Homework 0: Alohomora!
Basic Color Segmentation and Boundary Detection

Wenyan Li
Email: wenyanli@umd.edu

***Abstract*—This project mainly focuses on two image processing problems: color segmentation and boundary detection. In the first task, a gaussian filter was generated for smoothing the raw image. Then the HSV color space and morphological operations are used for segmenting colored objects from the background. To further find individual colored objects, k-means clustering method is applied to cluster objects with the same color. In the second part, a boundary detector similar to Ref. [1] is created, which outperforms sobel and canny edge detection by combining brightness , color and texture gradients of the image with baseline detections. Filter banks and half-disc masks are created and chi-square measure is used for gradients computing.**

I. Introduction

In this project, color segmentation and a well-performed boundary detector are fully implemented. Works have been done in the following aspects:

1) Colored pins are counted and segemented from the white background, and pins in the same color are clustered with k-means clustering.
2) First Derivative of Gaussian, Leung-Malik (LM), Schmid (S) , and Maximum Response (MR) filter banks are created for obtaining pixel responses and half-disc masks of multiple scales and orientations are generated for further gradients extraction.
3) Texture, brightness and color features are extracted for every image, and a boundary detector is well formed by combining the gradients information with weighted baseline detections.

II. Color Segmentation

In this part, first, the raw image is smoothed with a Gaussian filter generated according to

$$g(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{1}$$

Since the image provided has a white background which means the background has a high saturation, the image is transformed in to HSV color space, and the saturation layer is extracted to show clear separation between the pins and the background. Then, dilation operations are conducted on the image with structure element carefully choosed so that the dilated pins can be approximately displayed as circles. With proper dilations done, it becomes much easier to find desired circles in the image. Hence, circles that are representing colored pins can be easily found and counted by thresholding the radius, as shown in Fig. 1. The next step is to implement color-based segmentation using k-means clustering. The smoothened image is transformed in to L*a*b space and then the pins are clustered based on color. In this way, individual colored objects can be found. This step can be easily implemented by applying the kmeans function. The result of clustering is shown in Fig. 2. To count the number of pins in each color, again, the segmented images representing each cluster were transformed into HSV color space and dilated. Similarly, after dilation, circles can be found and counted in individual color. The result is shown in Fig. 3

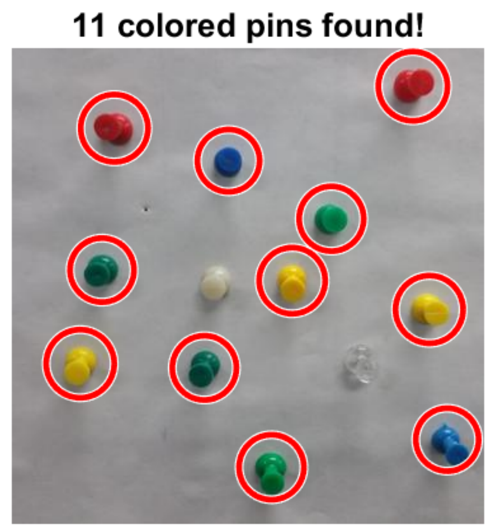


Fig. 1: Detected colored pins

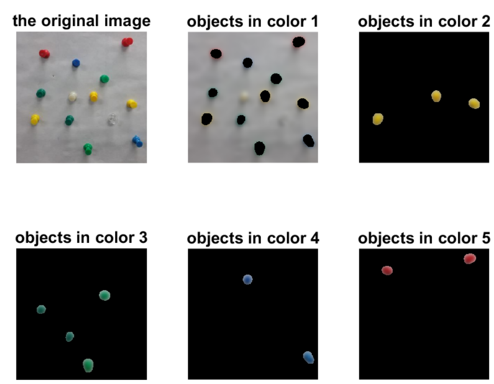


Fig. 2: Detected colored pins

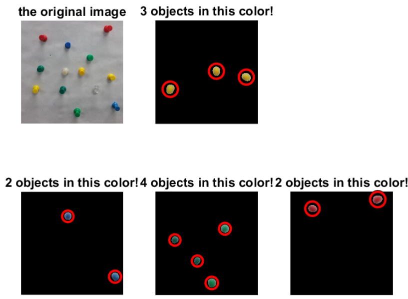


Fig. 3: Individual Colored Objects

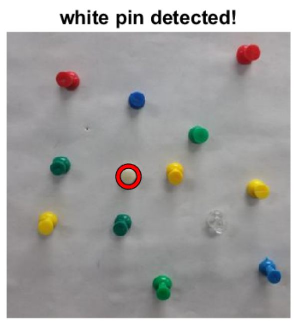


Fig. 4: White pin detection

A. *Extra Credit: Find the white pin*

During clustering, the number of clusters is set as five, and it can be observed in Fig. 2 that in the cluster of white color(which includes the white and transparent pins, together with the white background), pins of other color are covered in black. Inspired by this, I changed the black dots into gray-white to soften the comparison so that the white pin would be detected as a bright white circle in the saturation layer by transforming the image into HSV space. Now, by finding circle in the layer, the white pin can be succefully detected as shown in Fig. 4

III. Boundary Detection

This part aims to implement a simplified pb boundary detector which outperforms the classical detectors such as Sobel and Canny edge detectors by taking texture, brightness, and color gradients into account. In order to fetch these gradients, a filter bank and half-disc masks need to be defined first. The idea is that by clustering filter bank responses on discrete texture elements, a local distribution of textons can be obtained, and the texture gradient can be further computed using the chi-square distance between the half disks. Brightness and color gradients can also be obtained in a similar way after transforming the image into L*a*b color space to get the corresponding values.

A. *Create filter banks*

In this part, four filter banks are created in total with multi scales and orientations and images are used to convolve with them for generating filter responses and further creating texton maps. The filter banks generated in this part include a 16(orientations)*2(scales) First Derivative Gaussian Filter Bank(as shown in Fig.5), a Leung-Malik (LM) Filter Bank, a Schmid (S) Filter Bank, and a Maximum Response (MR) Filter Bank. The derivatives of Gaussian can be computed by convolving a Sobel filter with Gaussian, and Laplacian of Gaussian filters in LM and MR filters can be computed in a similar way (convolve for one more time).

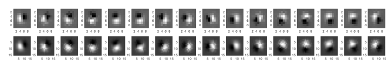


Fig. 5: Gaussian filter bank

The S filters, as shown in Fig.6 can be easily built according to Eq.2 from Ref. [2].

$$F(\gamma,\sigma,\tau) = F_0(\sigma,\tau) + \cos(\frac{\pi\tau\gamma}{\sigma})e^{-\frac{\gamma^2}{2\sigma^2}} \tag{2}$$

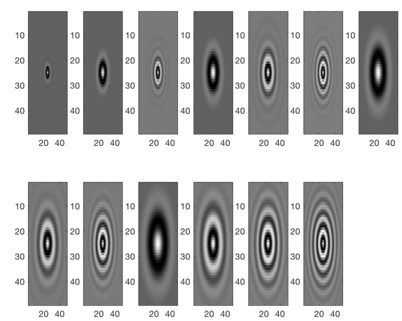


Fig. 6: S filter bank

Once the First Detrivative Gaussian Filter Bank is succefully implemented, LM and MR Filter bank can be accomplished by adding elonged factor and convolution, as shown in Fig.7 and Fig.8.

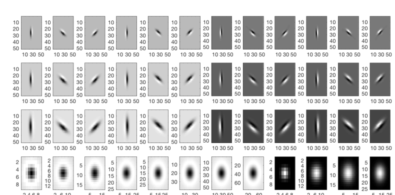


Fig. 7: LM filter bank

B. *Half-disc Masks*

Half-disc masks can be simply generated by creating binary matrix representing the left and right parts of the disks, The

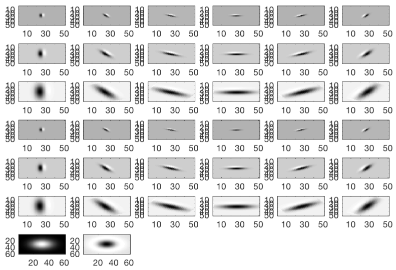


Fig. 8: MR filter bank

discs can be created in multiple orientations with rotation operations and the scales can also be easily motified. The mask set generated with 8 orientations and 3 scales is shown in Fig.9.

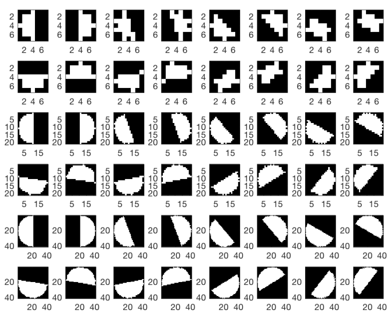


Fig. 9: Half-disc masks

C. Texton Maps

By filtering each pixel in the image and clustering the responses with k-means, alll the pixels in the image can be clustered in to 64 textons and the texton map can be generated. Here, when using kmeans in MATLAB, in my first several tries, it would not converge within the default iteration setting. After changing the iteration parameter for kmeans, it can converge successfully and thus result in a better evaluation result than before. It can be observed that by using different filters, the generated texton maps are slightly different, howevery, by comparing the pb edge evaluation results, using LM, S, or MR filterbanks do not necessarily improve the detector performance as the F score of the precision-recall curve described in Ref. [1] has no significant increase. Hence, first derivative Gaussian filter bank is used for output my pb edge detection results while results of using other three filters can be viewed in the file.

D. Computing gradients

By using the chi-square measure, texton and brightness distributions are compared in half-disc pairs. While using the generated texton map for computing the texture gradient,

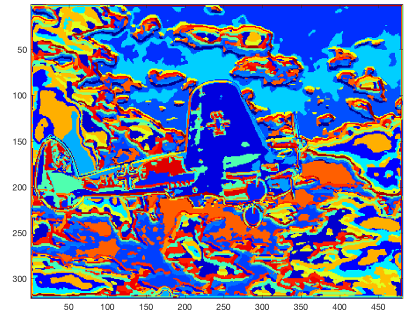


Fig. 10: Example texton map using first derivative Gaussian filter bank

images were transformed in to L*a*b space to obtain brightness (L channel) and color information (a, b channels) for computing brightness and color gradients.

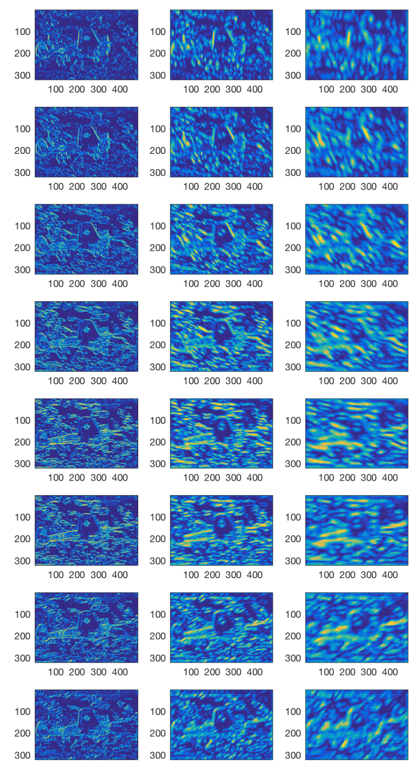
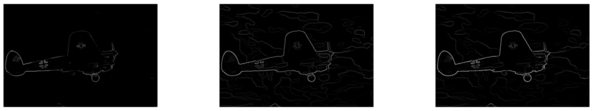


Fig. 11: Image brightness gradients when applying differnt half-discs

E. Pb-Lite boudary detection

With all the gradients acquired, Pb edges can be detected by combing the gradients information with baseline method.



(a) SobelPb (b) CannyPb (c) PbLite

Fig. 12: Edge detection result of Sobel, Canny and PbLite

Since the gradient values are computed for half-discs on all orientations and scales, before combining with baseline detections, maximum responses of the gradients over orientations are taken according to Ref. [1]. Here, by following the simple equation provided as follows

$$PbEdges = (tg + bg + cg) .* (w_1 * CannyPb + w_2 * SobelPb) \quad (3)$$

and simply set the weights for both baselines as 0.5, the features can be combined and pb edges can be extracted. The Pb Lite Outputs of image 1 in the image set is shown in Fig.12

F. Evaluation results

By using the aforementioned method, the generated perfomance probability of boundary can now be evaluted and compared to other baselines and methods. The evaluation result of my pb detector is shown in Fig. 13.

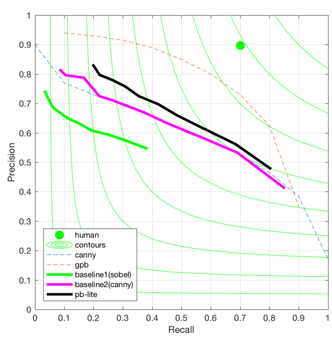


Fig. 13: Precision Recall curve

(Pb_Lite) Boundary:
ODS: F(0.68, 0.56) = 0.62 [th = 0.18]
OIS: F(0.73, 0.54) = 0.63

(Canny) Boundary:
ODS: F(0.69, 0.53) = 0.60 [th = 0.18]
OIS: F(0.73, 0.53) = 0.61

(Sobel) Boundary:
ODS: F(0.38, 0.55) = 0.45 [th = 0.09]
OIS: F(0.38, 0.55) = 0.45

It can be noticed that by combining texture, brightness and color gradients with baselines, Pb_Lite detector is fully implemented and has a fair performance with a higher F score and outperforms the two baseline methods.

IV. Conclusion

In this project, problems including color segmentation and edge detection are efficiently solved. In the first part, pins of different colors can be detected and segmented by applying knowledge in color spaces, morphological operations, and k-means clustering. Even the white pin in the test image whose color is similar to the background is successfully segmented. In the second part, different filter banks and multi-scale oriented half-disk masks are generated for encompassing gradient information into the probability of boundary. A pb detector with high F score is then fully implemented which combines the texture, brightness and color gradients with Sobel and Canny baselines. At the same time, I mastered the related matlab functions and methods for generating filters and disks, as well as k-means clustering, etc. However, there are still much work can be done such as to detect the transparent pin and dynamically allocating weights for baselines.

References

[1] Pablo Arbelaez, Michael Maire, Charless Fowlkess, and Jitendra Malik, *Contour detection and hierarchical image segmentation*. 33(5):898-916,2011

[2] Visual Geometry Group, University of Oxford.(2004, September 1). *Texture Classification*. Retrieved January 27, 2017, from http://www.robots.ox.ac.uk/ vgg/research/texclass/without.html