**Gabor filter**

**Introduction**

* named after [Dennis Gabor](http://en.wikipedia.org/wiki/Dennis_Gabor)
* [linear filter](http://en.wikipedia.org/wiki/Linear_filter) used for edge detection
* Frequency and orientation representations of Gabor filters are similar to those of the human visual system, and they have been found to be particularly appropriate for texture representation and discrimination
* In the spatial domain, a 2D Gabor filter is a Gaussian kernel function modulated by a sinusoidal plane wave.
* Simple cells in the [visual cortex](http://en.wikipedia.org/wiki/Visual_cortex) of [mammalian brains](http://en.wikipedia.org/wiki/Mammalian_brain) can be modeled by Gabor functions. [image analysis](http://en.wikipedia.org/wiki/Image_analysis" \o "Image analysis) with Gabor filters is thought to be similar to perception in the [human visual system](http://en.wikipedia.org/wiki/Human_visual_system).

**Equations**:

Its impulse response is defined by a sinusoidal wave (a plane wave for 2D Gabor filters) multiplied by a Gaussian function. Because of the multiplication-convolution property (Convolution theorem), the Fourier transform of a Gabor filter's impulse response is the convolution of the Fourier transform of the harmonic function and the Fourier transform of the Gaussian function. The filter has a real and an imaginary component representing orthogonal directions. The two components may be formed into a complex number or used individually.

1. complex

g(x,y;\lambda,\theta,\psi,\sigma,\gamma) = \exp\left(-\frac{x'^2+\gamma^2y'^2}{2\sigma^2}\right)\exp\left(i\left(2\pi\frac{x'}{\lambda}+\psi\right)\right)

1. real

g(x,y;\lambda,\theta,\psi,\sigma,\gamma) = \exp\left(-\frac{x'^2+\gamma^2y'^2}{2\sigma^2}\right)\cos\left(2\pi\frac{x'}{\lambda}+\psi\right)

1. imaginary

g(x,y;\lambda,\theta,\psi,\sigma,\gamma) = \exp\left(-\frac{x'^2+\gamma^2y'^2}{2\sigma^2}\right)\sin\left(2\pi\frac{x'}{\lambda}+\psi\right)

where

x' = x \cos\theta + y \sin\theta\, , y' = -x \sin\theta + y \cos\theta\,

* = wavelength

= orientation of the normal to the parallel stripes of a Gabor function



* = phase offset,

\sigma   = sigma/standard deviation of the Gaussian envelope

\gamma  = spatial aspect ratio, and pecifies the ellipticity of the support of the Gabor function

**Matlab code:**

sigma\_x = sigma;

sigma\_y = sigma/gamma;

nstds = 3; % range of the gaussian = -3sigma to 3sigma

xmax = max(abs(nstds\*sigma\_x\*cos(theta)),abs(nstds\*sigma\_y\*sin(theta)));

xmax = ceil(max(1,xmax));

ymax = max(abs(nstds\*sigma\_x\*sin(theta)),abs(nstds\*sigma\_y\*cos(theta)));

ymax = ceil(max(1,ymax));

xmin = -xmax; ymin = -ymax;

[x,y] = meshgrid(xmin:xmax,ymin:ymax);

x\_theta=x\*cos(theta)+y\*sin(theta);

y\_theta=-x\*sin(theta)+y\*cos(theta);

gb= exp(-.5\*(x\_theta.^2/sigma\_x^2+y\_theta.^2/sigma\_y^2)).\*cos(2\*pi/lambda\*x\_theta+psi);

figure(1);

imshow(gb);

figure(2);

[rows, cols] = size(gb);

[x\_image, y\_image] = meshgrid(1:cols, 1:rows);

surf(x\_image,y\_image,gb)

colorbar

Explanation:

1. scale sigma y to make elipse

sigma\_y = sigma/gamma;

1. find the maximum values for x and y

rnage of gaussian = 3 sigma

x' = x \cos\theta + y \sin\theta\,

so take the components

1. nstds\*sigma\_x\*cos(theta)
2. nstds\*sigma\_y\*sin(theta)

find the maximum out of those values

similarly do for y.

1. min = -max
2. create a mesh with all possible values of x and y in the range from min to max

[x,y] = meshgrid(xmin:xmax,ymin:ymax);

1. find the x’ and y’ components for these input values

x\_theta=x\*cos(theta)+y\*sin(theta);

y\_theta=-x\*sin(theta)+y\*cos(theta);

1. calculate gabor filter with all those x and y values

gb= exp(-.5\*(x\_theta.^2/sigma\_x^2+y\_theta.^2/sigma\_y^2)).\*cos(2\*pi/lambda\*x\_theta+psi);

[use the real part of the gabor function]

|  |  |  |  |
| --- | --- | --- | --- |
| test | parameters | gb plot (2d) | corresponding surface plot |
| 1 | sigma = 3;  gamma = 1;  theta = 0;  lambda = 1;  psi = 0; | 19x19 |  |
| 2 | increasing sigma.  sigma = 5;  gamma = 1;  theta = 0;  lambda = 1;  psi = 0; | 31x31 |  |
| 3 | changing gamma  sigma = 3;  gamma = 2;  theta = 0;  lambda = 1;  psi = 0; | 11x19 |  |
| 4 | changing psi  sigma = 3;  gamma = 2;  theta = 0;  lambda = 1;  psi = 0.5; | 11x19 |  |
| 5 | increasing psi  sigma = 3;  gamma = 2;  theta = 0;  lambda = 1;  psi = 1; | 11x19 |  |
| 6 | changing theta  sigma = 3;  gamma = 2;  theta = pi / 2;  lambda = 1;  psi = 0; | 19x11 |  |
| 7 | changing theta  sigma = 3;  gamma = 2;  theta = pi / 3;  lambda = 1;  psi = 0; | 17x11 |  |
| 8 | changing theta  sigma = 3;  gamma = 2;  theta = pi / 6;  lambda = 1;  psi = 0; | 11x17 |  |
| 9 | showing fourier plot for lambda |  |  |
| 10 | changing lambda  sigma = 3;  gamma = 2;  theta = 0;  lambda = pi/6;  psi = 0; |  |  |
| 11 | changing lambda  sigma = 3;  gamma = 2;  theta = 0;  lambda = pi/4;  psi = 0; |  |  |
| 12 | changing lambda  sigma = 3;  gamma = 2;  theta = 0;  lambda = pi/3;  psi = 0; |  |  |
| 13 | changing lambda  sigma = 3;  gamma = 2;  theta = 0;  lambda = pi/2;  psi = 0; |  |  |
| 14 | changing lambda  sigma = 3;  gamma = 2;  theta = 0;  lambda = 4 \* pi/6;  psi = 0; |  |  |

* to plot the fourier transform:

G2 = fft2(gb);

imshow(log(abs(fftshift(G2)) + 1), [])

frequency = 1/ wavelength

Observations:

* Looking at test 1: the sigma determines the range of the function. width of gb = range of Gaussian = 6 \* sigma + 1.
* Looking at test 1 and 2: increasing sigma increases the width of the gabor plot.

\sigma   = sigma/standard deviation of the Gaussian envelope

* Looking at test 1 and 3: test 1 has full Gaussian whereas as test 2 has Gaussian squashed in half of the space. It only changes the y component.

\gamma  = spatial aspect ratio, and pecifies the ellipticity of the support of the Gabor function

sigma\_y = sigma/gamma;

that’s why only y direction is affected by gamma

* Looking at 3,4,5: increasing psi makes it more dimmer (more shorter)

= phase offset,



cos(2\*pi/lambda\*x\_theta+psi)

so psi is added to the angle, that’s why it becomes shorter. cos function is decreasing function.

* Looking at 3,6,7,8: changing theta changes the orientation of the Gaussian. values other than 0 and 90 cause the shape to distort.

\theta = orientation of the normal to the parallel stripes of a Gabor function

xmax = max(abs(nstds\*sigma\_x\*cos(theta)),abs(nstds\*sigma\_y\*sin(theta)));

x\_theta=x\*cos(theta)+y\*sin(theta);

so according the orientation of the curve changes.

* Looking at 9-14: changing the lambda, separates the two principle frequencies more.

\lambda = wavelength

gb= exp(-.5\*(x\_theta.^2/sigma\_x^2+y\_theta.^2/sigma\_y^2)).\*cos(2\*pi/lambda\*x\_theta+psi);

Applying gabor filter on image

 