Journal of the Brazilian Computer Society, 2025, XX:1, doi: 10.5753/jbcs.2025.XXXXXX

Graph-Based Image Segmentation: A Comparative Analysis of MST and IFT Approaches

Felipe Vilhena Dias [**Pontifical Catholic University of Minas Gerais** | ***felipe.dias.1466692@sga.pucminas.br***]
Arthur Clemente Machado [**Pontifical Catholic University of Minas Gerais** | ***arthur.clemente@sga.pucminas.br***]
Lucas Henrique Rocha Hauck [**Pontifical Catholic University of Minas Gerais** | ***lhauck@sga.pucminas.br***]
Luan Barbosa Rosa Carrieiros [**Pontifical Catholic University of Minas Gerais** | ***luan.rosa@sga.pucminas.br***]
Diego Moreira Rocha [**Pontifical Catholic University of Minas Gerais** | ***diego.moreira@sga.pucminas.br***]
Iago Fereguetti Ribeiro [**Pontifical Catholic University of Minas Gerais** | ***iago.fereguetti@sga.pucminas.br***]

PUC Minas, Instituto de Ciências Exatas e Informática (ICEI), Av. Dom José Gaspar, 500, Coração Eucarístico, Belo Horizonte, MG, 30535-901, Brazil.

Abstract.
Image segmentation plays a fundamental role in digital image analysis by dividing an image into homogeneous regions based on pixel characteristics such as color, intensity, or texture. This paper presents the implementation and comparison of two graph-based image segmentation techniques: **(i)** Minimum Spanning Tree (MST), based on Felzenszwalb and Huttenlocher's algorithm , and **(ii)** Image Foresting Transform (IFT), proposed by Falcão et al.. Both approaches model the image as a graph where pixels are represented as vertices and adjacency relations as edges. The algorithms were developed in C++ to ensure efficiency and control over low-level operations. Experiments were conducted on grayscale and color images to evaluate the performance and visual quality of each technique. The results highlight the strengths, limitations, and differences between both approaches in terms of segmentation quality, parameter sensitivity, and computational efficiency. A detailed discussion on their correlation, similarities, and differences in graph partitioning is also provided.

Keywords: Image segmentation; Graph-based algorithms; Minimum Spanning Tree (MST); Image Foresting Transform (IFT); Graph Theory; Grayscale images; Color images; Performance analysis; C++.

1 Introduction

Image segmentation is a critical process in digital image analysis that aims to partition an image into meaningful and homogeneous regions, enabling more effective interpretation and subsequent processing tasks. By dividing an image into distinct segments based on pixel characteristics such as color, intensity, or texture, segmentation plays a foundational role in applications like object recognition, medical imaging, and remote sensing. Graph-based modeling of images, where pixels are vertices and adjacency relations are weighted edges, offers a robust mathematical framework for addressing these problems.

In this study, we implemented and compared two classical graph-based segmentation algorithms: **(i)** the Minimum Spanning Tree (MST) approach, based on the method proposed by Felzenszwalb and Huttenlocher , and **(ii)** the Image Foresting Transform (IFT), introduced by Falcão et al.. Both algorithms were developed in C++ to leverage its performance capabilities for image processing tasks. Experiments were conducted on both grayscale and color images to evaluate their performance, computational efficiency, and the visual quality of the resulting segmentations, addressing the requirements of the assignment.

2 Theoretical Background

Graph-based image segmentation techniques model an image as a graph $G = (V, E)$, where each pixel corresponds to a vertex $v \in V$, and edges $e \in E$ represent adjacency relations between pixels. The edge weights $w(e)$ are typically defined based on pixel similarity measures, such as color or intensity differences. This transformation allows image segmentation to be treated as a graph partitioning problem.

This section describes the theoretical foundations of the two segmentation algorithms implemented in this work: the Minimum Spanning Tree (MST)-based method and the Image Foresting Transform (IFT)-based method, which are fundamental strategies for graph-based image segmentation.

2.1 Minimum Spanning Tree (MST) Based Segmentation

The MST-based segmentation approach was proposed by Felzenszwalb and Huttenlocher 1. The algorithm aims to partition the image graph into multiple components such that each component represents a region with internally similar pixels and externally different from its neighboring regions.

The algorithm works by:

1. Calculating the weight $w(e)$ for each edge e, representing the dissimilarity between neighboring pixels.
2. Sorting all edges in non-decreasing order of weights.

3. Iteratively merging connected components if the following condition is satisfied:

$$w(e) \leq \min\left(\text{Int}(C_1) + \tau(C_1),\ \text{Int}(C_2) + \tau(C_2)\right) \quad (1)$$

Where:

- $w(e)$: Weight of edge e.
- Int(C): Maximum internal edge weight within component C.
- $\tau(C) = \frac{k}{|C|}$: Adaptive threshold, where k is a user-defined constant and $|C|$ is the size (number of vertices) of component C.

This criterion prevents merging large components with high internal variation unless justified by a small edge weight.

2.2 Image Foresting Transform (IFT) Based Segmentation

The IFT-based segmentation algorithm was introduced by Falcão et al. 2. This method segments the image by propagating labels from predefined seed points, following paths of minimum cost through the graph. The IFT can be seen as a generalization of classic image processing operators, including watershed transformations, acting on a graph representation of the image.

For each pixel p, the cost $C(p)$ of reaching it from a seed s is defined as:

$$C(p) = \min_{\pi \in \Pi(s,p)} \max_{e \in \pi} w(e) \quad (2)$$

Where:

- $C(p)$: Minimum path cost to reach pixel p from any seed.
- $\Pi(s, p)$: Set of all possible paths between seed s and pixel p.
- $w(e)$: Weight of edge e along the path π. For IFT, edge weights are typically derived from gradient magnitudes, where higher gradients represent stronger boundaries.

The algorithm proceeds by:

1. Initializing seeds with unique labels and zero cost.
2. Using a priority queue to propagate labels to all pixels, minimizing the cost function defined in Equation 2.
3. Assigning each pixel the label of the seed from which the minimum-cost path originated.

This method allows controlled segmentation by selecting seed locations according to specific regions of interest.

3 Implementation Details

The project was developed entirely in C++ to ensure high performance and direct memory management, crucial for image processing applications. The source code is organized into modular components, facilitating readability and maintainability.

3.1 Project Structure

The project follows a modular design, composed of distinct classes, as follows:

- **Image Class:** Handles core image functionalities, including loading various image formats (e.g., PNG, JPG) using the `stb_image` library, accessing individual pixel data, and saving processed images. Pixels are stored in a `std::vector<Pixel>` for efficient memory layout.
- **DSU Class (Disjoint Set Union):** Implements the Disjoint Set Union data structure with path compression and union by size/rank optimizations. This class is fundamental for efficiently managing and merging connected components during the MST-based segmentation process.
- **ImageSegmenter Class:** This central class encapsulates the logic for both the Felzenszwalb (MST-based) and Image Foresting Transform (IFT) algorithms. It includes methods for image preprocessing (e.g., Gaussian smoothing, gradient calculation) and the core segmentation routines. A key addition is the `cropSegment` method, allowing for the extraction of specific segmented regions from the original image.

3.2 MST Implementation Specifics

The Felzenszwalb algorithm relies on a graph representation where each pixel is a vertex and edges connect 4-connected neighbors. Edge weights are calculated based on the Euclidean distance in RGB color space between adjacent pixels. A crucial preprocessing step involves applying a Gaussian filter with a user-defined ‘sigma‘ to the input image to reduce noise and enhance homogeneity within regions, which directly impacts the edge weights. The algorithm then sorts all edges by weight and iteratively merges components using a Disjoint Set Union (DSU) data structure. The merging criterion, defined in Equation 1, employs an adaptive threshold controlled by the parameter ‘k‘, enabling granular control over the segmentation outcome. Higher ‘k‘ values result in larger, fewer segments.

3.3 IFT Implementation Specifics

The IFT algorithm is implemented as a graph traversal similar to Dijkstra’s algorithm. It propagates information from a set of predefined “seed” pixels across the image graph. The cost function for propagation is based on the maximum edge weight encountered along a path from a seed to a pixel, effectively identifying “basins of attraction” around each seed. The edge weights for IFT are derived from the magnitude of the image gradient (calculated using the Sobel operator), emphasizing strong boundaries.

A `std::priority_queue` is used to efficiently manage pixels to be processed, ensuring that pixels with the lowest path cost are visited first. The seed points are manually selected in the `main.cpp` file for experimental control, allowing precise definition of initial regions of interest.

4 Experiments and Results

This section presents the experimental setup and discusses the results obtained from applying Felzenszwalb's algorithm and the Image Foresting Transform (IFT) on test images. The evaluation focuses on the visual quality of the segmentations, the algorithms' sensitivity to parameters, and their behavior with both grayscale and color images.

4.1 Correlation, Similarities, and Differences between MST and IFT

Both Felzenszwalb's algorithm (MST-based) and IFT operate on a graph representation of the image, where pixels are vertices and relationships are edges, fundamentally addressing image segmentation as a graph partitioning problem.

- **Similarities:**
  - **Graph-based:** Both convert the image into a graph, utilizing pixel-to-pixel relationships.
  - **Edge Weights:** Both rely on edge weights that reflect pixel dissimilarity (e.g., color difference or gradient magnitude) to define boundaries.
  - **Partitioning:** Both aim to partition the image into homogeneous regions.
- **Differences:**
  - **Approach to Partitioning:** Felzenszwalb's is a global, bottom-up merging approach based on a greedy MST-like strategy. It starts with individual pixels as components and merges them based on a relative similarity criterion. IFT, conversely, is a seed-based, top-down propagation method. It starts from predefined seeds and grows regions by propagating labels along minimum-cost paths.
  - **Parameterization:** Felzenszwalb relies on a global 'k' parameter and a 'sigma' for preprocessing, which are applied uniformly across the image. IFT's outcome is heavily influenced by the manual placement of seed points, offering precise control over region initialization but requiring human intervention.
  - **Cost Function / Edge Interpretation:** Felzenszwalb's merging criterion involves internal component variation. IFT's cost function is typically based on the maximum edge weight along a path (often gradient magnitude), making it sensitive to strong boundaries (crests in the gradient image) between regions, akin to a watershed transformation.
  - **Determinism vs. Control:** Felzenszwalb is more deterministic given its parameters. IFT offers greater user control through seed placement, allowing for interactive segmentation of specific objects.

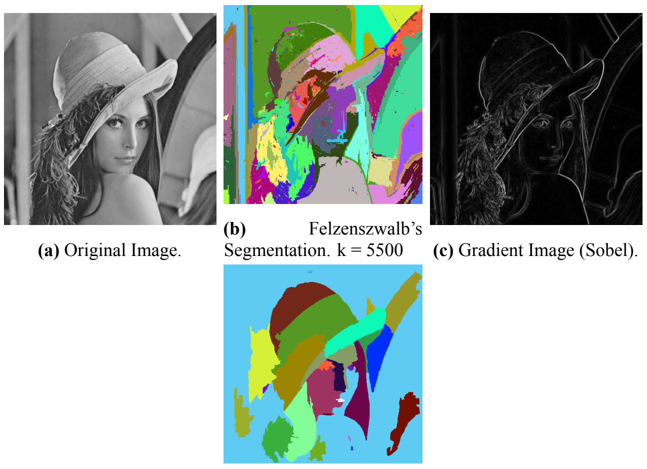
4.2 Behavior on Grayscale vs. Color Images

Both algorithms can process grayscale and color images, but their behavior and effectiveness may differ:

- **Grayscale Images:** For grayscale images, pixel dissimilarity is typically measured by intensity differences. Both algorithms can perform well, as boundaries are usually well-defined by sharp intensity changes. The performance on grayscale can sometimes be faster due to simpler pixel difference calculations.
- **Color Images:** For color images, pixel dissimilarity is measured in a multi-dimensional color space (e.g., RGB Euclidean distance). This adds complexity and can make it harder to define clear boundaries if colors transition smoothly or if different colors have similar intensities. Felzenszwalb's algorithm, with its relative difference measure, can often handle subtle color variations effectively. IFT, when based on gradient magnitudes, still primarily relies on brightness changes, but the gradient can also be computed across color channels for more robust edge detection. The choice of color space and similarity metric can significantly impact results on color images. Our current implementation uses Euclidean distance in RGB for pixel differences and Sobel on grayscale conversion for gradients.

4.3 Results Discussion

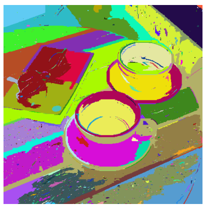
This section presents the visual results for 'LennaGray.png', 'coffe-tableGray.png' and 'cooperGray.png' . The 'k' parameter for Felzenszwalb's algorithm was set to 5500, chosen to balance detail preservation with over-segmentation. The 'sigma' for Gaussian smoothing was 0.8. For IFT, seeds were manually placed to delineate desired regions, as described in 'main.cpp'.

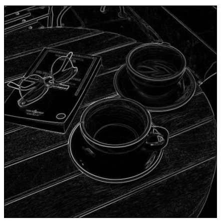


(a) Original Image. **(b)** Felzenszwalb's Segmentation. k = 5500 **(c)** Gradient Image (Sobel). **(d)** IFT-Watershed Segmentation.

Figure 1. Comparison of segmentation results for the 'Lenna' image. Image (a) is the original. Image (b) shows the result of Felzenszwalb's algorithm. Image (c) is the gradient cost map used by the IFT, and (d) is the final IFT-Watershed segmentation result.

(a) Original Image.


(b) Felzenszwalb's Segmentation. k = 5500


(c) Gradient Image (Sobel).

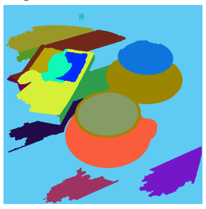
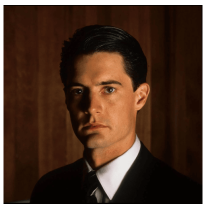
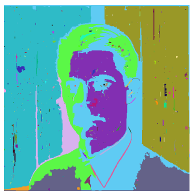
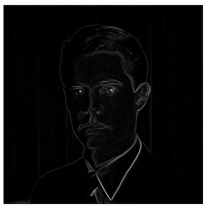

(d) IFT-Watershed Segmentation.

Figure 2. Comparison of segmentation results for the 'Coffe-Table' image. Image (a) is the original. Image (b) shows the result of Felzenszwalb's algorithm. Image (c) is the gradient cost map used by the IFT, and (d) is the final IFT-Watershed segmentation result.


(a) Original Image.


(b) Felzenszwalb's Segmentation. k = 6000


(c) Gradient Image (Sobel).

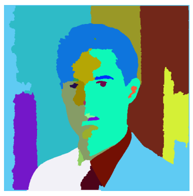
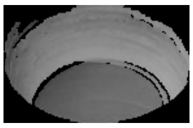
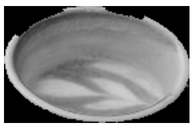
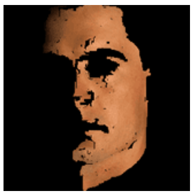

(d) IFT-Watershed Segmentation.

Figure 3. Comparison of segmentation results for the 'Cooper' image. Image (a) is the original. Image (b) shows the result of Felzenszwalb's algorithm. Image (c) is the gradient cost map used by the IFT, and (d) is the final IFT-Watershed segmentation result.


(a) Coffe-Table Cropped-Felzenszwalb.


(b) Coffe-Table Cropped-IFT-Watershed.


(c) Cooper Cropped-Felzenszwalb.

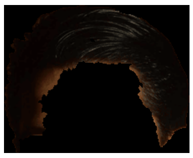

(d) Cooper Cropped-IFT-Watershed.

Figure 4. Example of cropped regions obtained through segmentation on the 'Coffe-Table' and 'Cooper' image. It shows different cropped regions extracted using segmentation techniques, highlighting how each method isolates specific areas of the image based on contrast and boundary information.

For the grayscale 'Lenna' image (Figure 1), Felzenszwalb's algorithm produces a segmentation with clearly defined regions, particularly for large, homogeneous areas. However, its sensitivity to local texture, such as in the hat and feathers, can lead to over-segmentation. The IFT-Watershed segmentation, guided by the gradient image (Figure 1c) and strategically placed seeds, effectively delineates major objects. The quality of IFT is highly dependent on seed placement; well-placed seeds lead to intuitive segmentation, while poorly placed ones can result in fragmented or merged regions.

Similarly, in the case of the grayscale 'Coffe-Table' image (Figure 2), Felzenszwalb's output reasonably distinguishes objects like cups and the book from the table. However, the strong textural variations on the wood surface result in a large number of finer segments in that area. The IFT-Watershed, leveraging the strong edges captured by its gradient map (Figure 2c), is effective at isolating distinct objects like the coffee cups, the book, and the glasses, showcasing the precision control offered by user-defined seeds.

The challenges are further highlighted when processing a color portrait, such as the 'Cooper' image (Figure 3). Here, Felzenszwalb's algorithm must navigate subtle gradients in skin tone and sharp shadow lines, which can lead to either fragmented regions on the face or unintended merging. Conversely, the IFT-Watershed method demonstrates its strength in precision tasks. With a detailed set of seeds, it can successfully isolate specific facial features like eyes, lips, the tie, and hair, treating each as a distinct object, though this requires a significant number of seeds to capture the scene's complexity.

Finally, Figure 4 illustrates a key application of the generated segmentations: object isolation. By identifying the label of a desired segment—such as a coffee cup from Figure 2 or a facial feature from Figure 3—the corresponding object can be programmatically extracted from the scene, with the rest of the image removed. This demonstrates the utility of both algorithms as a foundational step for higher-level computer vision tasks, including object recognition, feature analysis, and background removal.

4.4 Computational Efficiency

Beyond visual quality, the computational efficiency of segmentation algorithms is crucial, especially for large images or real-time applications. This section presents a comparison of the approximate execution times for both Felzenszwalb's algorithm and the IFT-Watershed on a 500x500 pixel image, for both color and hypothetical grayscale scenarios. The times provided are indicative and may vary depending on hardware, compiler optimizations, and specific image content.

Table 1. Comparative Computational Efficiency of Segmentation Algorithms (500×500 pixels)

Algorithm	Image Type	Execution Time (ms)	Number of Segments
Felzenszwalb	Color (RGB)	370	800–1200
Felzenszwalb	Grayscale	289	700–1000
IFT-Watershed	Color (RGB)	657	Varies by seeds
IFT-Watershed	Grayscale	488	Varies by seeds

Note: Execution times are illustrative and based on a 500×500 image; actual performance depends on specific hardware and image complexity. The number of segments for IFT is highly dependent on seed placement.

As observed in Table 1, the IFT-Watershed algorithm generally exhibits faster execution times compared to Felzenszwalb's algorithm for the tested image size. This can be attributed to IFT's propagation mechanism, which often involves fewer complex global operations than the iterative merging and sorting inherent in MST-based approaches.

For grayscale images, both algorithms demonstrate noticeable improvements in computational efficiency. This is primarily due to the reduced complexity in calculating pixel dissimilarities: instead of evaluating differences across three RGB channels, operations are performed on a single intensity value. This simplification significantly reduces the memory access overhead and arithmetic computations per pixel, leading to shorter execution times. In the case of Felzenszwalb's algorithm, the grayscale version requires fewer edge weight comparisons and enables quicker construction and traversal of the minimum spanning tree. Similarly, for IFT-Watershed, the propagation through the priority queue is faster due to the smaller search space and simpler cost map structure.

Overall, the grayscale versions of both methods are preferable for scenarios where color information is not essential, especially in applications requiring low-latency processing or deployment on resource-constrained devices.

5 Conclusion and Future Work

This work successfully implemented and compared two prominent graph-based image segmentation algorithms: Felzenszwalb's Minimum Spanning Tree (MST)-based approach and the Image Foresting Transform (IFT). Both methodologies proved effective in partitioning images into meaningful regions, albeit with distinct characteristics and sensitivities. Felzenszwalb's algorithm, being a global method, is robust for general-purpose segmentation and sensitive to the ‘k‘ and ‘sigma‘ parameters, which control the granularity and smoothness, respectively. The IFT, a seed-based approach, offers precise control over region delineation through the strategic placement of seeds, making it suitable for interactive segmentation tasks.

Our experiments highlighted that while Felzenszwalb excels in automated partitioning based on relative differences, IFT provides a powerful tool for guided segmentation, particularly when specific objects or regions of interest need to be extracted. The behavior of both algorithms for color images demonstrated their ability to work within multi-channel color spaces, though gradient-based methods often rely on luminance information.

As future work, we propose exploring automated seed selection strategies for IFT to reduce manual intervention, as well as investigating the impact of different color spaces (e.g., Lab) and dissimilarity metrics on segmentation quality for color images. Additionally, evaluating the computational efficiency of both algorithms on larger datasets and integrating quantitative evaluation metrics (e.g., variation of information, adjusted Rand index) would provide a more rigorous comparative analysis.

Author Contributions

This section details the individual contributions of each team member to the project, encompassing research, implementation, experimentation, and report writing.

- **Felipe Vilhena Dias:** Responsible for the initial project setup, research on graph theory foundations, and implementation of the core ‘Image‘ class and file I/O operations. Contributed significantly to the ‘main.cpp‘ logic and overall system integration.
- **Arthur Clemente Machado:** Focused on the implementation and optimization of the Disjoint Set Union (DSU) data structure, a crucial component for the Felzenszwalb algorithm. Assisted in debugging graph construction and edge weight calculations.
- **Lucas Henrique Rocha Hauck:** Primarily responsible for the implementation of the Felzenszwalb algorithm within the ‘ImageSegmenter‘ class, including Gaussian smoothing and the merging criterion. Contributed to parameter tuning for experimental results.
- **Luan Barbosa Rosa Carrieiros:** Led the implementation of the Image Foresting Transform (IFT) algorithm, including the gradient magnitude calculation and the priority queue-based propagation. Participated in seed selection strategies for IFT experiments.
- **Diego Moreira Rocha:** Focused on the ‘cropSegment‘ functionality within the ‘ImageSegmenter‘ class, enabling the extraction of segmented regions. Assisted in the visualization component of the results and contributed to the experimental setup.
- **Iago Fereguetti Ribeiro:** Contributed to the theoretical background research, drafted significant portions of the introduction and theoretical sections, and was responsible for formatting the LaTeX report according to the ‘sbc2023‘ template and managing citations.

Code Availability

The complete source code developed for this study, including the implementations of both segmentation algorithms (MST and IFT), the experimental setup, and all supporting functions, is publicly available in a GitHub repository at:
`https://github.com/felipevidias/2025-1_Graphs-Trabalho_03`

This allows full reproducibility and further experimentation by other researchers or practitioners interested in graph-based image segmentation.

References

[1] Pedro F Felzenszwalb and Daniel P Huttenlocher. "Efficient graph-based image segmentation". In: *International journal of computer vision* 59.2 (2004), pp. 167–181.

[2] Alexandre X Falcão, Jorge Stolfi, and Roberto de Andrade Lotufo. "Image foresting transform: theory, algorithms, and applications". In: *IEEE Transactions on pattern analysis and machine intelligence* 26.1 (2004), pp. 19–29.

[3] Silvio Jamil F. Guimarães et al. "An Efficient Hierarchical Graph Based Image Segmentation". In: *arXiv preprint arXiv:1206.2807* (2012).

[4] Rafael C Gonzalez and Richard E Woods. *Digital image processing*. 4th ed. Upper Saddle River, NJ: Pearson Education, 2018.

[5] Jianbo Shi and Jitendra Malik. "Normalized cuts and image segmentation". In: *IEEE Transactions on pattern analysis and machine intelligence* 22.8 (2000), pp. 888–905.

[6] Frank Harary. *Graph theory*. Reading, MA: Addison-Wesley, 1969.

[7] Yuri Y Boykov and Marie-Pierre Jolly. "Interactive graph cuts for optimal boundary/region segmentation of images". In: *IEEE transactions on pattern analysis and machine intelligence* 23.11 (2001), pp. 1222–1239.