Foreground Extraction towards Object Recognition

3D Computer Vision | EE 645 | Final Project Report

Submitted by: Vinu Sankar S., 16110143, *vinu.sankar@iitgn.ac.in*

Supervised by: Prof. Shanmuganathan Raman

Indian Institute of Technology Gandhinagar - 382355

Abstract

Object recognition is one of the key function performed in computer vision tasks. While most of the object recognition tasks do not employ segmentation algorithms, a common assumption is that good segmentation improves the recognition results. The algorithm proposed in [1] runs for few seconds for a single query and can be efficiently implemented to aid recognition tasks unlike other segmentation algorithms with few training images. The algorithm is aided with other popular methods like superpixel-ing [2], visual word vocabulary [3], global image descriptor [4], and graph-cut based energy minimization [5]. This document reports the results from the implementations of algorithms from papers [1], [2], [3], and [4].

1. Introduction

The work by Rosenfeld et al. [1] introduces an algorithm that can be used for foreground extraction of images, linked to the goal of object recognition. Foreground extraction could help in cropping out the subject for recognition in the image, to make the search space smaller, making the task computationally efficient. The algorithm learns appearance and geometric prior of the object from train annotated training images. The query image is grouped into superpixel segments using the SLIC [2] algorithm, without losing the information about edges. The foreground is extracted from the image by converting the task into a graph partition problem using a graph-cut based energy minimization method [5].

2. Methodology

Geometric prior is calculated from the train images for the query image using GIST representation [4], which is a global image descriptor. GIST features are used to select similar images from the training set of the query image based on color and geometry. Geometric prior gives a pixel-level probability of a pixel in query image being a foreground. It tells us wherein the image can most probably the object be present. Appearance prior is calculated using a bag of words (BOW) representation [3] of the images. Appearance prior gives us information about the color, texture, and shape features of the object in the query image. Using both geometric and appearance prior, we get an energy map, and combining the superpixel segmentation information with the prior information, gives us an approximate estimation of the foreground. Now we can either solve the graph problem or minimize the energy by graph-cut optimization [5] to get a more accurate foreground extraction as stated in [1].

2.1. GIST: A Global Descriptor

GIST is a global image descriptor that is used in the calculation of geometric prior. Implementation gives a 960 length vector for each image that is used to find similar images to the test image from the dataset. The top similar images are then used to learn the geometric prior. Fig 1. shows the similar top 10 images for a test image from the VOC 12 dataset [8]. The first image is the test image in Fig 1. Python codes are provided on Github [9].

2.2. SLIC: Superpixel-ing

SLIC [2] is a fast algorithm for superpixel-ing images. Superpixels make processing images easier as it groups similar neighboring pixels. The SLIC algorithm performs the task quickly without losing the edge data. Fig 2. shows the results obtained from the algorithm. Python codes are provided on Github [10].

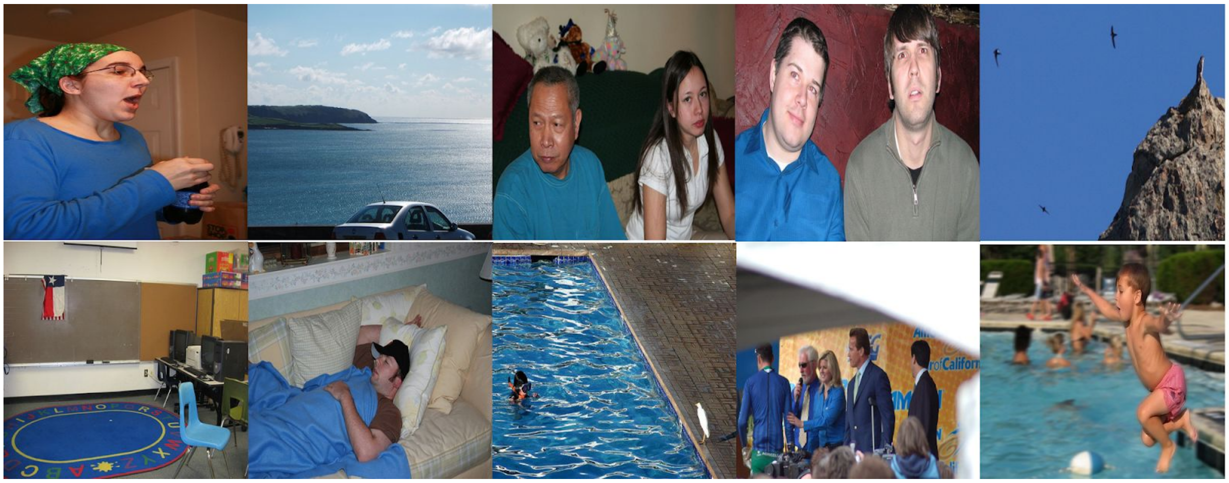

Fig 1: Geometrically similar images using GIST. The top left image is the query image.

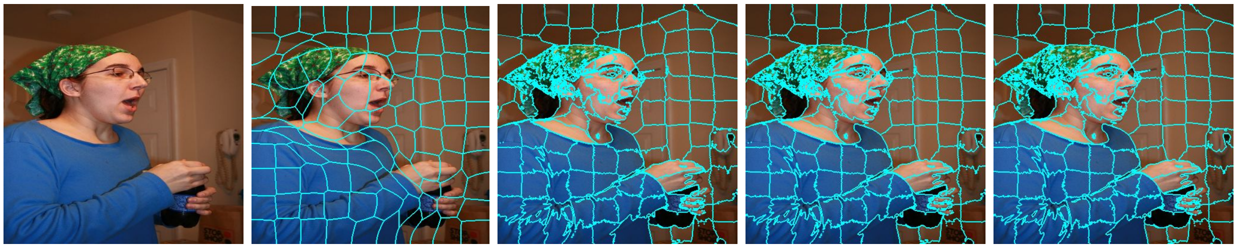

Fig 2: From left to right: Input image; SLIC output from the inbuilt module; Python implementation output after 6th; 8th; and 10th iteration.

2.3. Visual Word Vocabulary

The images are represented using Bag of Words (BoW) [3] histogram vectors to find out images from the dataset that are similar appearance wise to the query image. The top similar images are then used to learn the appearance prior. Fig 6. shows the output for a query image. Python codes are provided on Github [11].

3. Mathematical Results

The mathematical results used from [1] is stated here:

$P_f^G (x, y) = 1/K_G \Sigma_I [L_I (x, y) \varepsilon \text{ foreground}]$

is the geometric prior for pixel (x,y) obtained using GIST. K_G is the number of training images selected, I is each selected image, [.] is an in indicator function to indicate a true value by an output of 1, else a 0.

$P_f^{\sim}(w_p) = (\sum_{q \varepsilon \text{ foreground}}[w_q = w_p])/\{(\sum_{q \varepsilon \text{ foreground}} [w_q = w_p]) + (\sum_{q \varepsilon \text{ background}}[w_q = w_p])\}$

is the appearance prior calculated using the BOW representation, where w_v denotes the visual word assigned to the descriptor of pixel v.

$P_f(v) = 1/|S_v| \Sigma_{p \varepsilon Sv} P_f^{\sim}(w_p) (P_f^G(x, y))^{\gamma}$

calculates the energy map, where S_v denotes the

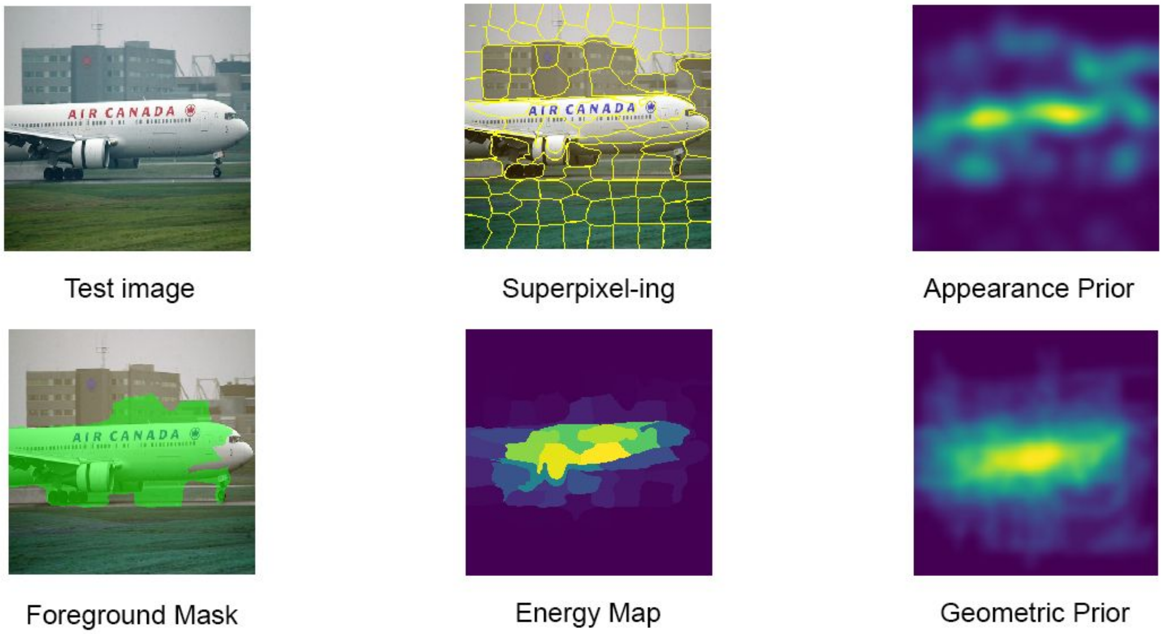

Fig 3: Flow of the algorithm. From top left, clockwise: Test image; Superpixel with SLIC; Learn appearance prior; Learn geometric prior; Energy map produces using priors and superpixel segments; Foreground extracted using energy map.

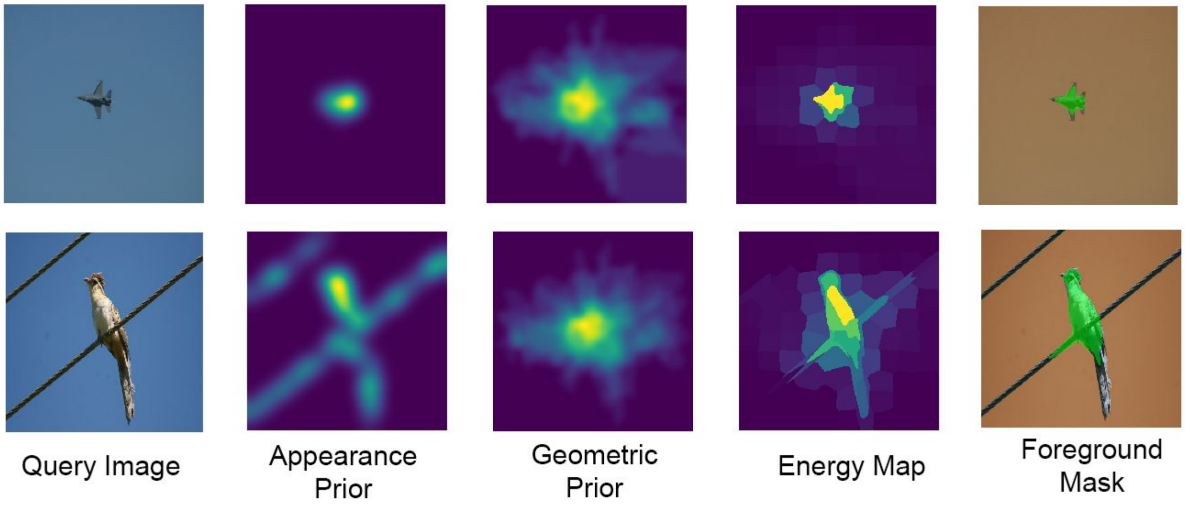

Fig 4: Results obtained from the implementation for two different query images from VOC 12 dataset.

superpixel v to which each pixel $p = (x, y)$ belong.
γ is set to be 0.5.

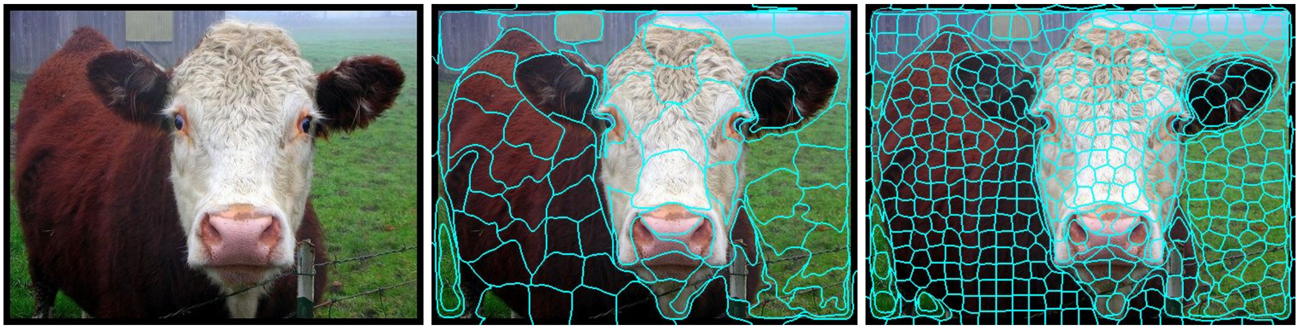

Fig 5: SLIC output from (left to right) input image; 100 superpixel clusters; 500 superpixel clusters.

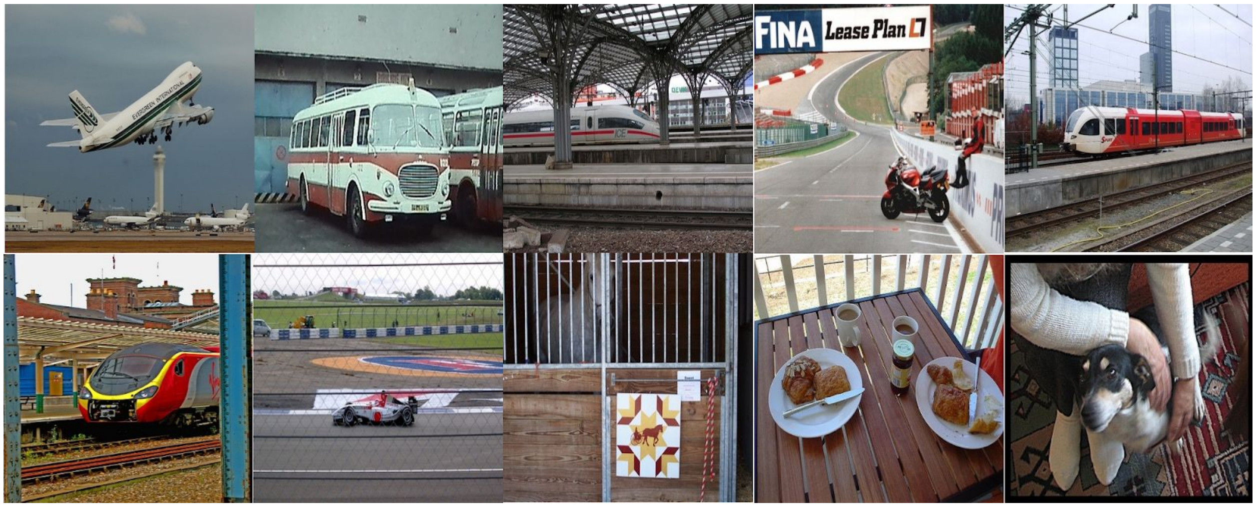

Fig 6: Appearance-wise similar images using BoW. The top left image is the query image.

4. Evaluations and Results

This report evaluates the Python implementation of foreground mask extraction [1], SLIC algorithm [2], BoW visual dictionary [3], and GIST image descriptor [4].

GIST descriptors are used to get a low-dimensional 960 length vector that could be used to find similar images (similar layout) in very large datasets. Fig 1. Shows the implementation result of GIST to find out 10 similar images from VOC 2012 dataset [7]. For reducing computation power for each query, the GIST descriptors for all the images in dataset are pre-computed and stored. Similar images to a query image is found by computing the nearest neighbors using the descriptor vector. GIST sums up response to different oriented Gabor filters over mesh grids, at multiple scales. The idea is very similar to SIFT [7]. For a dataset with over 2000 images when pre-computation takes about 5 minutes, each test query takes about 10 seconds to get top 20 similar images.

GIST enables us get images with similar layout but not with similar content. For this purpose we employ BoW [3] to get images of similar content from a large dataset. It uses dense SIFT [8] to extract features and then uses it to cluster to create visual vocabulary. SURF can be used to speed up the implementation. Pre-computations can be performed to save time. A pre-computation of 2 minutes will get us top 20 similar images in 5 seconds from a dataset of over 2000 images. Fig 6. shows the output from the implementation grouping

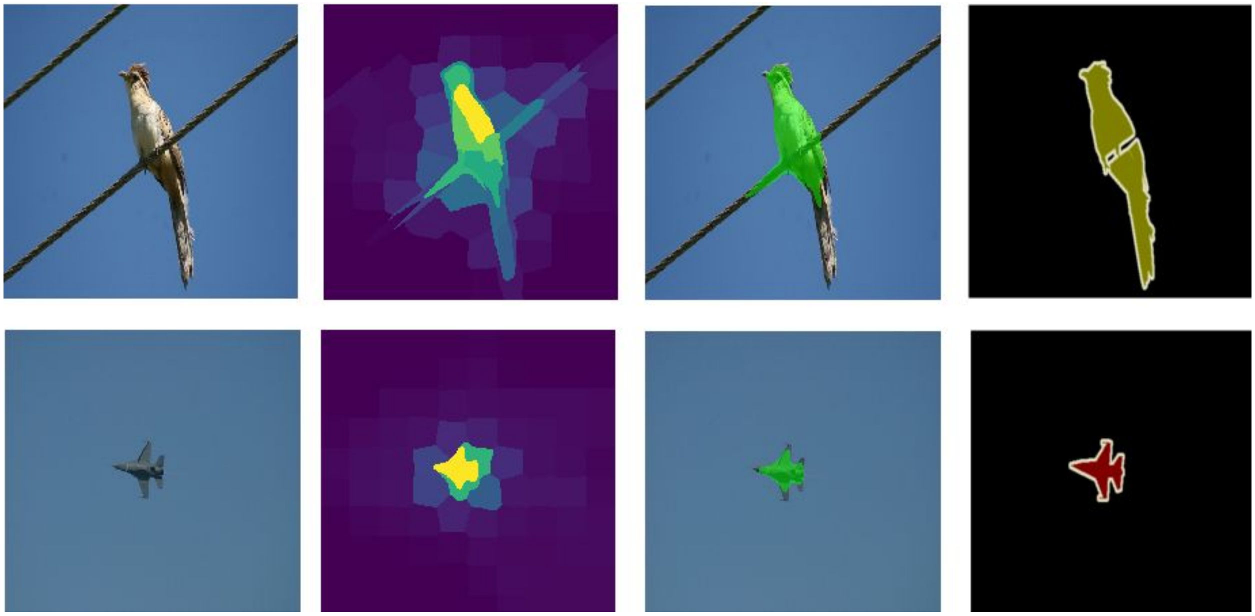

Fig 7: Output from Python implementation (left to right): Input image; Energy map; Output foreground mask; Ground truth segmentation

similar images using BoW to learn appearance prior.

SLIC algorithm [2] employs simple linear iterative clustering. It is faster and memory efficient. The result improves as number of iteration increases. The inbuilt SLIC module of Python does the task in under 5 seconds, whereas the Python implementation takes over 30 seconds per iteration. This is due to the C wrapper Python module uses that makes it faster. The results of implementation can be seen in Fig 5.

Foreground extraction towards object recognition is the main objective of the implementation. GIST, SLIC, and BoW were supplements for the same. Results can be seen in Fig. 7.

5. Conclusions and Discussions

The output from the Python implementation is extracting main object from the scene without loss of much data, which could suffice the objective of object recognition. Since geometric and appearance priors are learnt well, it can also be helpful in object recognition. The paper by Rosenfeld et al. [1] is implemented except for the part where they aided a graph-cut based energy optimization method to extract foreground as in [5]. Instead, in the implementation, a threshold value is input which classifies superpixels to be foreground or background based on their appearance and geometric prior. [5] is not used in the Python implementation as it uses a heavy MATLAB module that is unavailable on Python, and this explains the good results that [1] obtain. But the work can be done with time. With all pre-computations done, the implementation takes about 45 seconds to get the foreground extracted on Python, whereas in [1] they claim it takes only 3 seconds.

The current day deep learning techniques can do the task with much better results [6]. So the implementation successfully implements SLIC, GIST, and BoW methods to suffice the task of foreground extraction towards object recognition. Python implementation codes can be seen on Github [12].

Note
A major observation is that those images with blue color layouts (because of sky), like the one with birds or planes are abundantly found in the dataset and helps improving the priors. This gives better results for plane or bird images in sky. So a better balanced dataset could improve results. After performing [5] on energy map, we can compare results with states-of-the-art by using metrics like IoU (Intersection of Unions).

Acknowledgement
I would like to thank Prof. Shanmuganathan Raman for giving me the opportunity to do this project for his 3D computer vision course.

6. References

[1] A. Rosenfeld and D. Weinshall, "Extracting foreground masks towards object recognition," 2011 International Conference on Computer Vision, Barcelona, 2011.

[2] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk. SLIC Superpixels. Technical report, EPFL, 2010.

[3] N. M. Ali, S. W. Jun, M. S. Karis, M. M. Ghazaly and M. S. M. Aras, "Object classification and recognition using Bag-of-Words (BoW) model," *2016 IEEE 12th International Colloquium on Signal Processing & Its Applications (CSPA)*, Malacca City, 2016, pp. 216-220.

[4] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. International Journal of Computer Vision, 42(3):145–175, 2001.

[5] Y. Boykov, O. Veksler, and R. Zabih. Efficient approximate energy minimization via graph cuts. IEEE transactions on Pattern Analysis and Machine Intelligence, 20(12):1222–1239, November 2001.

[6] K. Wu and Y. Yu, "Automatic object extraction from images using deep neural networks and the level-set method," in *IET Image Processing*, vol. 12, no. 7, pp. 1131-1141, 7 2018.

[7] David G. Lowe. 2004. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vision* 60, 2 (November 2004), 91-110.

[8] http://host.robots.ox.ac.uk/pascal/VOC/voc2012/

[9] Github/vinusankars/Gist

[10] Github/vinusankars/Slic

[11] Github/vinusankars/BoW

[12] Github/vinusankars/fgextraction

[13] Github/slic