Pb-Lite Contour Detection

Saket Seshadri Gudimetla Hanumath
A. James Clark School of Engineering
University of Maryland
College Park, Maryland
Email: saketsgh@terpmail.umd.edu
Used 3 Late Days

***Abstract*—The objective of the project discussed in this report is to present a contemporary solution to one of the most fundamental problems of computer vision i.e. Contour or Edge detection. The algorithm implemented in this project is called the PbLite algorithm. It is considered better than the classical edge detection algorithms like Canny or Sobel because it is able to suppress false positives in the textured regions. The PbLite algorithm is discussed in various stages and its output is compared with the Canny and Sobel baselines in the end.**

I. Introduction

Edge detection is a fundamentally important problem in computer vision since it helps in analysing patterns in an image. Classical edge detection methods work by finding discontinuities in the intensity across an image; which are called edges. For example, the Sobel operator; a differentiation operator which when convolved on an image gives the gradient intensity function of that image. Other examples of popular algorithms include the Canny edge detector that works on the principle of non-maximum suppression.
Pb or probability of boundary edge detection algorithm has recently been developed and it outperforms the classical edge detection methods by accounting for the Texture, brightness and color discontinuities.
In this project, a simplified version of the Pb (Probability of Boundary) edge detection algorithm called the Pb-Lite algorithm has been implemented.

A. Phase 1: Shake my Boundary(Contour Detection)

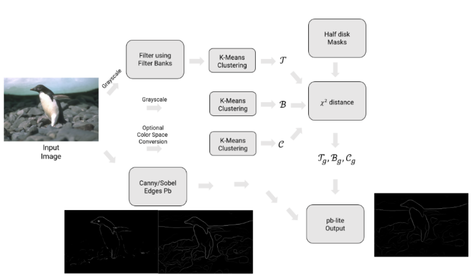

Fig. 1. Algorithm Overview

An overview of the Pb lite algorithm is shown in figure1. The first stage involves generation of three filter banks namely- Oriented DOG, Leung-Malik and Gabor respectively. These filter banks help in generating the texture details of an image. The filter banks are then convolved with the given set of images, thus encoding texture properties. This is achieved by clustering all the responses at all the pixels into textons using KMeans. This is followed by the generation of Brightness and Color maps by clustering brightness and color values respectively.
Next we compute differences of values across different shapes and sizes. This can be achieved very efficiently by the use of Half-disc masks. This step generates three types of gradients; Texture, Brightness and Color respectively. These encode how much the texture, brightness and color distributions change at a pixel.
The final step is to combine information from the features with a baseline method (based on Sobel or Canny edge detection or an average of both) using a simple equation.
The various stages explained above are discussed in detail with their respective results in the following sections.

B. Filter Banks

The first step in the Pb-Lite Contour Detection is to create a set of filters . A filter bank is defined as a collection of filters. There are three types of filter banks used for this process:

1) Oriented DoG Filters: :These filters can be generated by convolving a Gaussian kernel with a Sobel filter. Oriented DoG filters are generated with 'o' orientations and 's' scales where scale is given by the standard deviation σ of the Gaussian kernel to give o*s total filters. A set of 32 filter banks with 2 scales and 16 orientations were created and can be seen below in figure 2.

2) Leung-Malik Filters: Leung-Malik Filters: LM filter bank consists of first and second order derivatives of Gaussians at 6 orientations and 3 scales making a total of 36(18 each); 8 Laplacian of Gaussian (LOG) filters; and 4 Gaussians. There are two types of LM filters- LM Small(LMS) filter bank which occur at the basic scales of σ = [1, $\sqrt{2}$, 2, 2$\sqrt{2}$]. The first and second derivative filters occur at the first three scales with an elongation factor of 3(σ_x = σ, σ_y = 3σ_x). The Gaussians occur at the four basic scales while the 8 LOG filters occur at σ and 3σ. The LM Large (LML) filters occur at the basic scales σ = [$\sqrt{2}$, 2, 2$\sqrt{2}$, 4]. The LM filter bank generated is shown in figure 3 and 4.

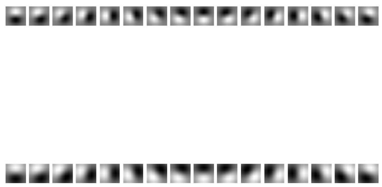

Fig. 2. Oriented Derivative of Gaussians Filter Bank

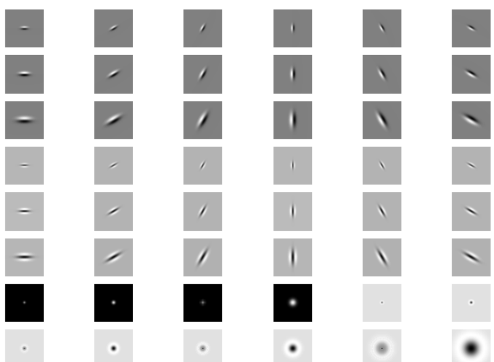

Fig. 3. LML filter bank

3) Gabor Filters: These filters are derived from the ones present in the human visual system, the Gabor filter bank is a Gaussian kernel function modulated by a sinusoidal plane wave. The Gabor filter bank generated is shown in figure 5.

C. Texton Map

To generate the Texton Map T, the input image is filtered with each filter of the filter bank. This gives N filter responses at each pixel of the image. An N dimensional vector is obtained on which a K-Means clustering is implemented with the number of clusters K = 64. The original images and their corresponding Texton maps are shown in figure 6, 7 and 9, 10 respectively.

D. Brightness Map

The Brightness Map B is generated by performing a KMeans clustering on the grayscale version of the color image with the cluster size K = 16. The brightness map of the given images are shown in figure 11 and 12.

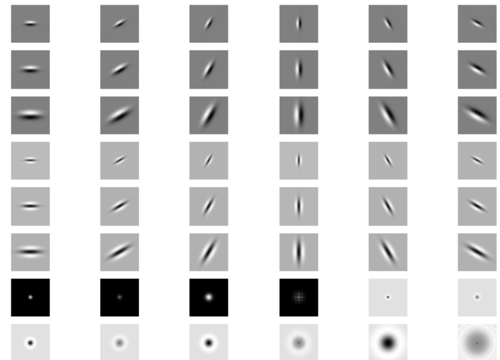

Fig. 4. LMS filter bank

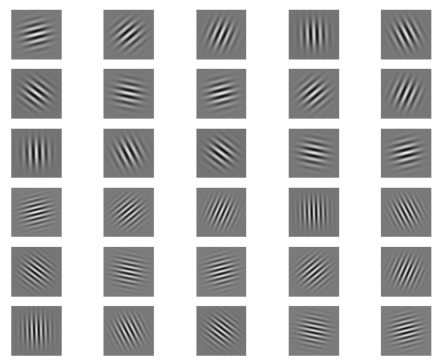

Fig. 5. Gabor filter bank

E. Color Map

Color map is generated by performing a K-Means clustering on the color image with the cluster size K = 16. It captures changes in chrominance in a given image. The Color maps generated are shown in Figures 13 and 14.

F. Texture, Brightness and Color Gradients - T_g, B_g, C_g

The three gradients are computed very efficiently by the use of Half-disc masks. The half-discs masks are binary images of half-discs that occur in pairs. This method of filtering is very fast compared to looping over each neighboring pixel and aggregating counts for histograms. Half disks masks help in computing differences of values across different shapes and sizes. These are pairs of binary images of half discs at different scales.
The gradients encode how much change is there in texture, brightness and color distributions accross the pixels of the image. The gradients are computed by convolving the various maps generated with these pairs of opposing filters(left and right) and then we look for changes between the two. The

Fig. 6. original image 1

Fig. 7. original image 7

gradient is large if the distributions have high dissimilarity and small otherwise. Since there are so many orientations and scales it can encode how quickly the three properties change. Figure 8 shows the various Half Disc Masks. It is an

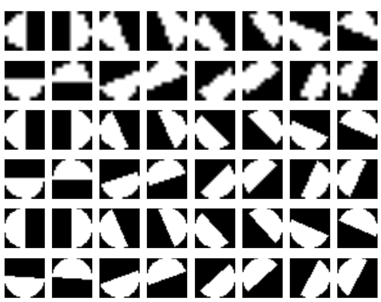

Fig. 8. Half Disc Masks

important step as it allows us to compute the χ^2 distance. χ^2 is the sum of squared difference between histogram elements.

G. PbLite output

Finally the gradients and the canny and sobel baselines are combined using the formula shown below to achieve the PbLite output where the dot in between is used to represent the Hadamard operator.

$$PbEdges = \frac{(T_g + B_g + C_g)}{3}.(w_1 * cannyPb + w_2 * sobelPb)$$

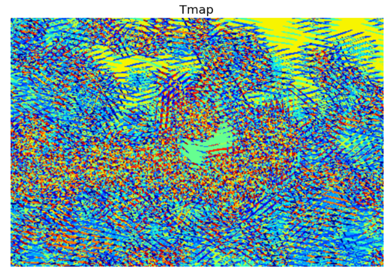

Fig. 9. Texton Map 1

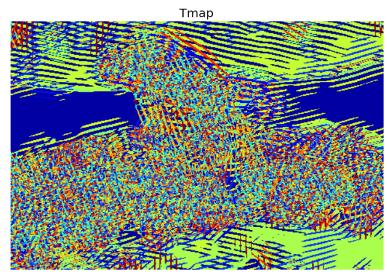

Fig. 10. Texton Map 2

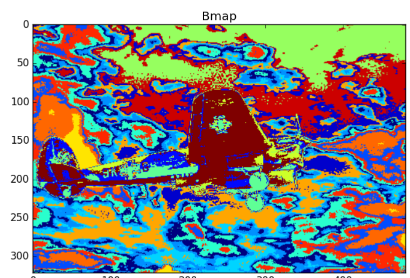

Fig. 11. Brightness Map 1

The magnitude of the features represents the strength of boundaries, hence, a simple mean of the feature vector at location i should be somewhat proportional to pb. The PbLite outputs can be seen in figures 23 and 24 respectively.

II. Conclusion

Both Sobel and Canny model edges as sharp discontinuities in the brightness channel upon which Canny adds to it the non maximum suppression and hysterisis thresholding. steps. The PbLite algorithm produces better results compared to the Canny and Sobel baselines since it takes into the account the texture, brightness and color information that the previous two are oblivious for.

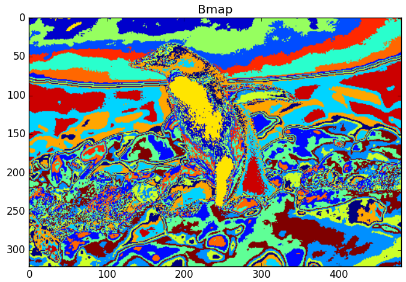

Fig. 12. Brightness Map 7

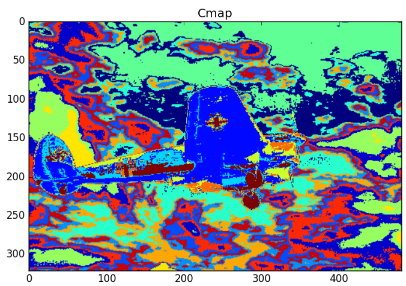

Fig. 13. Color Map 1

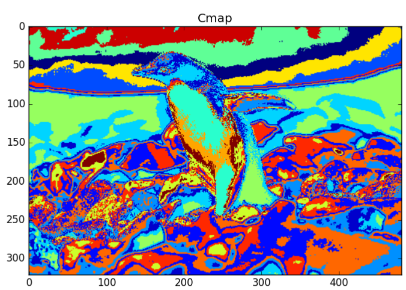

Fig. 14. Color Map 7

Use of different filter banks at various orientations and scales allows the algorithm to account for finer details present in images and suppress false positives.
The PbLite output can be further fine tuned to get even better results but this is a tedious process as the number of parameters that need to be tweaked are a lot.

REFERENCES

[1] https://cmsc733.github.io/2019/hw/hw0/

[2] P. Arbeláez, M. Maire, C. Fowlkes and J. Malik, "Contour Detection and Hierarchical Image Segmentation," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 5, pp. 898-916, May 2011. doi: 10.1109/TPAMI.2010.161

[3] https://en.wikipedia.org/wiki/Gabor filter

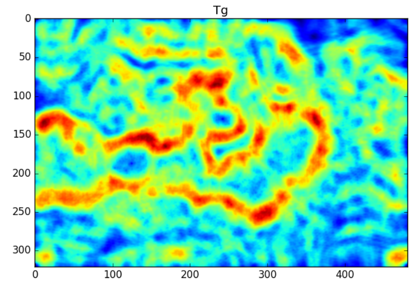

Fig. 15. Texton Gradient 1

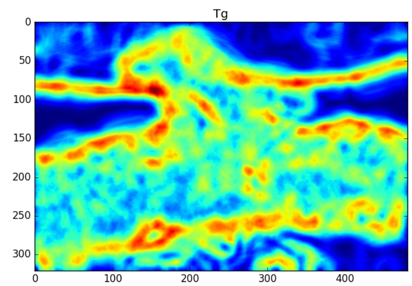

Fig. 16. Texton Gradient 7

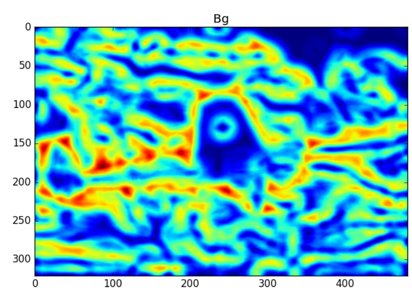

Fig. 17. Brightness Gradient 1

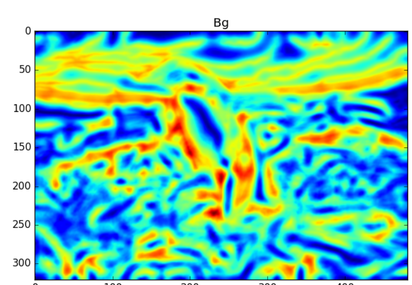

Fig. 18. Brightness Gradient 7

Fig. 19. Canny Baseline 1

Fig. 20. Canny Baseline 7

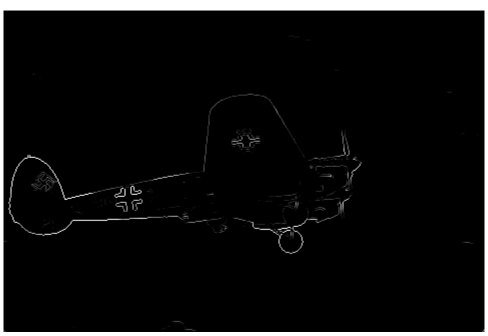

Fig. 21. Sobel Baseline 1

Fig. 22. Sobel Baseline 7

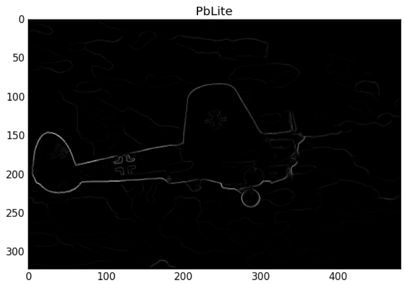

Fig. 23. PbLite output 1

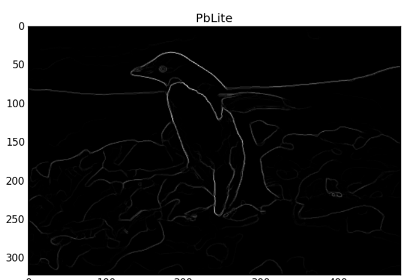

Fig. 24. PbLite output 7