Gabor filters: Esercitazione

Benedetta Barbetti – Michaela Servi

Università degli studi di Firenze

May 2013

Two exercises

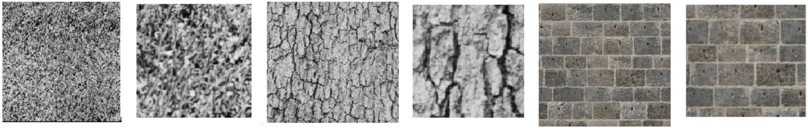
- Filter Bank Approach for Texture Similarity Measure


- Optimal Gabor Filter for Specific Texture Recognition


Filter Bank Approach

Data Set

- 1 set of bitmaps is available in the folder DataSet
- Each bitmap rappresent a texture



Project Goal

- Define a procedure to compute the features vector for each bitmap
- Compute the distance matrix between all the images in the data set
- Compare the matrix values with the corresponding textures

Guidelines

- Read the bitmaps
- Inizialize the values for ψ and γ
- Create the values vector for the three parameters θ, λ, σ

```
psi     = [0 pi/2];
gamma   = 0.5;
%numero di valori per theta
N = 4;
theta = zeros(N);
%numero di valori per lambda
M = 2;
lambda = zeros(M);
%numero di valori per sigma
P = 2;
sigma = zeros(P);
%creazione dei vettori di valori per i tre parametri
for i =1:N
   theta(i+1) =  theta(i) + pi/N;
end

for i =1:M
   lambda(i) =  i * 2;
end

for i =1:P
   sigma(i) = 2^i;
end
```

Guidelines

- Create for each texture the features vector

```
%creazione del vettore di features
filter_number = N*P*M;
feature_vector= zeros(2*filter_number, size(galleryNames,1));
for j = 1:size(galleryNames, 1)
    k=1;
    %lettura dell'immagine
    img_in = im2double(imread(galleryNames{j}));
    for n = 1:N
        for m = 1:M
            for p = 1:P
                [e_mean, e_var] =E_Features( img_in, theta(n),lambda(m),sigma(p), psi, gamma);
                feature_vector(k,j)=e_mean;
               feature_vector(k+1,j)=e_var;
                k=k+2;
            end
        end
    end
end
```

- Implement `E_features`

E_features

- Compute the real and immaginary part of the Gabor filter:

```
gb_R = B_gabor_fn2(par(1), par(2), par(3), psi(1), gamma);
gb_I = B_gabor_fn2(par(1), par(2), par(3), psi(2), gamma);
```

- You have to compute the `gabor_fn`
- Use `imfilter` for the convolution between the filters and the image
- Compute the energy: $E_{\lambda,\theta} = \sqrt{C_{Re}^2 + C_{Im}^2}$
- Extract the two features mean and variance

gabor_fn

- $g_{\lambda,\theta,\psi}(x,y) = e^{-(\frac{x'^2}{2\sigma_x^2}+\frac{y'^2}{2\sigma_y^2})}\cos(\frac{2\pi}{\lambda}x'+\psi)$
- $x' = x\cos(\theta) + y\sin(\theta)$
- $y' = -x\sin(\theta) + y\cos(\theta)$
- $\sigma_x = \sigma$
- $\sigma_y = \frac{\sigma}{\gamma}$
- The two vector x and y can be set usign `meshgrid`

Distance Matrix

- Normalize the features vector
- Compute the distance matrix

```
%normalizzazione dei vettori di features
for i = 1:2*filter_number
    m = mean(feature_vector(i,:));
    sigma = std(feature_vector(i,:));
    for j = 1:size(galleryNames, 1)
        feature_vector(i,j) =  (feature_vector(i,j)- m)/ sigma;
    end
end


distMatrix = zeros(size(galleryNames, 1),size(galleryNames, 1));

%costruzione della matrice di distanza
for i = 1:size(galleryNames, 1)
    for j = 1: size(galleryNames, 1)
        distMatrix(i,j)= norm(feature_vector(:,i)-feature_vector(:,j));
    end
end
```

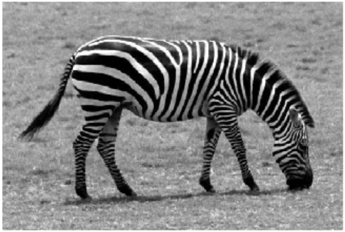
Optimal Gabor Filter

Data Set

- Two sets, each of two bitmaps:
  - One represets a specific texture; images,
  - The other is an image, that contains the texture;

Goal

- Tuned the 3 parameters of the Gabor function (θ, λ, σ) to match the texture, maximizing the mean value of $\frac{E}{\sigma^2}$
- Apply the Gabor filter with the best parameter to the second image in the dataset



Guidelines

- Initialize the value for ψ and γ
- Read the texture
- Use `fminsearchbnd` (a function that apply the Matlab function `fminsearch` with bound constraints) to extract the best parameters from the `Energy` function

```
%best par è il vettore [theta lambda sigma] dei parametri ottimali

[best_par best_val]= fminsearchbnd(@(par) -Energy(img_in, par, psi, gamma), [0 2 2], [2 0 2], [2*pi 8 16]);
```

Energy

- It's similar to `E_Features` but insted of σ λ γ use the parameter vector
- The value to return is the mean value of $\frac{E}{\sigma^2}$

Output

- Compute the Gabor filter with the best parameters, using `gabor_fn`
- Convolve the optimal filter with the second image in the dataset

```
function [  ] = optimal_filter( img_in, theta, lambda, sigma, psi, gamma)
%
% gamma = aspect ratio, (0.5)
% psi   = phase shift, [0 pi/2]
% lambda= wave length, (>=2)
% theta = angle in rad, [0 2pi]

img_out = zeros(size(img_in,1), size(img_in,2));
gb = gabor_fn(theta, lambda, sigma,psi(1),gamma) + 1i *  gabor_fn(theta, lambda, sigma,psi(2),gamma);
figure;
% gb/2 + 0.5 per ottenere l'immagine in scale di grigio
imshow(abs(gb/2+0.5));
img_out(:,:) = imfilter(img_in, gb, 'conv');
img_out = img_out./max(img_out(:));
figure;
subplot(1,2,1),imshow(img_in);
subplot(1,2,2),imshow(double(imag(img_out)));
end
```