Analysis and Synthesis of Texture

CMPE 264: Image Analysis and Computer Vision
Hai Tao
Extracting image structure by filter banks

- Represent image textures using the responses of a collection of filters
- An appropriately designed filter bank will extract useful information such as spots and edges
- An example of filter banks and the responses
Extracting image structure with filter banks

- The responses of an image to the previous filter bank
- Based on the pixels with large magnitudes in the particular filter response, we can determine the presence of strong edges of certain orientation. We can also find spot patterns from the responses of the first two filters
Extracting image structure with filter banks

Filtering can be performed at different scales to find patterns of different sizes. Here, the responses of the low-resolution version of the original image is shown.
Filter banks as weighted sums of Gaussian filters

- The following eight filters each is the sum of several weighted Gaussian filters
  - The first spot filter is the sum of Gaussian filters with sigmas of 0.62, 1, and 1.6, and weights of 1, -2, 1.
  - The second spot filter is the sum of Gaussian filters with sigmas of 0.71, 1.14, and weights of 1, and -1
  - The six bar filters are the rotated versions of the horizontal bar, which is the weighted sum of three Gaussian filters, each has sigma_x of 2, and sigma_y of 1, with centers at (0,1), (0,0), and (0,-1)
Gabor filters

- Gabor filters are the products of a Gaussian filter with oriented sinusoids. Gabor filters come in pairs, each consisting of a symmetric filter and an anti-symmetric filter.

\[
G_{\text{Symmetric}}(x, y) = \cos(k_x x + k_y y) \exp\left\{-\frac{x^2 + y^2}{2\sigma^2}\right\}
\]

\[
G_{\text{Antisymmetric}}(x, y) = \sin(k_x x + k_y y) \exp\left\{-\frac{x^2 + y^2}{2\sigma^2}\right\}
\]

- \((k_x, k_y)\) determines the spatial frequency and the orientation of the filter and \(\sigma\) determines the scale of the filter.

- A filter bank is formed by varying the frequency, the scale, and the filter orientation.
Gabor filters

- Example
Texture analysis with oriented pyramid

- The Gaussian pyramid

\[ P_{\text{Gaussian}}(I)_0 = I \]
\[ P_{\text{Gaussian}}(I)_k = \downarrow G(P_{\text{Gaussian}}(I)_{k-1}) \]

where \( G() \) is a Gaussian filter, \( \downarrow \) is a downsampling operator.
Texture analysis with oriented pyramid

- The Laplacian pyramid

\[ P_{\text{Laplacian}}(I)_M = P_{\text{Gaussian}}(I)_M \]
\[ P_{\text{Laplacian}}(I)_k = P_{\text{Gaussian}}(I)_k \uparrow P_{\text{Gaussian}}(I)_{k+1} \]
\[ \uparrow \text{ is an upsampling operator} \]
The oriented Laplacian pyramid

- For each level of the Laplacian pyramid of an image, apply filters with different orientations and the same scale to obtain oriented pyramid

- Oriented filters are designed so that the original image can be reconstructed from the filter response
Texture synthesis

- The problem of texture synthesis
  - Generation of large texture images from a small example so that the synthetic texture has a similar appearance as the example
Texture synthesis by matching histograms of filter responses

- **Motivation:** if two homogeneous texture samples are drawn from the same probability model, then histograms of the outputs of various filters applied to the samples will be the same.

- **Histogram_Matching_Algorithm**
  - Make a working image from noise.
  - Match the working image histogram to the example image histogram.
  - Make a pyramid from the example image.
  - Until convergence:
    - Make a pyramid from the working image.
    - For each level in the two pyramid:
      + Modify the the working image so that the histograms matches.
    - Synthesize the working image from the working pyramid.
Matching the histograms

- Histogram equalization

![Graph showing histogram equalization](diagram.png)
Texture synthesis by matching histograms of filter responses

- Some good results
Texture synthesis by matching histograms of filter responses

- Some bad results
  - The method does not capture the dependency between the filter responses of different locations and scales
Texture synthesis by sampling the conditional densities of filter responses

- Motivation: capture the dependency between filter responses at different scales using the so called conditional histogram
- Conditional histogram is a representation of the conditional distribution on values at certain level, conditioned on the values at coarser levels
- Conditional_Densities_Algorithm
  - Make a pyramid Pe from the example image
  - Make an empty working pyramid Pw for the image to be synthesized
  - Pw(M)=Pe(M). If Pw is larger than Pe, replicate copies of Pe to fill Pw
  - For each other layer of Pe, going from coarsest to finest
    - For each element e of the layer
      + Obtain all elements with the same parent structure (corresponding pixels at coarser scales)
      + Choose one of this collection uniformly at random
      + Insert the value of this element into e
    End
  End
- Synthesize the texture image from Pw
Texture synthesis by sampling the conditional densities of filter responses

- Some results
Texture synthesis by sampling the conditional densities of filter responses

- Comparison with the histogram equalization method