CS 294-2, Grouping and Recognition (Prof. Jitendra Malik) October 6, 1999
Lecture #12 (Analyzing Texture) **DRAFT** Notes by Laura Walker

- Gaussian Filters
- Characterizing Texture
- Texture Segregation

1 Gaussian Filters

Gaussian filters can be used to create textons that are image dependent. The filters are constructed in 2D by multiplying a gaussian G by its first derivative G' or its second derivative G''. Typically, we will use even and odd symmetric filters at 6 orientations and 3 scales, giving us a total of 36 filters. A typical aspect ratio for 2D filters is 3:1. By convolving the image with these 36 filters, we will obtain a response vector at each pixel that corresponds to the number of filters.

Equation of Gaussian:

$$G_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-x^2}{2\sigma^2}}$$

Edge Detection Filter:

$$f(x,y) = G_{\sigma_1}(x) G'_{\sigma_2}(y)$$

Bar Detection Filter:

$$f(x,y) = G_{\sigma_1}(x) G''_{\sigma_2}(y)$$

To obtain rotated copies of the filters:

$$Rot_\theta f(x', y') = f(x\cos\theta - y\sin\theta, x\sin\theta + y\cos\theta)$$

$$\theta = [0, \frac{\pi}{6}, \frac{\pi}{3}, \frac{\pi}{2}, \frac{2\pi}{3}, \frac{5\pi}{6}]$$

The three filter scales are $\frac{1}{2}$ octave apart (one octave step corresponds to a doubling in frequency), for example, use $\sigma_2 = 1$ pixel, $\sqrt{2}$ pixels, 2 pixels. Furthermore, filter values beyond 3σ are assumed to be zero.

2 Characterizing Texture

After convolving our images (available on the course website and the corel database elib.cs.berkeley.edu) with the 36 filters from above, we obtain a vector $\vec{R}$ of 36 filter outputs at every pixel. Texture can then be characterized by the distribution of $\vec{R}$.

Texture is a regional property, so we must characterize filter responses over a region (say, window W) of the image. What are the different strategies for doing this?

2.1 Mean Response

Compute the mean value of each filter output, perhaps after taking the square (response energy).

$$\sum_W \frac{R_1^2}{k}, \sum_W \frac{R_2^2}{k}, ..., \sum_W \frac{R_{36}^2}{k}$$

where k is the number of pixels in the window W. The various windows can then be compared using the Eulidean norm of the response vectors $\vec{R}$.

2.2 Marginal Histograms

First, consider a random variable X that can take on the value of 1 or 2. The histogram for this variable is simply a count of its frequency, i.e. how many times does $X = 1$ and how many times does $X = 2$? If X can take on real values, we divide the values into a set number of bins, then count occurences. For example, if X can take on values from 1 to 20, then we might divide the results into 5 bins. Every time X is between 1 and 4, a count is added to the first bin, and so on.

In our problem, the total number of counts (for a filter response value) will equal the number of pixels in the window W. We can then convert the resulting histogram into a probability distribution by dividing by the total number of pixels (total probability = 1). This probability distribution is called the "marginal histogram". We can now compute a response profile over the window for each filter. This is a richer representation of the filter outputs than the previous strategy, because the mean response can still be calculated from the data.

2.3 Joint Probability Density

The aformentioned histogram is called "marginal" becuase the filter responses $R_1, R_2, ...R_{36}$ are NOT independent. We are really taking a measure of the joint probability distribution. The distribution of any single filter response can still be derived.

How many numbers do we need to store for these characterizations of texture? Let's assume a window size of 10 x 10 pixels, or 100 pixels total. For the mean response approach, we'll need only 36 numbers, derived from the 100 pixels. For the marginal histogram, the numbers needed depends on the number of bins, so if we used a quantization of 10 bins, we would need 360 numbers to represent the responses. For the joint probability distribution, we have 10^{36} bins... or alternatively, we have 100 pixels, so 3600 numbers are needed. Most of these bins will be zeros, but subsequent analysis is difficult because the data will be noisy.

In the end, what we really need is more data then parameters. This is called the CURSE OF DIMENSIONALITY!

2.4 Mixture of Gaussians and Kmeans

Because most of the bins are empty in the joint probability distribution, we should find a way to focus on the bins that contain useful information. One approach is to forget about counting all the bins, instead, find clusters in the data. When fitting a mixture of Gaussians in EM, we "magically" know the number of clusters, place their centers, and move the centers based on the population mean.

Kmeans can be implemented to find the data clusters in 36 dimensions. A K of about 25 usually works. Each of the clusters is then called a TEXTON. Now assign each pixel to a cluster, thereby partioning the image into 25 channels. To review, first we find 25 clusters in R^{36}. Second, each cluster center is a point in R^{36}, we call this a texton. Third, every pixel in the image can be mapped to the nearest texton.

Hard clustering uses just the centers as textons, while soft clustering uses a mixture of Gaussians, where each pixel has probabilities of belonging to different clusters. One way to assign pixels to their textons is to minimize the sum of square of errors of distances to the cluster center. The big thing to remember, is that these derived textons are IMAGE DEPENDENT.

3 Texture Segementation

This topic is continued in the following lecture (lecture 13). An approach to segmentation with texture is through the use of texton histograms. Texton histograms can be constructed for overlapping windows and then compared to see if the two windows contain the same texture. A good comparison is to use a χ^2 test.

$$\chi^2(h_i, h_j) = \frac{1}{2}\sum_{m=1}^{K}\frac{[h_i(m) - h_j(m)]^2}{[h_i(m) + h_j(m)]}$$

where K is the number of clusters and h_i and h_j are the histograms over windows i and j respectively.

When constructing the weighting function for segementation, texture can now be incorporated easily.

$$w_{ij} = exp[\frac{-\chi^2(h_i, h_j)}{\sigma_{tex}}]$$

The choice of the number of textons or clusters to use becomes a trade-off between BIAS and VARIANCE.

......