Homework 0: Alohamora!

Abhinav Modi
Masters of Engineering in Robotics
University of Maryland, College Park
Email: abhi1625@umd.edu
USING 2 LATE DAYS

***Abstract*—The abstract goes here.**

I. Introduction

This demo file is intended to serve as a "starter file" for IEEE conference papers produced under LaTeX using IEEEtran.cls version 1.8b and later. I wish you the best of success.

mds

August 26, 2015

A. Subsection Heading Here

Subsection text here.

1) Subsubsection Heading Here: Subsubsection text here.

II. PHASE 1: SHAKE MY BOUNDARY

In this section we perform boundary detection using a method called "PB-Lite". The outline of the process if shown in the figure below:
The first step in this process is to filter the image and find a texton map which is essentially the texture map of the image. This is computed by clustering the filter responses with K-Means clustering.

Filtering is used to access the low level features in the image. This helps us to measure and aggregate the regional texture, brightness and color properties. Different scales and orientations of a particular filter are used so that various types of textures can be addressed. Here, we have used three filter banks namely: Oriented DoG filters, Leung-Malik Filters and Gabor Filters.

A. Oriented DoG Filters

The Oriented Difference of Gaussian Filter is generated by taking the difference of two normal Gaussian Filters with same variance but the centers are of each is shifted by an amount equal to the standard deviation. This filter can also be created by convolving a simple Sobel Filter and a Gaussian kernel. The figure below shows the gaussian filter bank used for generating the results shown in the future sections.

The filter bank is generated by using 3 different scale values and 15 orientations for each scale, linearly-spaced from $0\,\mathrm{deg}$ to $360\,\mathrm{deg}$. Hence, the total number of filters is $3 * 15 = 45$.

B. Leung-Malik Filters

The Leung-Malik filter bank is a collection 48 filters with multiple scales and orientations. It consists of first and second order derivatives of Gaussians, Laplacian of Gaussian(LOG) filters and 4 Gaussian filters. All these filters account for different types of features in the image. The filter bank is shown in the figure below:

C. Gabor Filters

Gabor filters are inspired based on the way human visual system. A Gabor filter is generated by modulating a gaussian kernel with a sinusoidal plane wave. This is a linear filter that basically analyses if there is any specific frequency(governed by λ) content in the image in specific directions around the point of interest. The Gabor filterbank used in this project is shown in the figure below: The filter bank with is generated for λ=1 with three scales:[9,16,25]. Also, for each scale value, 15 filters with different orientations, uniformly spaced from 0 to 360 degrees are generated.

D. Texton Map $\mathcal{T}$

Once the filtering process of the image is complete, we end up with a stack of images of size $m \times n \times N$, where m, n are the dimensions of the image and N is the total number of filters used. Thus, each pixel value can now be represented as a distribution of these N values. Each distribution is then represented by a unique Texton-ID. These different distributions for all the pixels are then clustered into K textons using K-Means. This generates an image which captures the texture changes in the original image and can be seen below:

E. Brightness Map $\mathcal{B}$

The Brightness map captures the changes in intensity of light in the image. Similar to Texton map generation the K-Means clustering of the grayscale image is performed for K=16 and the output can be seen below:

F. Color Map $\mathcal{C}$

The Color map captures the changes in color/ chrominance in the image. The color values were clustered using K-Means clustering into 16 clusters. This generates an output image which can be seen below:

[height=6, width=8]canny.png

Fig. 1. (a)Canny Baseline, (b)Sobel Baseline (c)Pb-lite output

G. Gradient Maps

The Maps generated above are used to calculate gradient maps $\mathcal{T}_{\}}$,$\mathcal{B}_{\}}$ and $\mathcal{C}_{\}}$. These maps encode the texture, brightness and color distributions changing at each pixel. These are generated by comparing the values at each pixel by convolving the image with a left/right half-disc pair centered at the pixel. The basic concept behind this is that if the values are similar the gradient should be small and if the values are dissimilar, the gradient will be large.

The half-disks are generated by multiplying an array of size equal to the radius/scale of the circular disk with all values which lie inside this radius equal to 1 and rest 0, with an array of equal size but where one half of the array is 0s and the other half consists of 1s. This multiplication results in a half-disk which can be rotated to produce the desired half-disk mask.

Here if you rotate the disk after you've multiplied the two arrays will result in pixel voids. This can be avoided by rotating the rectangular block matrix of 0s and 1s and then by applying a "logical OR" operator on them. This gives the required half-disk masks which are shown below:

Using the above generated Half-Disk masks we compute the $\mathcal{T}_{\}}, \mathcal{B}_{\}}$ and $\mathcal{C}_{\}}$ maps by comparing the distributions generated using each half-disk pair with a χ^2 measure. The binning scheme is defined for $\mathcal{K}$ indexes which is equal to the number of K-Means clusters for each $mathcalT, mathcalB, mathcalC$. This procedure is repeated for all the half-disk pairs to generate a 3D matrix of size $m \times n \times N$ where m, n are the dimensions of the image and N is the number of filters.

The output of the mean of each $\mathcal{T}_{\}}, \mathcal{B}_{\}}$ and $\mathcal{C}_{\}}$ is given as follows:

H. PB-Lite Output

Finally, the gradient maps generated are combined with Canny and Sobel baselines using the equation:

$$PbEdges = \frac{(\mathcal{T}_{\}} + \mathcal{B}_{\}} + \mathcal{C}_{\}})}{3} \odot (w_1 * cannyPb + w_2 * sobelPb) \tag{1}$$

The $\odot$ is the Hadamard operator which is the element-wise multiplicatin of the arrays in the equation. The choice of the weights w_1 and w_2 is based on the canny and sobel baselines and the features we want from each baseline. The only constraint is that $w_1 + w_2$ = 1. The Canny and Sobel outputs and the resulting Pb-lite outputs are shown in the figures below:

mds
August 26, 2015

I. Subsection Heading Here

Subsection text here.

1) Subsubsection Heading Here: Subsubsection text here.

III. Conclusion

The conclusion goes here.

Acknowledgment

The authors would like to thank...

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.