Homework 0: Alohomora!

Nitin Suresh
School of Electrical and
Computer Engineering
University of Maryland - College Park
UID: 113638855

(USING 6 LATE DAYS)

I. Phase 1

The main objective of this project was to develop an algorithm for boundary detection from a single image, by using multiple filters to identify gradients in texture, brightness and color in the image. The information obtained from these gradients was then combined with the well-known Canny and Sobel baselines to obtain the final output.

The first step in the Pb-lite algorithm is the generation of 3 sets of filters which are used to create the texton map. The first of these is the oriented Derivative of Gaussian (DoG) filterbank, an example of which is displayed below. This filterbank was generated using 2 scales (or sigmas) and 8 orientations (shown in Fig. 1).

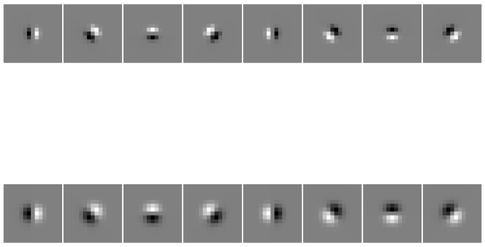


Fig. 1. DoG filter bank

The second set of filter banks generated were the Leung-Malik Small and Large filters, which vary with respect to the scales employed for the filters. These filter banks consist of first and second order DoG filters, LoG filters and Gaussian filters. These are displayed in Fig. 2 and Fig. 3.

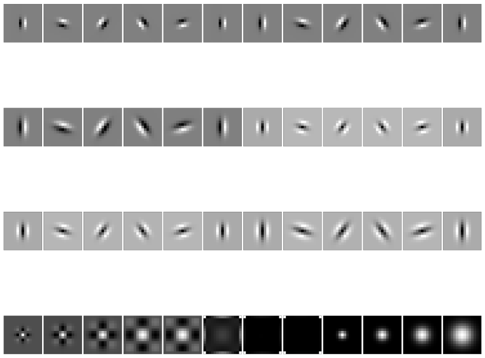


Fig. 2. Leung-Malik (small) filter bank

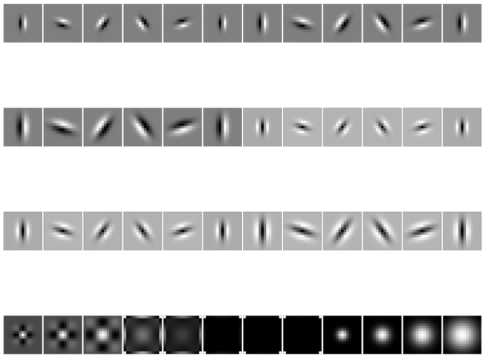


Fig. 3. Leung-Malik (large) filter bank

Gabor filters are also generated and used, which are gaussian kernels modulated by sinusoidal plane waves (Fig. 4).

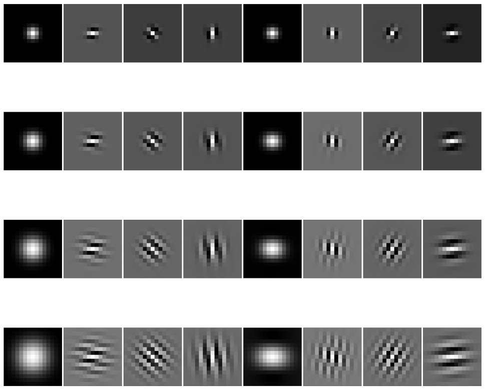


Fig. 4. Gabor filter bank

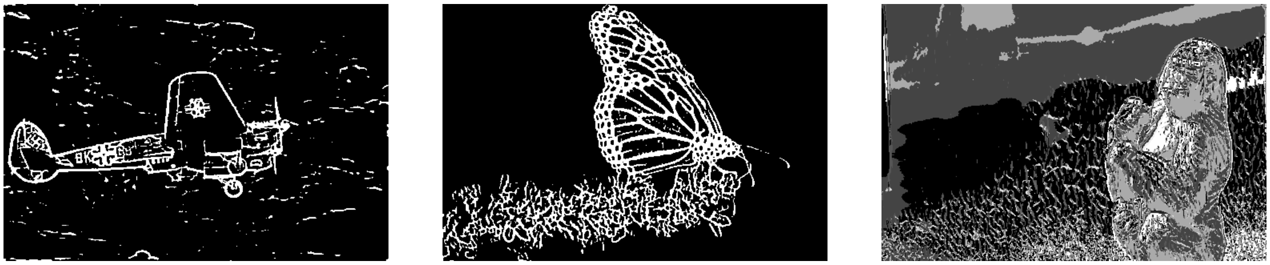
In this manner, a total of 144 filters were generated, including those at several scales and orientations. These filters were used to create feature vectors at each pixel of the input image, which was then clustered using KMeans. The number of clusters was varied to see which gave a good tradeoff between performance and processing time, and a finally sixteen clusters were used for generation of all three relevant maps. Example texton maps are shown in Fig. 5.

For generation of the brightness and color maps (Fig. 6 and Fig. 7), the input image was converted from RGB to the CIELAB space. The luminance channel L was then used for brightness map generation and the a and b channels were used for color map generation. Clustering was again carried out using the KMeans algorithm. For calculating the gradients from these maps, pairs of half-disc masks are used which are basically just binary images generated at different scales. The chi-squared distance was then measured by using a bin-filtering method where the bins are the clusters which were generated in the previous step. Using this technique, and by filtering with each pair of half-discs, the gradients were generated (Fig. 8, Fig. 9 and Fig. 10). While the texton, brightness and color maps themselves had the visual meaning of quantization, the chi squared distances representing the gradients maps seemed to be a bit more abstract in the visual sense.

The final step was to combine the gradient information with the Sobel and Canny baselines by using weighting factors $w1$ and $w2$ (=0.5). Results are shown in Fig. 11.

II. Conclusion

The Pb-lite approach generates the edge probabilities of pixels in the original image, by considering changes in intensity, color and texture between neighboring groups of pixels. The Sobel or Canny edge detectors are limited with respect to the scales and orientations that they take into account. These baseline algorithms also just consider changes in the image intensity, and do not consider any changes in texture. In the case of the Pb-lite approach, texton, brightness and color gradients are generated and combined for use. In most cases, the Pb-lite output performed better as compared to the baseline algorithms. In the case of some of the highly textured images though, we saw that the Pb-lite algorithm generated a lot of edges which were part of the textures in the image.

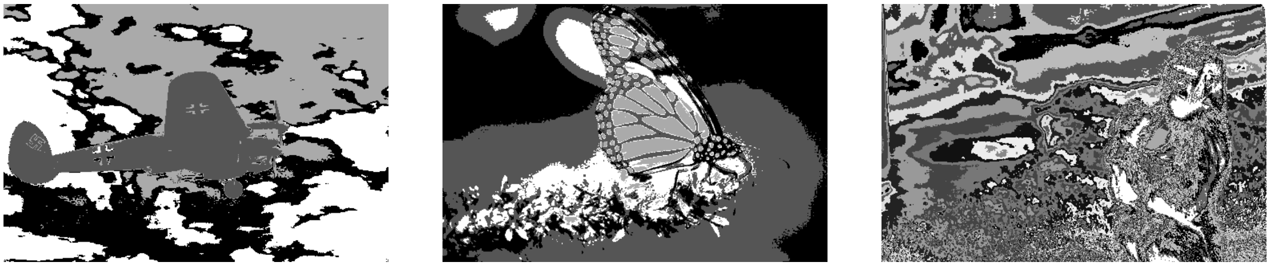


(a)

(b)

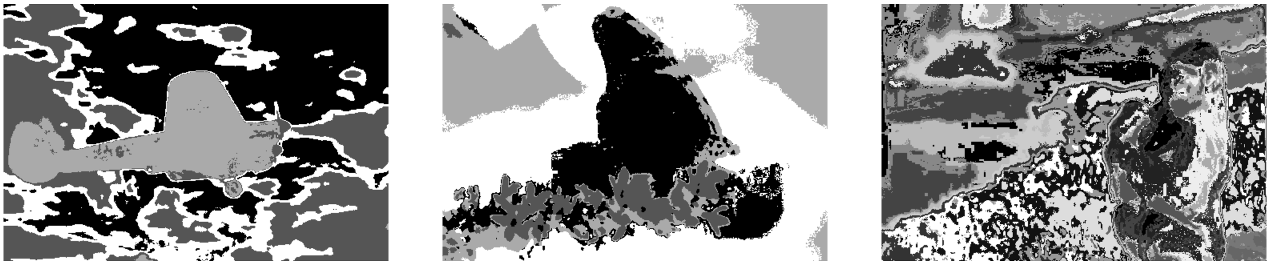
(c)

Fig. 5. Texton Maps



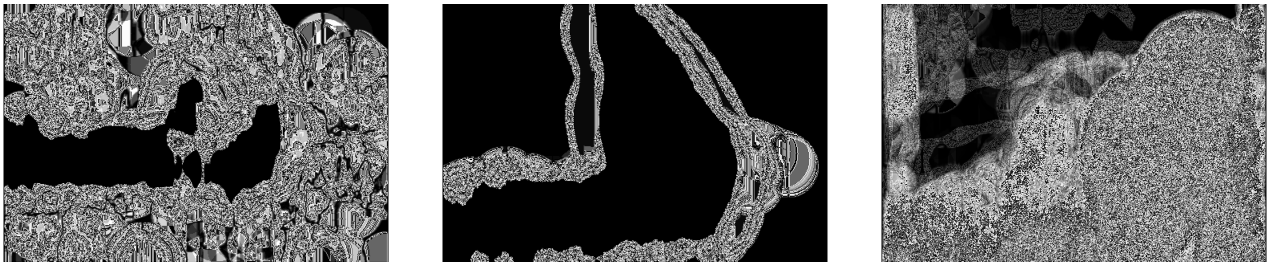
(a) (b) (c)

Fig. 6. Brightness Maps



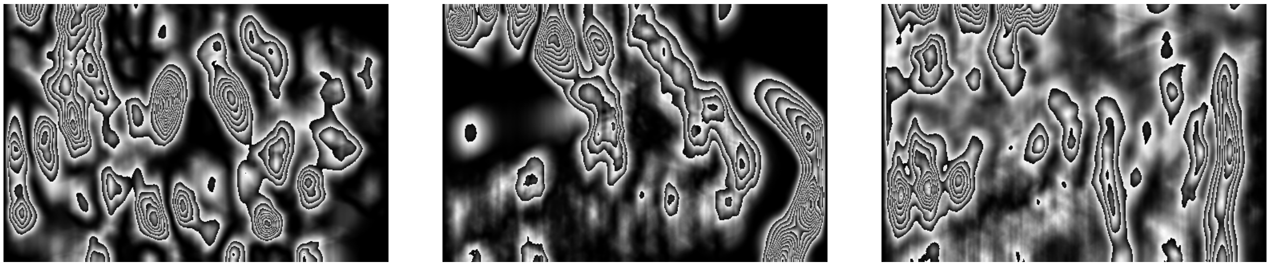
(a) (b) (c)

Fig. 7. Color Maps



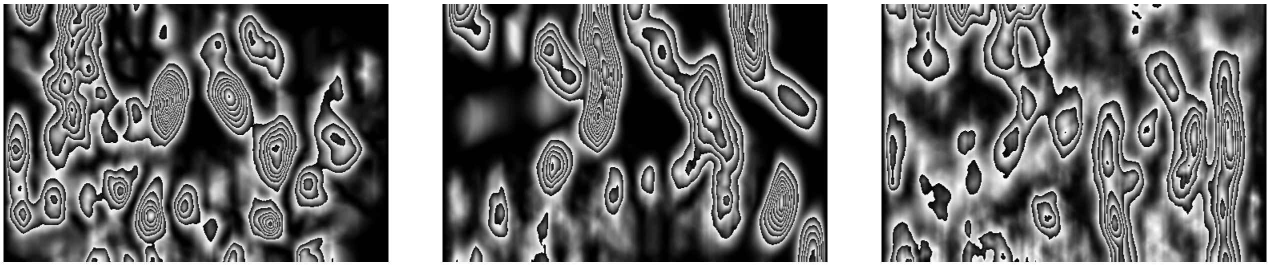
(a) (b) (c)

Fig. 8. Tg Maps



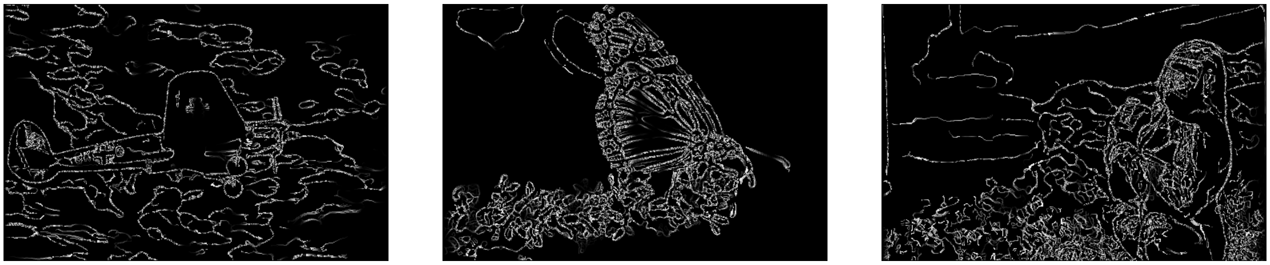
(a) (b) (c)

Fig. 9. Bg Maps



(a) (b) (c)

Fig. 10. Cg Maps



(a) (b) (c)

Fig. 11. PbLite outputs