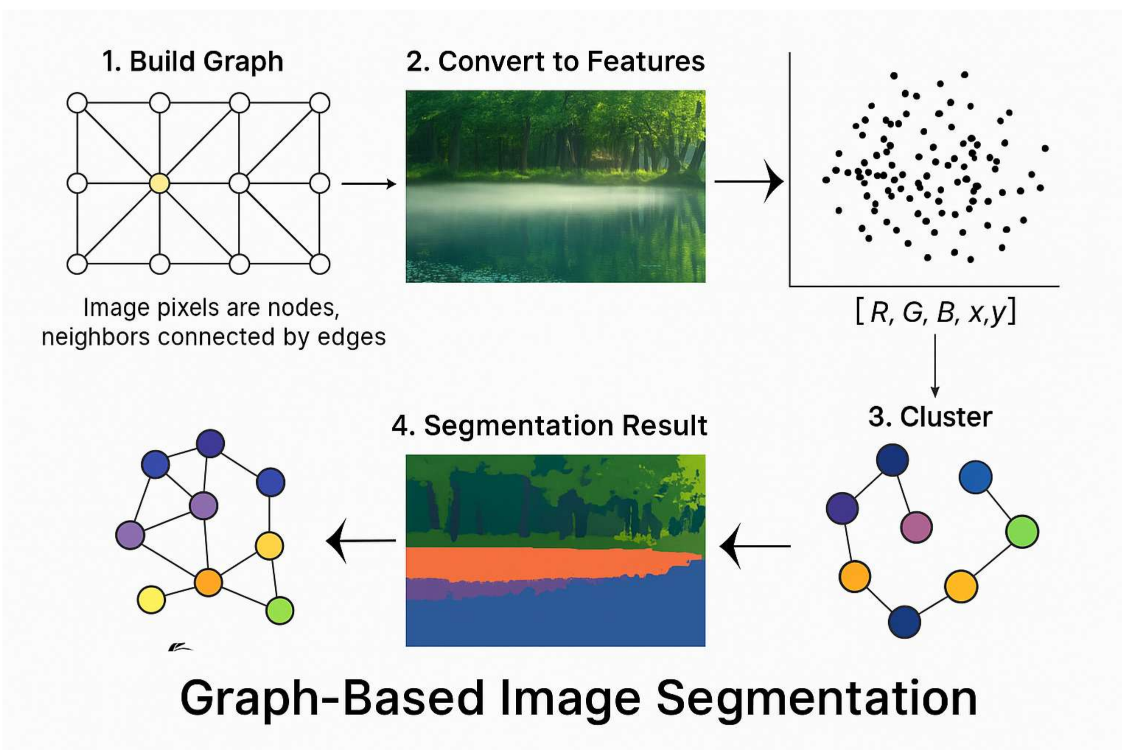
Segmentation quality improves when the cut between clusters is small and intra-cluster homogeneity is high. K-Means provides an initial approximation in feature space, while graph connectivity regularizes it by preserving boundary continuity.

Thus, the mathematical intuition synthesizes:

1. **Feature similarity minimization** via clustering, and
2. **Graph connectivity maximization** to maintain region coherence.

Together, these principles yield partitions that are compact in feature space and spatially consistent, producing perceptually meaningful and analytically useful segmentation results.

Task Overview and flowchart:



The process illustrated in the diagram explains how an image is segmented into meaningful regions using a combination of **graph representation** and **clustering**:

1. Build Graph

Each pixel in the image is treated as a **node** in a graph.

Edges connect neighbouring pixels based on spatial proximity.

This step transforms the image into a structured network where relationships between pixels are explicitly modelled.

2. Convert to Features

Every pixel (node) is converted into a **feature vector** containing information such as:

Color values (R, G, B)

Spatial location (x, y)

These features transform image data from raw pixels into measurable attributes suitable for clustering.

3. Cluster (K-Means)

K-Means clustering is applied to these pixel feature vectors.

The algorithm groups pixels into clusters based on similarity:

Pixels with similar color and spatial characteristics fall into the same group.

After clustering, each node (pixel) is assigned a label representing its region membership.

4. Segmentation Result

The cluster membership information is mapped back to the image space.

Pixels belonging to the same cluster are colored identically.

The resulting image clearly shows different **segments** or regions that correspond to coherent visual structures (e.g., sky, water, vegetation).

Concept summary:

Algorithm / Model Used:

• **K-Means Clauster** – K-Means Clustering is an unsupervised algorithm that groups data into K distinct clusters based on similarity by iteratively updating cluster centroids.In this project, it is used to segment images by clustering pixels into meaningful regions based on their visual characteristics.

Input:

• **sample jpg or png image**

Output:

• Segmented image after using unsupervised ML algorithm.

Model Training:

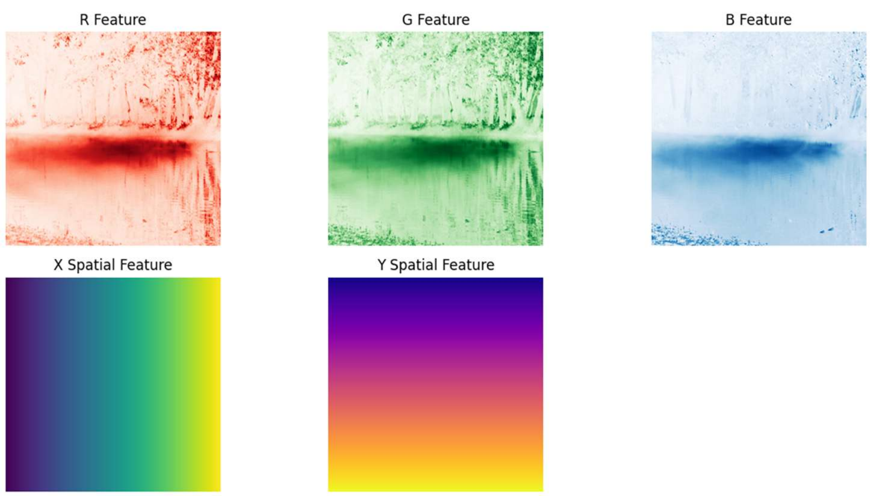
1. **Input Image Acquisition**
    - Read the input image (e.g., RGB or grayscale).
    - Optionally resize or normalize it for consistent processing.
2. **Preprocessing**
    - Convert to a convenient color space (e.g., RGB → HSV or Lab) if needed.
    - Flatten the image into a list of pixels, where each pixel becomes a data point.
3. **Feature Vector Construction**
    - For each pixel, build a feature vector such as:
        - Color components (R, G, B) or (L, a, b).
        - Optionally add spatial coordinates (x, y) to encourage spatially smooth clusters.
    - Result: a matrix of shape [number_of_pixels × number_of_features].
4. **Choose Number of Clusters (K)**
    - Decide how many segments you want in the final image (e.g., K = 3, 4, 5).
    - This K represents the number of regions/classes the image will be divided into.
5. **Initialize Cluster Centroids**
    - Randomly select K data points or use a method like K-Means++ to initialize K centroids in feature space.
    - These centroids represent the "center" of each cluster.
6. **Assignment Step (E-Step-like)**
    - For every pixel's feature vector, compute the distance (usually Euclidean) to each centroid.
    - Assign each pixel to the cluster with the nearest centroid.

- This gives a cluster label for every pixel.

7. **Update Step (M-Step-like)**
   - For each cluster, recompute its centroid as the mean of all feature vectors assigned to that cluster.
   - This updates the cluster centers to better match the assigned pixels.
8. **Iterate Until Convergence**
   - Repeat the **Assignment Step** and **Update Step**:
     - Reassign pixels to the nearest updated centroid.
     - Recompute centroids again.
   - Stop when either:
     - Cluster assignments stop changing significantly, or
     - A maximum number of iterations is reached.
9. **Generate Segmented Image**
   - Use the final cluster labels to recolor the image:
     - All pixels in the same cluster are given a common color or label.
   - The result is a segmented image where each region corresponds to one K-Means cluster.
10. **Evaluate and Visualize Results**
    - Visually inspect the segmentation result.
    - Optionally compute quantitative metrics (e.g., inertia, silhouette score, or compare with ground truth segmentation if available).

Result analysis and explanation:

1. **Feature Visualization (RGB + Spatial Feature Maps):**



This diagram explains *what information the clustering algorithm sees* and why segmentation is possible.
It introduces:

- The pixel color variation (R, G, B channels),
- The spatial coordinates (X, Y) that enforce region smoothness.
- This sets a foundation for readers to understand how clusters form.

These five diagrams show how the image pixels contribute information used by K-Means.
What they represent:

- **R, G, B feature maps** visualize intensity patterns in each color channel.
- **X and Y spatial features** encode pixel position.

Relation to silhouette scores:

- The visible contrast in RGB channels reveals **natural texture and intensity separation**, explaining why low K values (e.g., 3 clusters) work best.
- Including X and Y spatial features encourages clusters to form **spatially coherent regions** instead of scattered pixels.

Observation:

- Because different areas (sky, water, trees) show distinct feature patterns, K-Means can easily separate a few regions.
- However, the features **do not support many fine-grained separations**, which is why silhouette scores drop for K = 6–10.

2. **PCA Feature Distribution Plot**

After showing the raw features, PCA helps the reader visualize how those features behave in reduced space, revealing:

- How many natural groups exist,
- Why lower K values perform better.

This supports your later clustering results.

What it shows:

- Each point is a pixel projected into 2D after PCA.
- The distribution indicates how separable the pixel groups are in feature space.

Relation to silhouette scores:

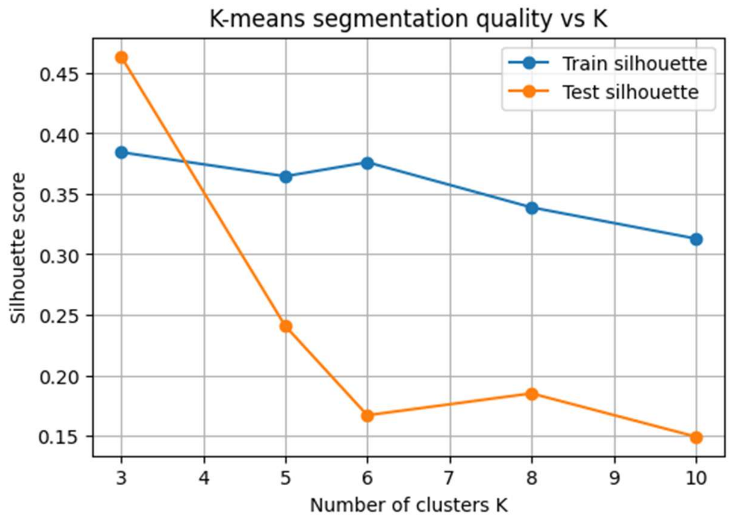
- We visibly see **three or so dense strokes/shapes**, meaning the data inherently forms around three clusters.
- There are **no clearly isolated 6–10 groupings**, supporting why segmentation performance declines with larger K.

Interpretation:

PCA confirms that:

- The feature space contains **few dominant groups**, agreeing with silhouette peak at K = 3.
- Trying to force more clusters splits existing clouds unnaturally, reducing separation quality.

3. **Silhouette Score Plot — "K-means segmentation quality vs K":**



Only after understanding the features and their separability does the silhouette curve make sense. This plot clearly:

- Confirms analytically what the previous diagrams hinted visually,
- Shows optimal K and performance degradation for higher K.

What it shows:

- The blue curve shows clustering quality on training data.
- The orange curve shows clustering quality on unseen (test) data.

Relation to silhouette scores:

- **K = 3 has the highest train (0.38) and test (0.46) silhouettes**, meaning the image naturally separates into three dominant regions.
- As K increases, both curves **decline steadily**, showing that forcing more clusters makes segmentation weaker.
- Low test silhouette scores for K ≥ 6 indicate that extra clusters lack strong structure and do not generalize.

Interpretation:

The plot visually confirms that the dataset supports **3 strong clusters** and more clusters dilute meaningful separation.

Connecting All Diagrams to Results:

DIAGRAM	WHAT IT REVEALS	LINK TO SILHOUETTE SCORES
Silhouette graph	Best performance at K = 3	Confirms natural clustering structure
RGB + spatial features	Strong contrasts only form few dominant regions	Explains why clusters beyond 3 weaken
PCA distribution plot	Only a few dense groups exist in feature space	Matches peak silhouette at K = 3

The clustering experiments were conducted for multiple values of K to evaluate segmentation quality using silhouette scores. The results consistently indicate that **K = 3 yields the highest performance**, with a training silhouette of **0.3844** and a test silhouette of **0.4637**. This suggests that the image space naturally organizes into three dominant regions, allowing K-Means to form compact, well-separated clusters.

As K increases, both training and testing silhouettes progressively decline. Values of **K = 5, 6, 8, and 10** show noticeably lower scores, especially on the testing data, indicating weaker generalization and over-segmentation. When more clusters are imposed than the data structure supports, the algorithm begins splitting coherent regions into smaller fragments, resulting in poorly defined boundaries and reduced separation between segments.

The **feature visualizations (RGB channels and spatial maps)** demonstrate strong variation across broad regions, supporting the idea that a limited number of clusters can capture dominant color and spatial patterns. Meanwhile, the **PCA scatter plot** reveals only a few dense groupings in feature space, further validating that segmentation performance peaks at low K values.

Overall, the results show that the dataset exhibits a **clear, low-dimensional structure**, and forcing excessive clusters leads to deterioration in segmentation quality. Thus, **K = 3 represents the optimal segmentation configuration** for this experiment, delivering the strongest boundary coherence and region distinctiveness.

Conclusion and future scope:

This project successfully demonstrated that K-Means clustering can perform meaningful image segmentation when appropriate feature representations, such as RGB intensities and spatial coordinates, are used. Silhouette score analysis confirmed that **K = 3** produced the best segmentation quality, indicating that the dataset naturally forms a small number of dominant visual regions. Increasing the number of clusters resulted in declining silhouette values, highlighting the limitations of forcing excessive segmentation and validating the importance of selecting an optimal K. Overall, the study shows that K-Means remains an effective baseline method for visual grouping when properly parameterized and evaluated.

Although the results are promising, there is substantial scope for improvement and extension. More advanced techniques such as Gaussian Mixture Models, Spectral Clustering, and deep-learning-based segmentation networks (e.g., U-Net, Mask R-CNN) could be explored to capture more complex region boundaries and semantic structure. Incorporating additional descriptors—like texture, edge response, or gradient patterns—could enhance region separability. Automated K-selection strategies or adaptive clustering criteria may also reduce manual tuning. Furthermore, pre-processing methods such as super pixel segmentation (SLIC) and feature weighting strategies could improve spatial coherence.

In summary, while K-Means provides a strong foundation for segmentation, future work can enhance accuracy, automation, and generalization by integrating richer features, smarter optimization, and more sophisticated clustering architectures.

Reference:

1. http://gnjatovic.info/imageprocessing/computer.vision.tasks.pdf
2. https://www.sciencedirect.com/science/article/abs/pii/S0020025522014633
3. Chat Gpt
4. Copilot