Project Report on

“Efficient Graph-Based Image Segmentation”

CSD357: Image Processing and Its Applications

Authors : Vaishnav Varma - 2110110574

Aryaman Pattnayak - 2110110137

Aditya Katiyar - 2110110659

# Abstract:

In this project report, we analyze and implement the paper [Efficient Graph-Based Image Segmentation](https://cs.brown.edu/people/pfelzens/papers/seg-ijcv.pdf) by Pedro F. Felzenszwalb from the Artificial Intelligence Lab, Massachusetts Institute of Technology and Daniel P. Huttenlocher from the Computer Science Department, Cornell University.

# Introduction

Image segmentation is a fundamental process in image processing that involves partitioning an image into distinct, meaningful regions or objects. The primary goal of segmentation is to simplify the representation of an image, making it easier to analyze and extract relevant information. This crucial step lays the foundation for numerous applications in computer vision, enabling tasks such as object recognition, scene understanding, and image editing.

Based on the performance of existing methods of image segmentation, the objective of this paper has been to develop an algorithm for image segmentation that :

* Capture perceptually important groupings or regions, which often reflect global

aspects of the image.

* Be highly efficient, running in time nearly linear in the number of image pixels.

# Literature Review

The authors of this article propose a new algorithm for image segmentation that is based on a graph representation of the image. In this graph, each pixel is represented as a node, and edges connect neighboring pixels. The weight of an edge is a measure of the dissimilarity between the two pixels that it connects.

Unlike previous methods, this technique adaptively adjusts the segmentation criterion according to the degree of variability in neighboring regions of the image. This results in a method that, while making greedy decisions, can be shown to obey certain non-obvious global properties.

The method that they propose measures the evidence for a boundary between two regions by comparing two quantities: one based on intensity differences across the boundary, and the other based on intensity differences between neighboring pixels within each region.

Intuitively, the intensity differences across the boundary of two regions are perceptually important if they are large relative to the intensity differences inside at least one of the regions. An algorithm is developed which computes segmentations using this idea.

For some context, graph-based image segmentation techniques generally represent the problem in terms of a graph where each node corresponds to a pixel in the

image, and the edges in E connect certain pairs of neighboring pixels. A weight is associated with each edge based on some property of the pixels that it connects, such as their image intensities. Depending on the method, there may or may not be an edge connecting each pair of vertices. The earliest graph-based methods use fixed thresholds and local measures in computing a segmentation.

We take a graph-based approach to segmentation. Let G = (V, E) be an undirected graph with vertices , the set of elements to be segmented, and edges ∈ corresponding to pairs of neighboring vertices. Each edge ∈ has corresponding weight , which is a non-negative measure of the dissimilarity between neighboring elements vi and vj.

In the case of image segmentation, the elements in V are pixels and the weight of an edge is some measure of the dissimilarity between the two pixels connected by that edge (e.g., the difference in intensity, color, motion, location or some other local attribute).

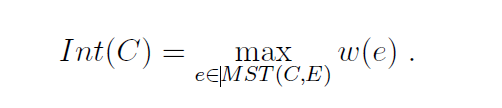
In the graph-based approach, a segmentation S is a partition of V into components such that each component (or region) C ∈ S corresponds to a connected component in a graph G’ = (V, E’), where E’ ⊆ E. In other words, any segmentation is induced by a subset of the edges in E.

For a segmentation, in general we want the elements in a component to be similar, and elements in different components to be dissimilar. This means that edges between two vertices in the same component should have relatively low weights, and edges between vertices in different components should have higher weights.

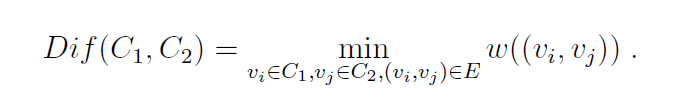
### Partition Strategy

Define a predicate, D, for evaluating whether or not there is evidence for a boundary between two components in a segmentation (two regions of an image). This predicate is based on measuring the dissimilarity between elements along the boundary of the two components relative to a measure of the dissimilarity among neighboring elements within each of the two components. The resulting predicate compares the inter-component di®erences to the within component di®erences and is thereby adaptive with respect to the local characteristics of the data.

Define the internal difference of a component C ⊆ V to be the largest weight in the minimum spanning tree of the component, MST(C,E), i.e.,



Defne the difference between two components C1,C2 ⊆ V to be the minimum weight edge connecting the two components, i.e.,

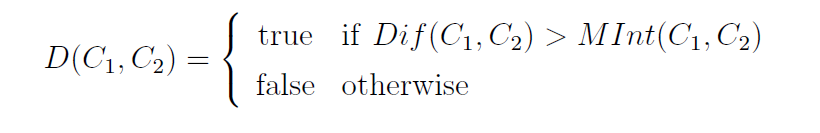


If there is no edge connecting C1 and C2, let

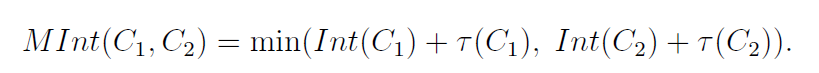
The region comparison predicate evaluates if there is evidence for a boundary between a pair or components by checking if the difference between the components, , is large relative to the internal difference within at least one of the components, and .

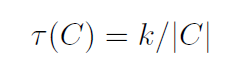
A threshold function is used to control the degree to which the difference between components must be larger than minimum internal difference.

Define the pairwise comparison predicate as:



where the minimum internal difference, , is defined as:



The threshold function τ controls the degree to which the difference between two components must be greater than their internal differences in order for there to be evidence of a boundary between them (D to be true).A threshold function based on the size of the component is used:  


where denotes the size of , and is some constant parameter.

# Methodology

Here is the segmentation algorithm implemented:

Input:

* A graph G = (V, E) with n vertices and m edges.
* A constant parameter k.

Output:

* A partition of V into segments S=(C1, C1, ...)

Steps:

1. Consider each vertex a single element component.
2. Initialize each component Ci with Int(Ci) = 0.
3. Sort all edges e ∈ E into (e1, e2, ..., em) according to their weights in a non-decreasing order.
4. Iteration Step q=(1, m):
   1. Take step eq = (vi, vj), where vi = Ci and vj = Cj.
   2. If Cj != Cj
      1. If boundary predicate D(Cj, Cj) = false, merge all the components Ci and Cj.
      2. If Ci and Cj are merged, Int(Ci ∪ Cj) = w(eq).
   3. q = q + 1

Merge condition in the iteration step defined as *D*(*Ci*,*Cj*) = *false* occurs when

*Dif*(*Ci*,*Cj*) ≤ *min*(*Int*(*Ci*)+*τ*(*Ci*), *Int*(*Cj*)+*τ*(*Ci*)).

This implies that *Dif*(*Ci*,*Cj*) ≤ *Int*(*Ci*) + *k/τ*(*Ci*)

Or *Dif*(*Ci*,*Cj*) ≤ *Int*(*Cj*) + *k/τ*(*Cj*)

The condition can also be written as: *w*(*eq*)≤*Int*(*Ci*) + *k*/*τ*(*Ci*)

Or *w*(*eq*)≤*Int*(*Cj*) + *k*/*τ*(*Cj*)

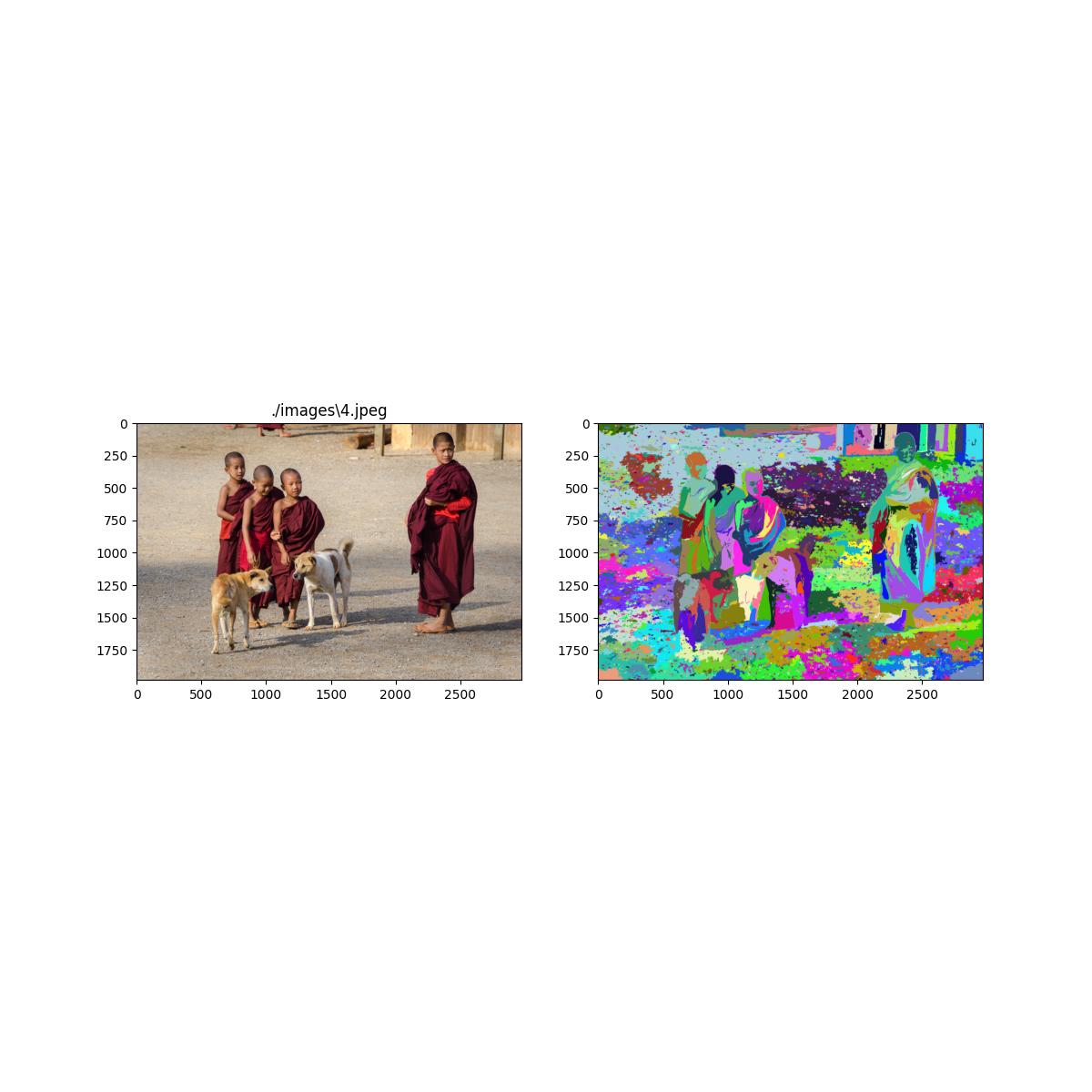
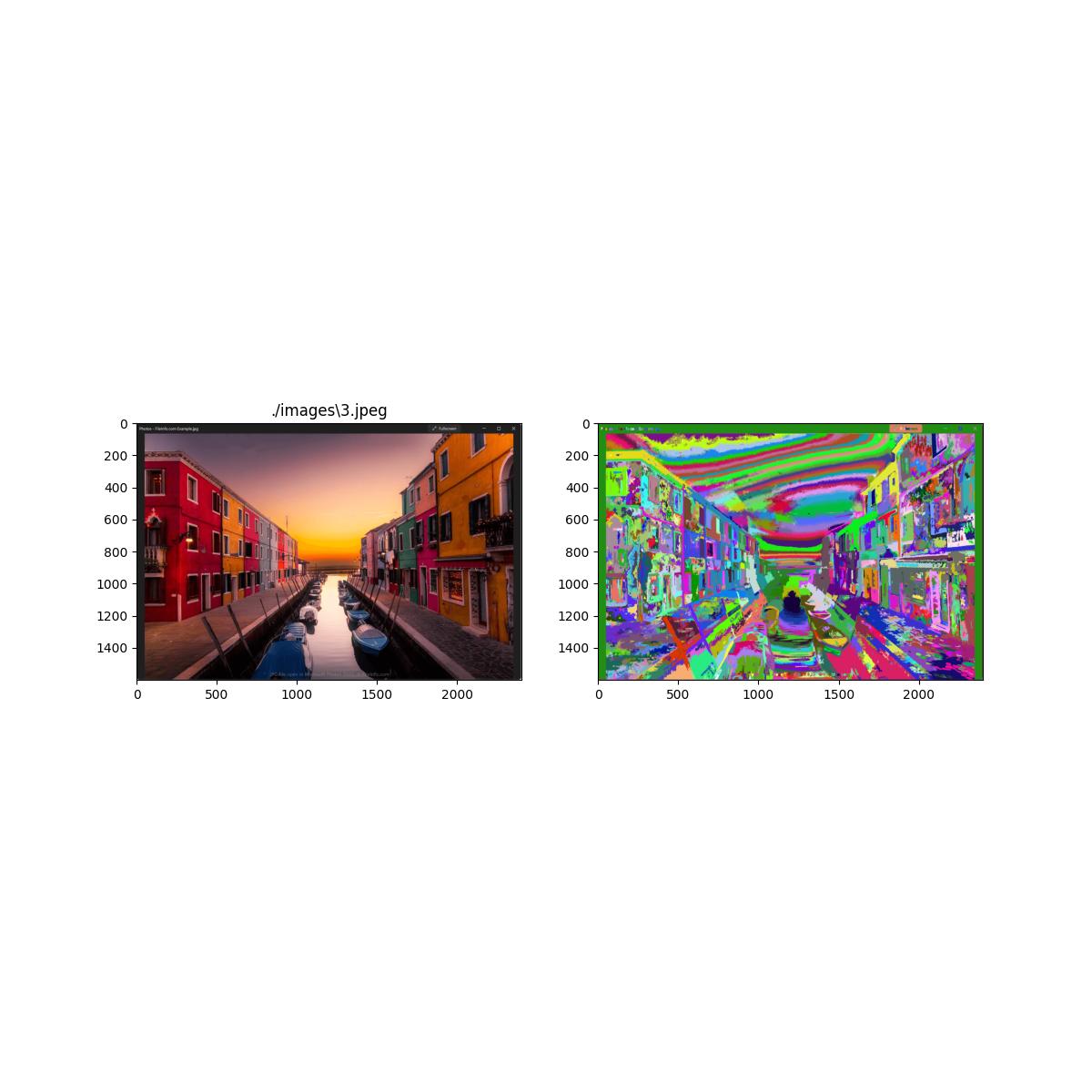
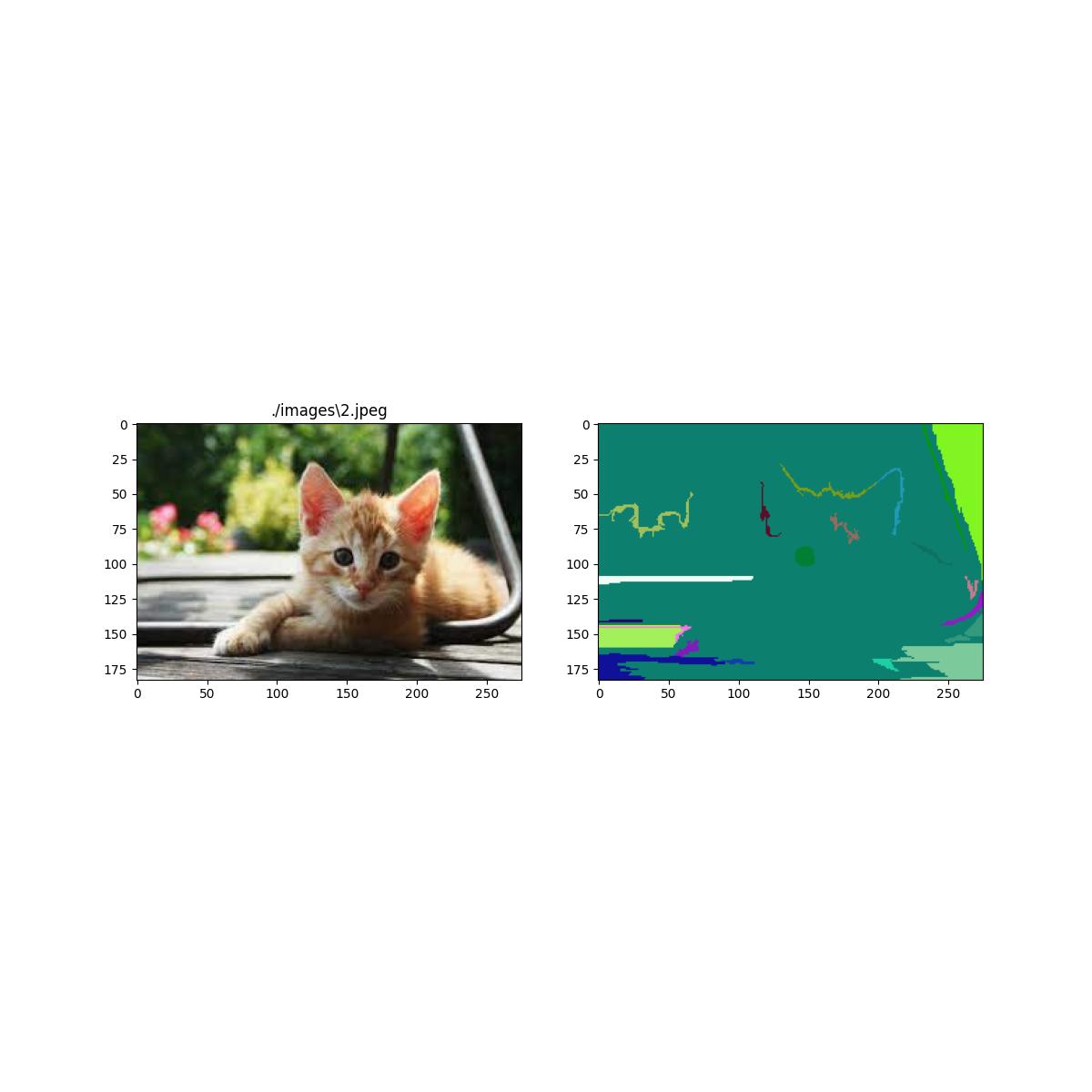
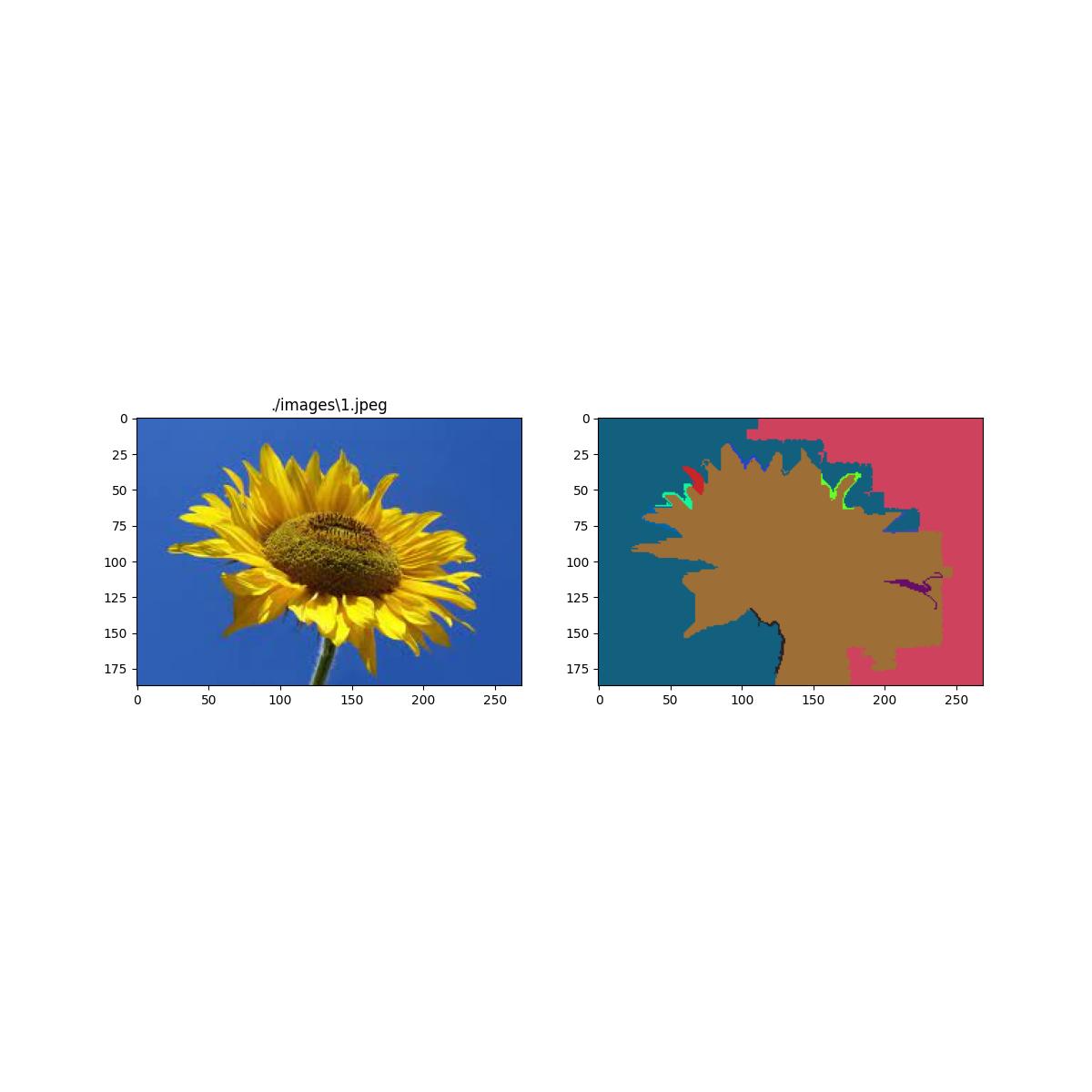
The implementation of the algorithm can be found in the Jupyter Notebook *main.ipynb*.

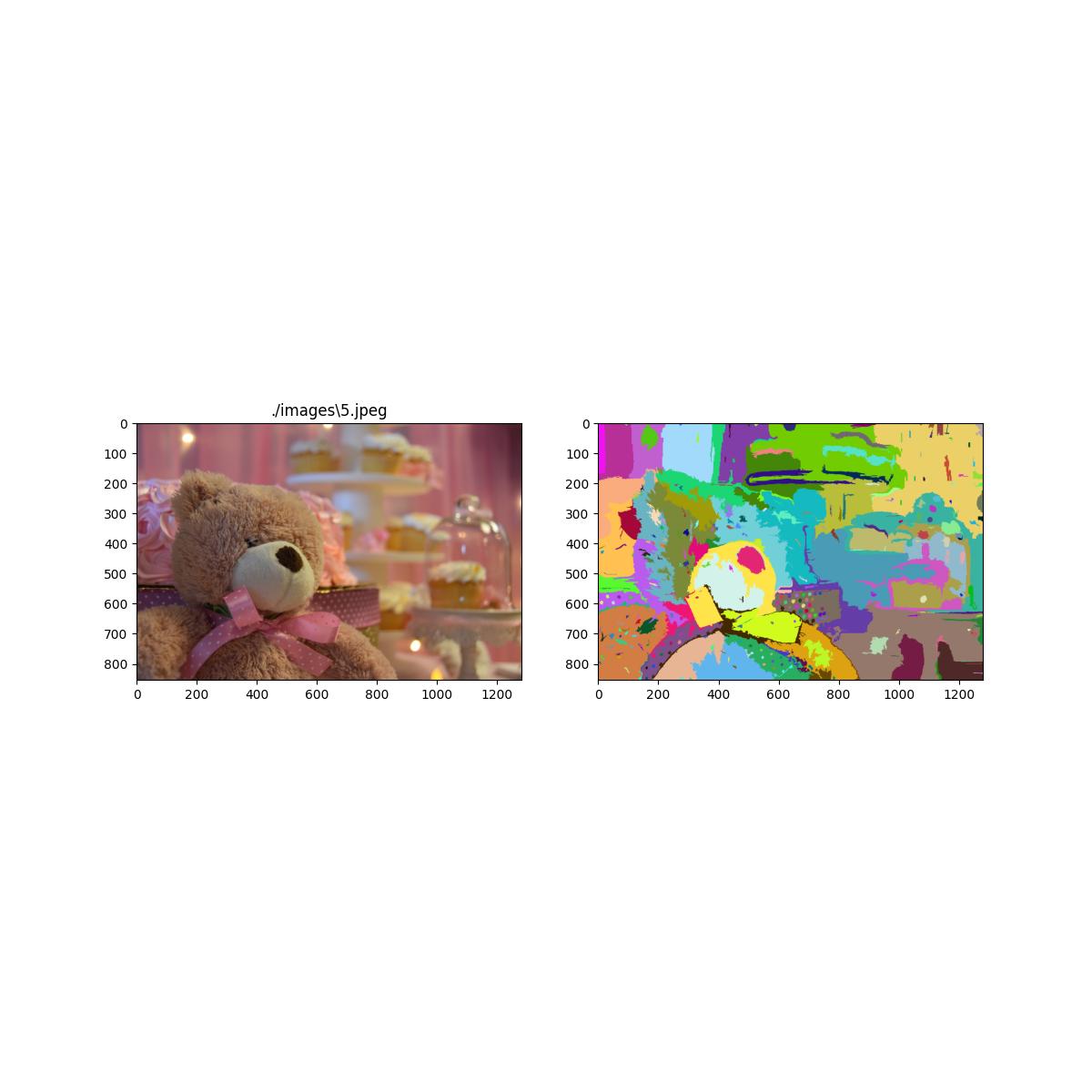
The code has been well-documented with appropriate explanations given.

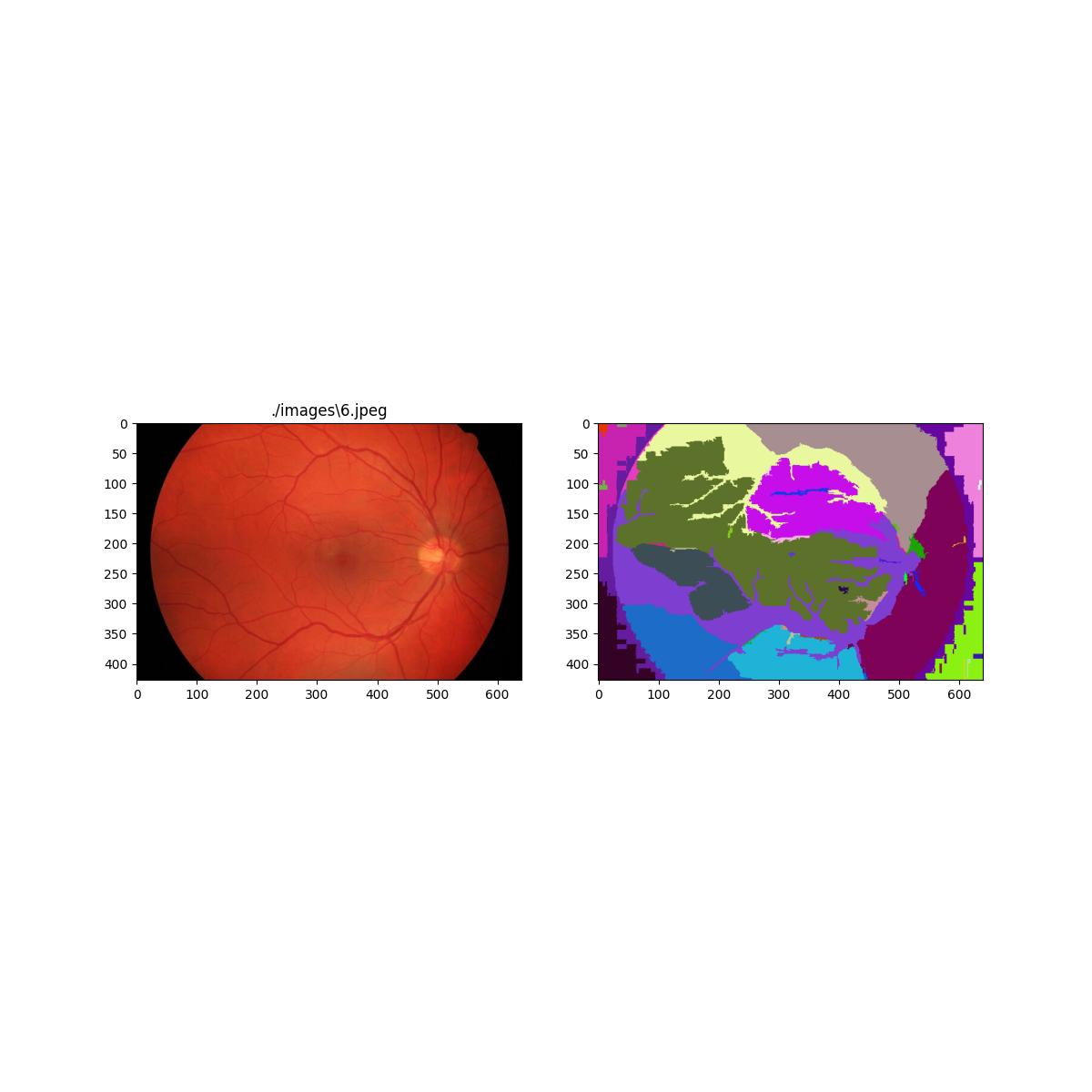
The results of running the algorithm on images are also present in the notebook.

# Results

The algorithm was implemented in Python, and it was run on 6 images of the .jpeg format.

Results are attached below:  






# Conclusion

The ground-breaking work by Felzenszwalb et al. has potential to be of great use in the field of image segmentation, and that image segmentation based on pairwise region comparison gives promising results while still being efficient with respect to time complexity of the algorithm.

The segmentation method, coupled with the use of a nearest neighbor graph, can capture very high level properties of images, while preserving perceptually important region boundaries.

The algorithm has proven to be efficient, running in time for m graph edges. It has also shown that though it makes greedy decisions, it still segments perceptually important regions which reflect global aspects of the image.

This research paper furthered our understanding of how graph-based algorithms work, the normalized cuts criterion, and about image segmentation in general.

# References

* [Efficient Graph-Based Image Segmentation](https://cs.brown.edu/people/pfelzens/papers/seg-ijcv.pdf); Pedro F. Felzenszwalb, Daniel P. Huttenlocher.
* [This](https://github.com/soumik12345/felzenszwalb_segmentation) implementation of the aforementioned paper.
* [Normalized Cuts and Image Segmentation](https://people.eecs.berkeley.edu/~malik/papers/SM-ncut.pdf); Jianbo Shi, Jitendra Malik.