Term Project: Interactive Foreground-Background Segmentation using GrabCut with Gaussian Mixture Models and Graph Cuts

Debanjan Guha
Roll No: CE22B050
CS6870: Digital Video Processing
Indian Institute of Technology Madras

Sarvesh Shanbhag
Roll No: CE22B103
CS6870: Digital Video Processing
Indian Institute of Technology Madras

***Abstract*—This report presents an implementation of the GrabCut algorithm for interactive image segmentation using Gaussian Mixture Models (GMM) and graph cuts. We focus on accurately segmenting foreground objects from complex backgrounds with minimal user input. The core components of our method include initializing foreground and background GMMs, iterative energy minimization via max-flow/min-cut graph optimization, and user-guided refinement using a simple rectangle or mask. The segmentation pipeline is designed to be robust to color similarities and noise, and the report also discusses various implementation challenges, design choices, and potential improvements.**

I. Introduction

Image segmentation plays a critical role in computer vision applications such as photo editing, medical imaging, and object recognition. Traditional segmentation techniques often struggle in separating objects from backgrounds when textures, colors, or lighting conditions are similar. Interactive segmentation approaches address this challenge by incorporating user input to guide the segmentation process.

GrabCut is one such technique that combines user interaction, probabilistic modeling (GMMs), and energy minimization using graph cuts. Originally proposed by Rother et al., it improves upon standard graph-cut-based segmentation by iteratively refining pixel labels and model parameters to achieve cleaner object boundaries with fewer user annotations.

This project aims to implement the GrabCut algorithm from scratch, focusing on the following:

- Constructing and updating GMMs for foreground and background pixel distributions.
- Building an image-based graph where edge weights reflect both pixel color similarity and neighborhood connectivity.
- Applying the max-flow/min-cut algorithm to determine the optimal binary segmentation.
- Enabling user input through a rectangle or mask to initialize known foreground and background regions.

Our implementation emphasizes modular design and interpretability, making it suitable for further extension or integration with modern deep learning-based methods.

A. Problem Statement

How can graph-based image segmentation be improved by combining the theoretical edge-weight formulation from Boykov and Jolly's graph cuts with the iterative refinement capabilities of the GrabCut algorithm?

B. Motivation

Accurate and efficient segmentation is critical in scenarios where object boundaries must be precisely identified, such as in medical imaging or image editing. By integrating the edge-weight design proposed by Boykov and Jolly with the GrabCut framework, we aim to leverage both the optimal boundary definition of graph cuts and the iterative refinement and user interactivity of GrabCut, achieving improved segmentation quality in challenging real-world images.

II. Algorithmic Description

A. Graph-Based Segmentation Formulation

Consider a set of pixels P, with a neighborhood system N consisting of all unordered pairs $\{p, q\}$ of adjacent pixels. In the case of a 2D image, P includes all pixel locations, and N typically follows an 8-connected neighborhood.

Let the binary label assignment for all pixels be represented by the vector $L = (l_1, l_2, \ldots, l_{|P|})$, where each label $l_p \in \{\text{"obj"}, \text{"bkg"}\}$ assigns pixel $p \in P$ to either the object or the background class. The overall segmentation is defined by this labeling vector L, and its cost is represented by the energy function:

$$E(L) = \lambda \cdot R(L) + B(L) \tag{1}$$

- $R(L)$: Region term — penalizes assignments that do not conform to appearance models.
- $B(L)$: Boundary term — penalizes discontinuities between neighboring pixel labels.
- $\lambda \geq 0$: Regularization parameter that balances region fidelity and boundary smoothness.

To compute the region term $R(L)$, we use user-defined seed sets: $S_{\text{obj}} \subset P$ for foreground (object) and $S_{\text{bkg}} \subset P$ for background, such that $S_{\text{obj}} \cap S_{\text{bkg}} = \emptyset$. The appearance of pixels in each seed set is modeled using Gaussian Mixture

Models (GMMs). Let $\mathcal{G}_{\text{obj}}$ and $\mathcal{G}_{\text{bkg}}$ denote the GMMs trained on the intensities of S_{obj} and S_{bkg}, respectively.

To compute the region term $R(L)$, we use user-defined seed sets: $S_{\text{obj}} \subset P$ for foreground (object) and $S_{\text{bkg}} \subset P$ for background, such that $S_{\text{obj}} \cap S_{\text{bkg}} = \emptyset$. The appearance of pixels in each seed set is modeled using Gaussian Mixture Models (GMMs). Let $\mathcal{G}_{\text{obj}}$ and $\mathcal{G}_{\text{bkg}}$ denote the GMMs trained on the intensities of S_{obj} and S_{bkg}, respectively.

Each GMM is defined as:

$$\Pr(I_p \mid \mathcal{G}) = \sum_{k=1}^{K} w_k \cdot \mathcal{N}(I_p; \mu_k, \Sigma_k)$$

where:

- K is the number of Gaussian components,
- w_k is the weight of the k-th component,
- μ_k and Σ_k are the mean and covariance of the k-th Gaussian,
- $\mathcal{N}(I_p; \mu_k, \Sigma_k)$ is the multivariate Gaussian PDF evaluated at I_p.

For any pixel $p \in P$ with observed intensity I_p, the region cost is given by:

$$R_p(\text{"obj"}) = -\log \Pr(I_p \mid \mathcal{G}_{\text{obj}})$$
$$R_p(\text{"bkg"}) = -\log \Pr(I_p \mid \mathcal{G}_{\text{bkg}})$$

Therefore, the region cost for each pixel $p \in P$ is then:

$$R_p(\text{"obj"}) = -\log \left(\sum_{k=1}^{K} w_k^{\text{obj}} \cdot \mathcal{N}(I_p; \mu_k^{\text{obj}}, \Sigma_k^{\text{obj}}) \right)$$
$$R_p(\text{"bkg"}) = -\log \left(\sum_{k=1}^{K} w_k^{\text{bkg}} \cdot \mathcal{N}(I_p; \mu_k^{\text{bkg}}, \Sigma_k^{\text{bkg}}) \right)$$

Thus, the total region energy for the data term is:

$$R(L) = \sum_{p \in P} R_p(l_p)$$

The boundary term $B(L)$ measures the penalty for label transitions between neighboring pixels, also known as the smoothness term:

$$B(L) = \sum_{\{p,q\} \in N} B_{p,q} \cdot [l_p \neq l_q]$$

where $[l_p \neq l_q] = 1$ if $l_p \neq l_q$, and 0 otherwise. The penalty $B_{p,q}$ is defined to be higher when the pixel intensities I_p and I_q are similar and lower when they differ significantly:

$$B_{p,q} = \gamma \cdot \exp\left(-\frac{(I_p - I_q)^2}{2\sigma^2} \right) \cdot \frac{1}{\text{dist}(p,q)}$$

where γ is a scaling constant and $\text{dist}(p,q)$ is the Euclidean distance between pixels p and q.

This formulation encourages label boundaries to align with strong image gradients, promoting piecewise smooth segmentation while respecting object boundaries.

B. Graph Construction and Min-Cut Optimization

To compute the globally optimal segmentation minimizing $E(A)$, we construct a graph $G = (V, E)$, where:

- $V = P \cup \{S, T\}$ consists of one node per pixel and two terminals: source S ("object") and sink T ("background").
- $E =$ n-links $\cup$ t-links, where:
  - **n-links** connect each neighboring pair $\{p, q\} \in N$ with weight $B_{\{p,q\}}$.
  - **t-links** connect each pixel $p \in P$ to both terminals:

$$\text{weight}(p \to S) = R_p(\text{"obj"})$$
$$\text{weight}(p \to T) = R_p(\text{"bkg"})$$

The graph is undirected for simplicity, as we consider unordered pairs $\{p, q\}$. Using ordered (directed) edges (p, q) and (q, p) allows for asymmetric penalties depending on label transitions (e.g., object-to-background vs. background-to-object), but requires prior boundary knowledge and more complex graph structures.

Using the user-provided hard constraints:

$$\forall p \in O,\ A_p = \text{"obj"} \quad \text{and} \quad \forall p \in B,\ A_p = \text{"bkg"} \tag{2}$$

we solve for the global minimum of $E(A)$ via the min-cut/max-flow algorithm. The optimal cut partitions the graph into two disjoint sets: the set connected to S is labeled "object", and the set connected to T is labeled "background".

C. Gaussian Mixture Modeling (GMM)

To refine the likelihood estimates, we use Gaussian Mixture Models (GMMs) to approximate the color distributions of the object and background. This is implemented through the `GaussianMixture` class, which supports:

- **Component Assignment:** Assign each pixel to the most probable Gaussian component using maximum likelihood.
- **Parameter Update:** Re-estimate the component weights, means, and covariances via the EM algorithm.
- **Log-Likelihood Evaluation:** Used to compute region penalties in graph construction.

This GMM-based approach allows GrabCut to capture complex, multimodal color distributions in both regions and adapt dynamically during iteration.

D. Iterative GrabCut Refinement

The segmentation process proceeds iteratively as follows:

1) **User Initialization:** A rectangle or scribbles specify definite foreground/background seeds.
2) **Initial GMM Fitting:** Fit GMMs to seed pixels.
3) **Graph Construction:** Use updated region and boundary costs.
4) **Min-Cut Computation:** Solve for optimal segmentation.
5) **GMM Update:** Reassign pixels and update parameters based on new labels.

Repeat steps 3–5 until convergence.

This interactive approach allows users to refine the result with minimal effort, supporting both hard constraints and automatic adaptation of color models.

III. Output

The implemented algorithm yielded the following outcomes:

- The parts of the image in the bounding box that fall under the background category were detected accurately
- Most parts of the image that were supposed to be categorized as foreground were detected accurately; however, a few of them, which shared similar intensities with the background, were eliminated (as seen in Fig. 1.a and b)
- In Fig. 1.b, due to poor lighting, objects like the cell phone and parts of the object like the laptop's screen were initially categorized as a part of the background, which were later manually categorized as foreground using the scribbling option enabled in our code.

IV. Expected methods of improving performances and alternative methods

Alpha matting significantly enhances GrabCut segmentation by introducing soft transitions between foreground and background regions, which are often crucial in accurately capturing object boundaries. While GrabCut performs hard segmentation by classifying pixels strictly as foreground or background using color-based models and graph cuts, it often struggles with complex boundaries such as hair, fur, or semi-transparent regions. Alpha matting addresses this limitation by estimating an opacity value (alpha) for each pixel, where $\alpha = 1$ indicates full foreground, $\alpha = 0$ indicates background, and $0 < \alpha < 1$ corresponds to mixed regions. By applying alpha matting to the trimap generated from GrabCut, typically consisting of definite foreground, definite background, and unknown regions, the segmentation result becomes significantly more refined. This combination allows for smoother edges, better preservation of fine details, and more natural compositing of segmented objects into new backgrounds.

Alternatively, we also attempted to implement the Boykov-Kolmogorov graph cut segmentation method. However, due to the limited availability of reliable implementations and references, we encountered challenges in the coding process and were unable to reach a definitive outcome.

V. Contribution by individual team members

- Sarvesh: Worked on implementing the GrabCut algorithm and gathered relevant papers and references.
- Debanjan: Explored the Boykov-Kolmogorov Graph Cut algorithm but discontinued it due to challenges. Provided some assistance with the GrabCut implementation, resource collection, and report writing.

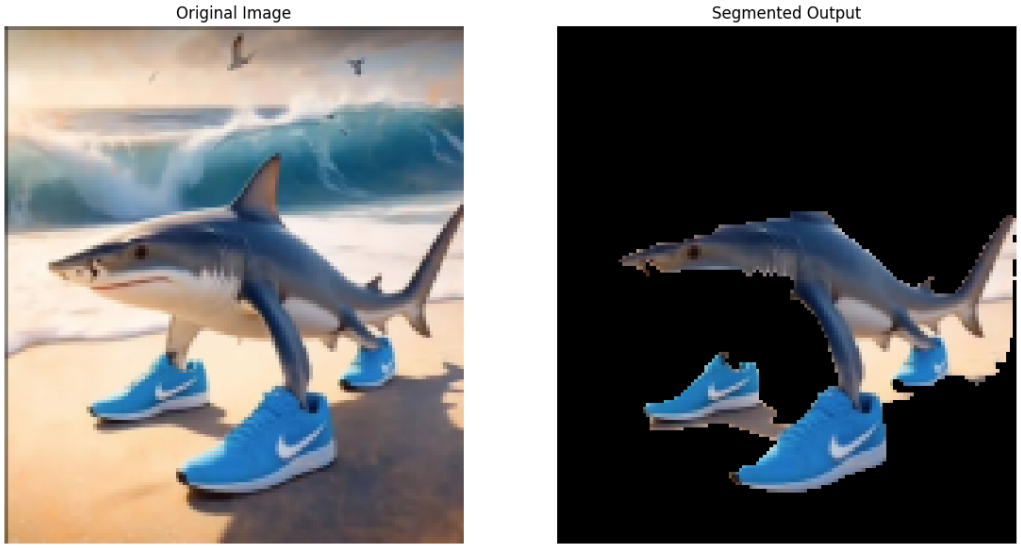
References

[1] Yuri Y. Boykov and Marie-Pierre Jolly , *Interactive Graph Cuts for Optimal Boundary & Region Segmentation of Objects in N-D Images*, 2001. [Online]. Available: https://www.csd.uwo.ca/~yboykov/Papers/iccv01.pdf.

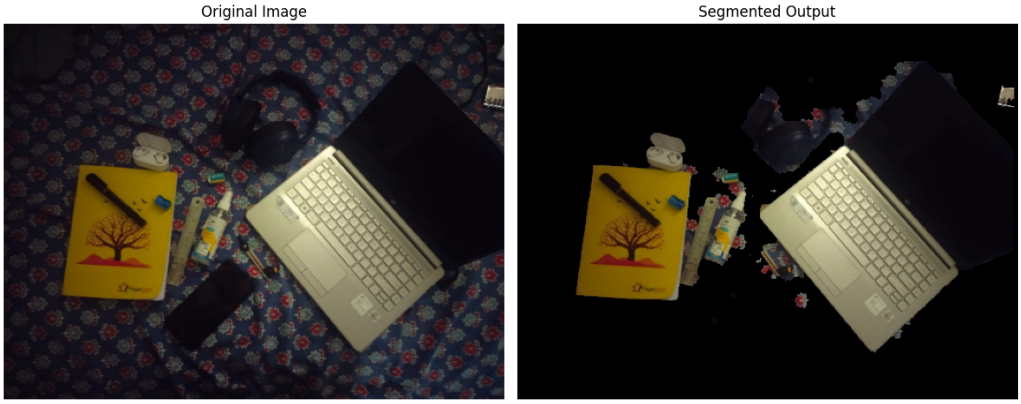
[2] Carsten Rother, Vladimir Kolmogorov and Andrew Blake, *"GrabCut" — Interactive Foreground Extraction using Iterated Graph Cuts*, 2004. [Online]. Available: https://pub.ista.ac.at/~vnk/papers/grabcut_siggraph04.pdf.

[3] Yuri Boykov and Vladimir Kolmogorov, *An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision*, 2004. [Online]. Available: https://www.csd.uwo.ca/~yboykov/Papers/pami04.pdf.

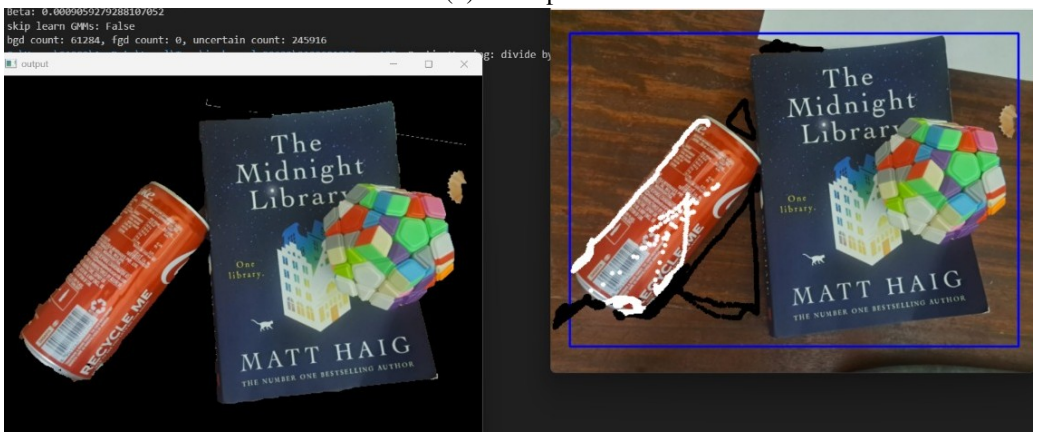
[4] Graph Cut Code Reference https://networkx.org/documentation/stable/_modules/networkx/algorithms/flow/boykovkolmogorov.html https://github.com/Gnimuc/BKMaxflow.jl/tree/master.



(a) Example 1



(b) Example 2



(c) Example 3

Fig. 1: Original and segmented images