CIS580 Machine Perception

Homework 5
Apr.15.19

Chun Chang

1 Gabor 1d

(a) Gaussian 1d

```
% use gaussian1d.m
n = -(len-1)/2:(len-1)/2;
gau = exp(-n.^2/(2*sigma^2))/(sigma * sqrt(2*pi));
gau_norm = gau / sum(gau);

g = gau_norm;
```

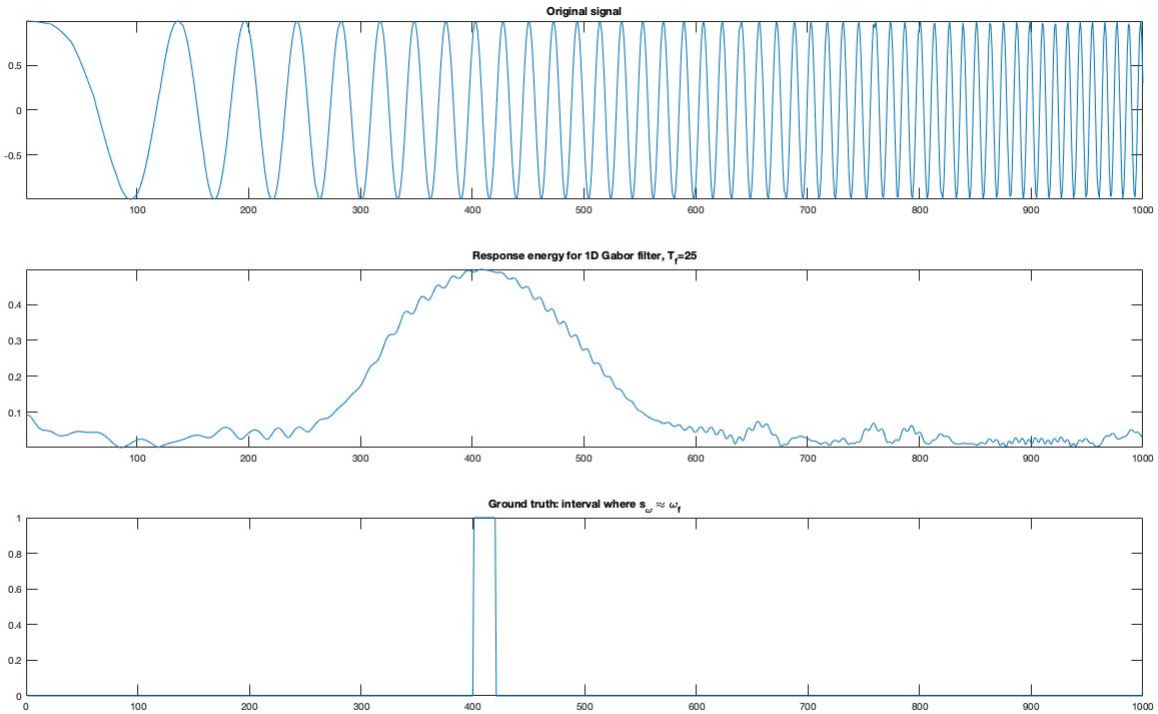
(b)gabor 1d

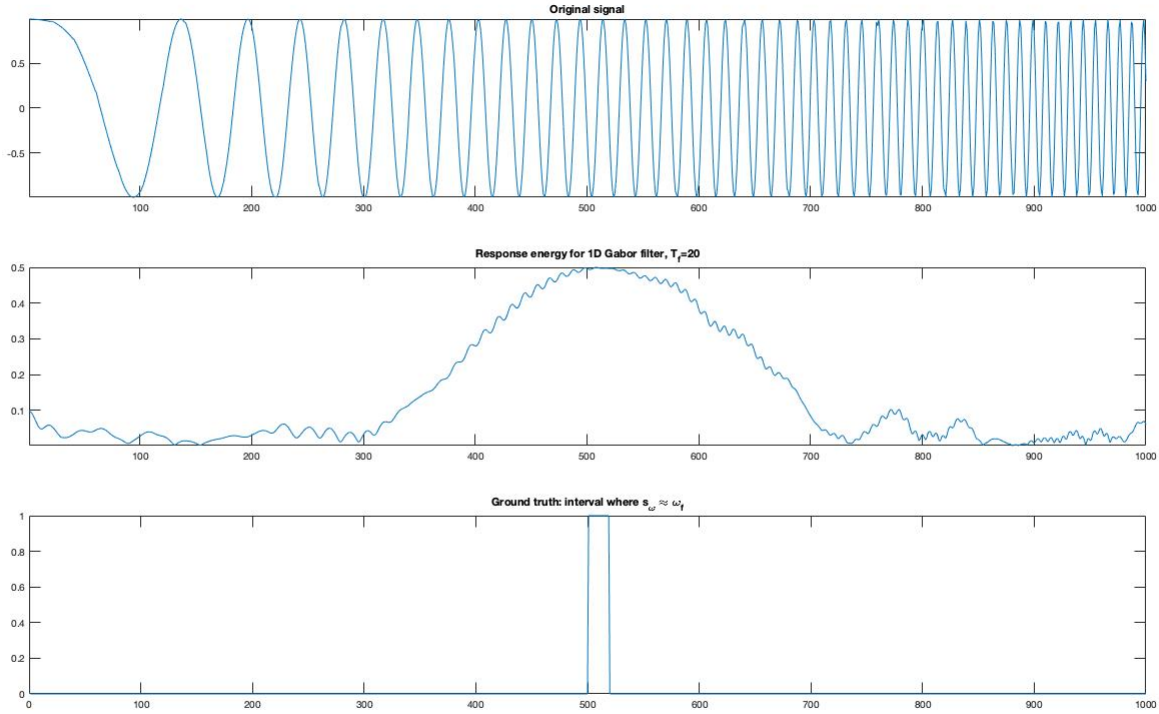
```
n = -(len-1)/2:(len-1)/2;
X = gaussian1d(sigma, len);
filter_cos = X .* cos(n*2*pi / T_f);
filter_sin = X .* sin(n*2*pi / T_f);
```

(c) Gabor script

```
% Your code goes here %%%%%%%%%%%%%%%%%%%%
r1 = conv(s,g2,'same');
r2 = conv(s,g1,'same');
% energy = abs(r1) + abs(r2);
energy = sqrt(r1.^2 + r2.^2);
% End of your code %%%%%%%%%%%%%%%%%%%%%%%%
```

(d) two distinct time periods (20, 25)




Original signal
0.5
0
-0.5
100
200
300
400
500
600
700
800
900
1000
Response energy for 1D Gabor filter, T_f=20
0.5
0.4
0.3
0.2
0.1
100
200
300
400
500
600
700
800
900
1000
Ground truth: interval where $s_\omega \approx \omega_f$
1
0.8
0.6
0.4
0.2
0
0
100
200
300
400
500
600
700
800
900
1000

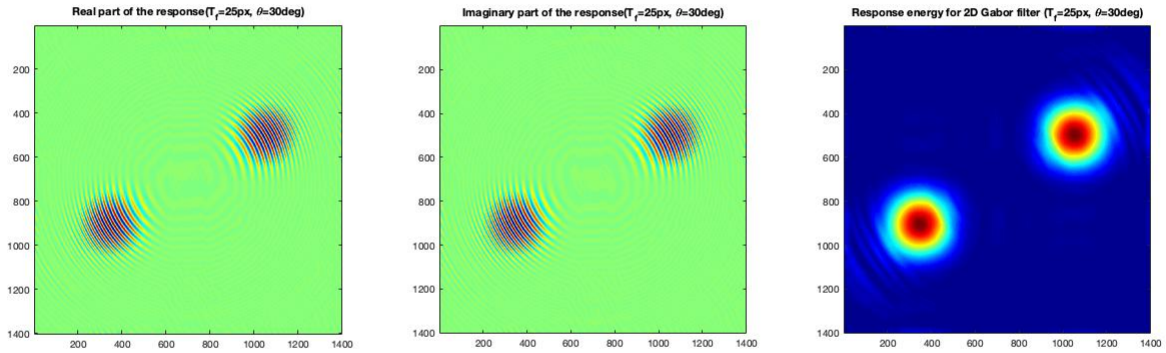
2. Gabor 2D

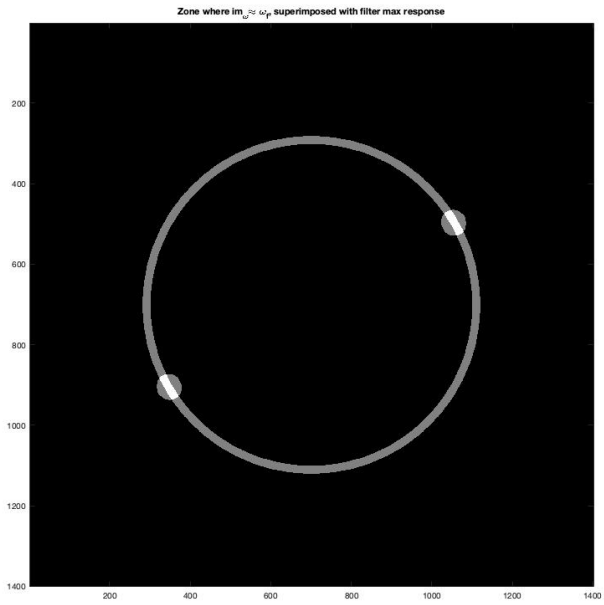
```
(a)Gaussian 2d
% Your code goes here %%%%%%%%%%%%%%%%%%%%%%
n = -(len-1)/2: (len-1)/2;
g = zeros(len,len);
for i = 1 : len
    for j = 1: len
        g(i,j) = exp(-0.5 * [n(j), n(i)] * inv(Sigma) * [n(j);n(i)])...
            / (2* pi * sqrt(det(Sigma)));
    end
end
g = g / sum(g(:));
end
% End of your code %%%%%%%%%%%%%%%%%%%%%%%%%
```

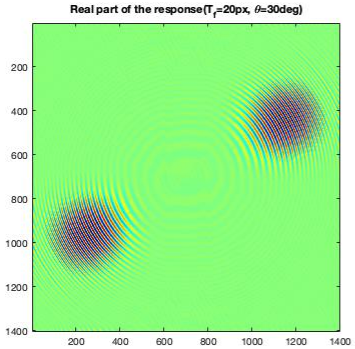
```
   (b)gabor 2d
% use gaussian2d.m
phi = theta / 180 * pi;
g = gaussian2d(Sigma, len);
n = -(len-1)/2 : (len-1)/2;
an = 2*pi/T_f *( n * cos(phi) - n'*sin(phi));
filter_cos = g .* cos(an);
filter_sin = g .* sin(an);
end
% End of your code %%%%%%%%%%%%%%%%%%%%%%%%%
```

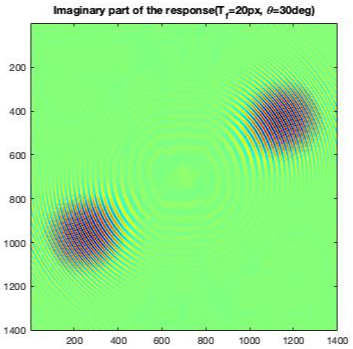
```
   (c) Gabor2d script
% Your code goes here %%%%%%%%%%%%%%%%%%%%
r1 = conv(s,g2,'same');
r2 = conv(s,g1,'same');
% energy = abs(r1) + abs(r2);
energy = sqrt(r1.^2 + r2.^2);
% End of your code %%%%%%%%%%%%%%%%%%%%%%%
```

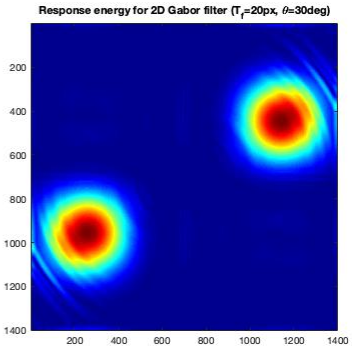
(d) different time periods (20 & 25)

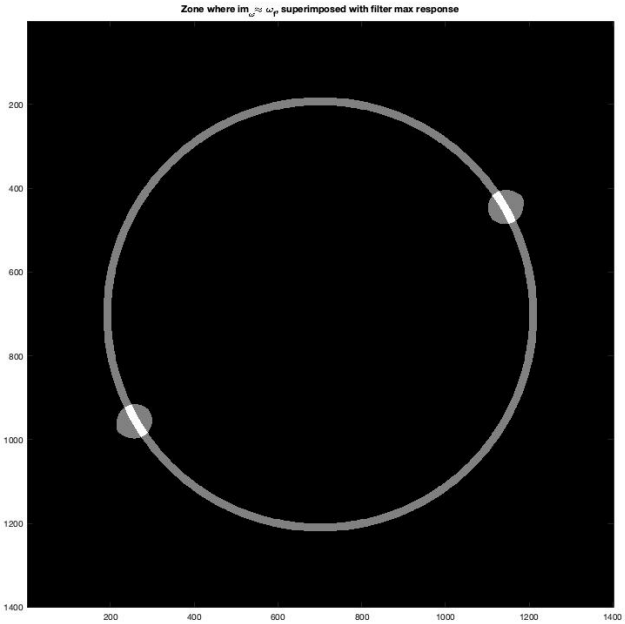



Zone where im$_{\omega} \approx \omega_r$ superimposed with filter max response
200
400
600
800
1000
1200
1400


Real part of the response(T_f=20px, θ=30deg)
200
400
600
800
1000
1200
1400


Imaginary part of the response(T_f=20px, θ=30deg)
200
400
600
800
1000
1200
1400


Response energy for 2D Gabor filter (T_f=20px, θ=30deg)
200
400
600
800
1000
1200
1400


Zone where im$_{\omega} \approx \omega_r$ superimposed with filter max response
200
400
600
800
1000
1200
1400

2 Scale invariant detection

1 Approximate LoG by DoG

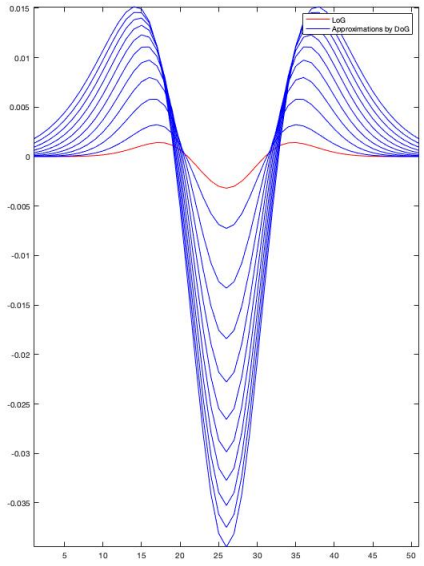
(a) DoG

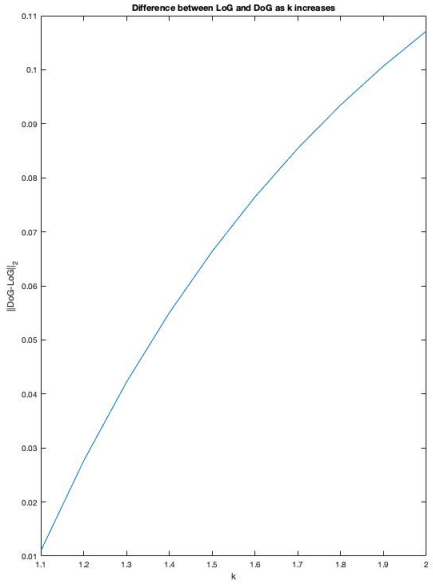
```
DoG = dog1d(sigma, k, fSize);
% use dog1d.m
LoG_approx(i,:) = DoG;
```

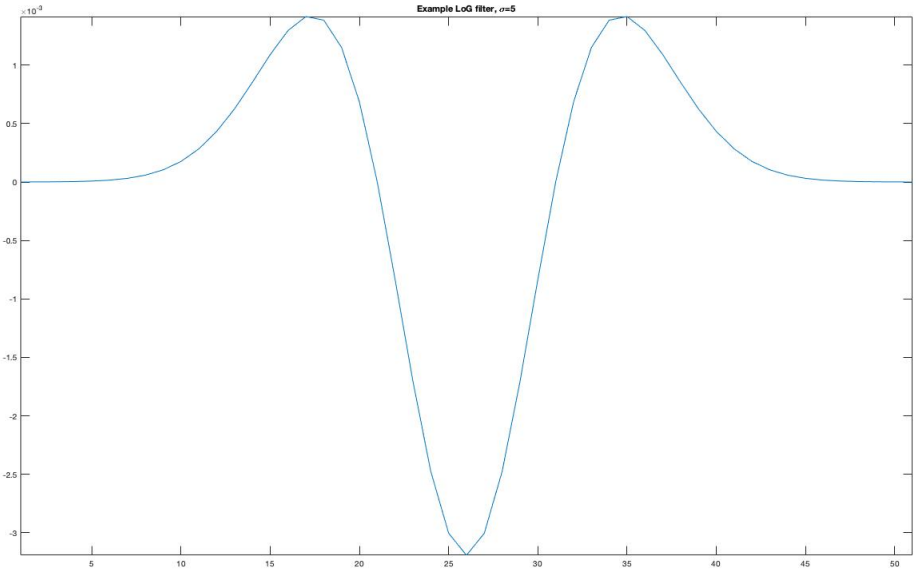
Dog 1d

```
dog = gaussian1d(k*sigma, len) - gaussian1d(sigma,len);
```

Quick comment: Difference of Gaussian is closer to laplace of gaussian when the coefficient k approaches to 1







(b)

$$D_o G \equiv \frac{g(x, k\sigma) - g(x, \sigma)}{\sigma(k-1)} \cong \frac{\partial g}{\partial \sigma} d\sigma, \text{ when } k \to 1, \quad \frac{1}{(k-1)\sigma} \simeq \frac{1}{d\sigma}$$

$$L_o G = \frac{\partial^2 g}{\partial x^2} = \frac{\partial^2 g}{\partial \sigma^2} = \frac{\partial g(x,\sigma')}{\partial \sigma} \cdot \frac{1}{d\sigma} = \frac{\partial g(x,\sigma)}{\partial \sigma} d\sigma \frac{1}{(k-1)\sigma^2}$$

$$= \frac{1}{(k-1)\sigma^2} D_o G \text{ for } k \to 1$$

(c)

$$L_o G \cong \frac{D_o G}{(K-1)\sigma^2} \quad D_o G$$

$$(k-1)\sigma^2 L_o G \cong D_o G = G(\sigma_1) - G(\sigma_2)$$

Since we are find maximum of $D_o G$. if k is not less than 1. a positive scalar would not influence the location of maximum.

2 flower detection

(a)convolution with 1d gaussian

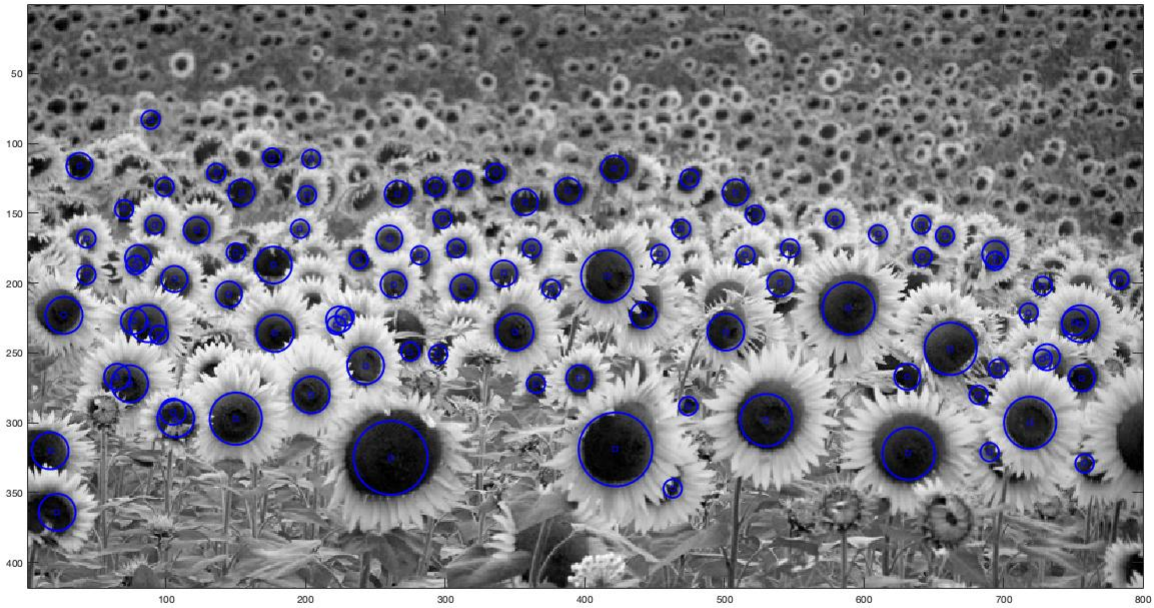
```
      % Your code starts here %%%%%%%%%%%%%%%%%%%%
    % convolve with gaussians to build the scale space
    gau_i = gaussian1d(sigma_i, round(fSize));
    scales(:,:,i) = conv2(im, gau_i, 'same' );
    scales(:,:,i) = conv2(scales(:,:,i), gau_i', 'same' );
    % End of your code %%%%%%%%%%%%%%%%%%%%%%%%%
```

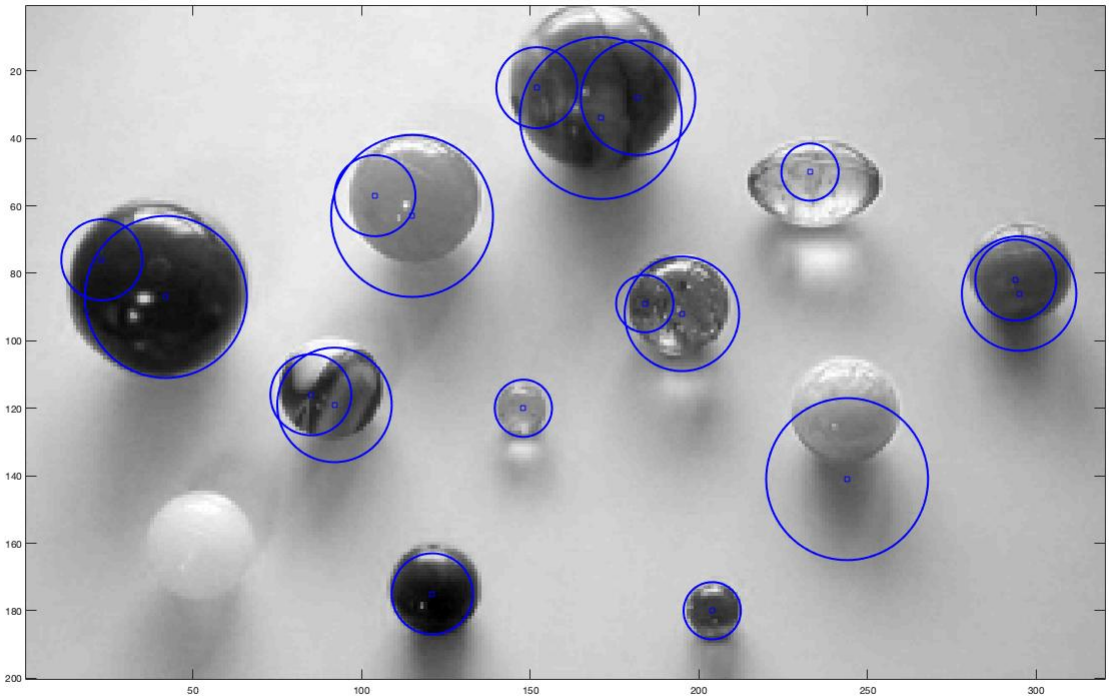
(b)find local maximum

```
ind = double.empty(0,1);
d_max = 0.5 * max(dog_max(:));

for i = 1 : size(xmax)
    if dog(ymax(i), xmax(i), smax(i)) > d_max
        ind = [ind; i];
    end
end

ymax = ymax(ind);
xmax = xmax(ind);
smax = smax(ind);
```

(c) radius

```
    % Your code goes here %%%%%%%%%%%%%%%%%%%%%%
    r = sqrt(2)*sigma*k^(smax(i));
    % End of your code %%%%%%%%%%%%%%%%%%%%%%%%%
```




20
40
60
80
100
120
140
160
180
200
50
100
150
200
250
300