CS 558: Homework Assignment 1 -- Edge Detection
Program Language: MATLAB
Name: Lan Chang
Date: 02/09/17

1. Gaussian Filtering
   (kernel size = sigma * 6 + 1)

(1) Kangaroo

σ = 1

σ = 2

(2) Plane

σ = 3

σ = 6

(3) Red

σ = 5

σ = 10

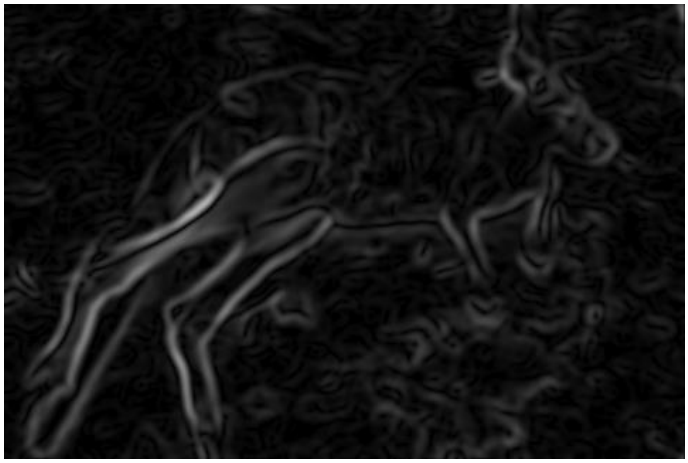
2. Gradient Computation
   (after Gaussian filtering)

(1) Kangaroo

σ = 1, threshold = 95

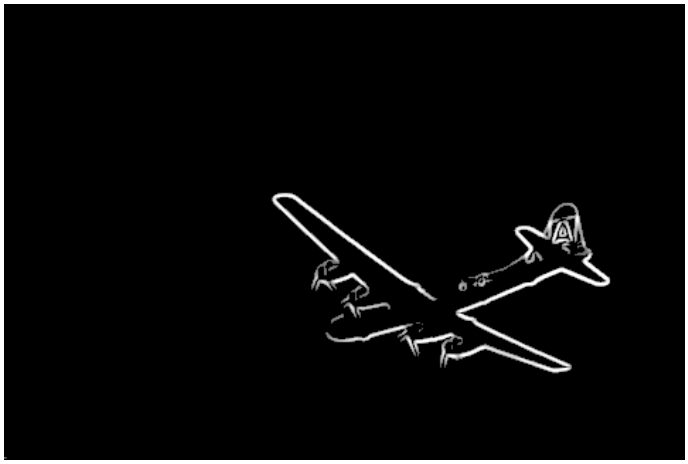
σ = 2, threshold = 50

σ = 3, threshold = 5



(2) Plane

σ = 1, threshold = 95



σ = 2, threshold = 50

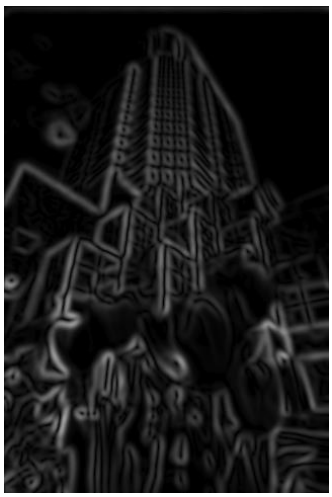
σ = 3, threshold = 5

(3) Red

σ = 1, threshold = 95

σ = 2, threshold = 50

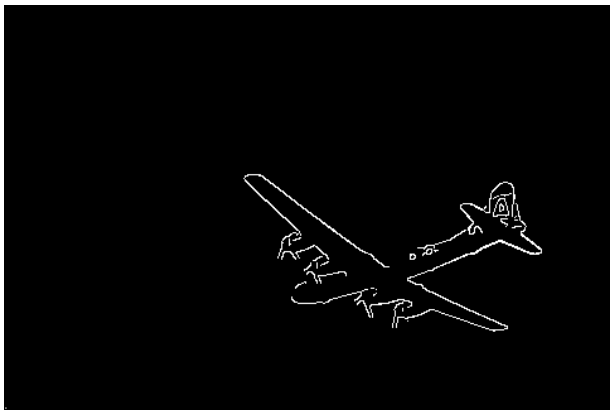
σ = 3, threshold = 5



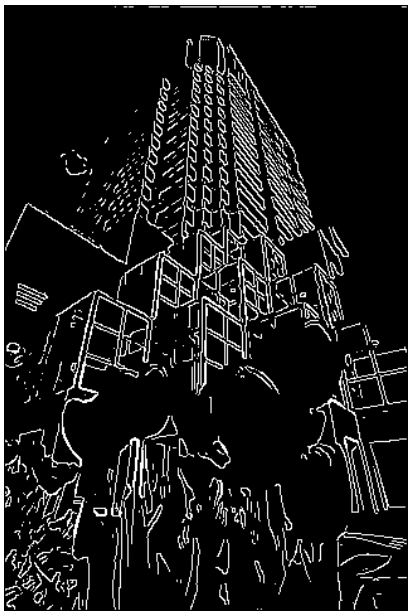
3. Non-Maximum Suppression
   (using the previous output image, σ = 1, threshold = 95)

(1) Kangaroo

(2) Plane



(3) Red



4. MATLAB Code

(1) homework1.m

```
clear all;

% input the image
image=imread('kangaroo.pgm');
%image=imread('plane.pgm');
%image=imread('red.pgm');

% output the original image
% imwrite(image,'kangaroo.bmp','bmp');
imshow(image);

% parameter
sigma=1;
threshold=95;

% width and height of image
i_height=size(image,1);
i_width=size(image,2);

% Problem 1.1: Gaussian filtering
% Gaussian filter matrix
size=sigma*6+1;
half=(size-1)/2;
[x,y]=meshgrid(-half:half,-half:half);
g_filter=exp(-(x.^2+y.^2)/(2*sigma^2))/(2*pi*sigma^2);
g_filter=g_filter./sum(g_filter(:));

% Gaussian filtering
image1=im2double(image);
image2=filtering(image1,g_filter);

% output the image after Gaussian filtering
% imwrite(image2,'kangaroo_sigma1.bmp','bmp');
figure,imshow(image2);

% Problem 1.2 Gradient computation
% Sobel filter matrix
s_filter_x=[-1,0,1;-2,0,2;-1,0,1];
s_filter_y=[1,2,1;0,0,0;-1,-2,-1];

% Sobel filtering
i_x=filtering(image2,s_filter_x);
```

```
i_y=filtering(image2,s_filter_y);

% image gradient
strength=sqrt(i_x.^2 + i_y.^2);
direction=atand(i_y./i_x);
image3=im2uint8(strength);
for i=1:i_height
      for j=1:i_width
            if image3(i,j)<threshold
                  image3(i,j)=0;
            end
      end
end

% output the image after gradient computation
% imwrite(image3,'kangaroo_sigma1_threshold95.bmp','bmp');
figure,imshow(image3);

% Problem 1.3 non-maximum suppression
% non-maximum suppression
image4=nms(image3,direction);
for i=1:i_height
      for j=1:i_width
            if image4(i,j)~=0
                  image4(i,j)=255;
            end
      end
end

% output the image after non-maximum suppression
% imwrite(image4,'kangaroo.bmp','bmp');
figure,imshow(image4);
```

(2) filtering.m

```
function image_fil=filtering(image_ori,filter)
% width and height of original image
width_i=size(image_ori,2);
height_i=size(image_ori,1);

% width and height of filter
width_f=size(filter,2);
height_f=size(filter,1);
half_wf=(width_f-1)/2;
half_hf=(height_f-1)/2;

% extend
image_temp1=zeros(height_i+half_hf*2,width_i+half_wf*2);
for i=1:height_i
    for j=1:width_i
        image_temp1(i+half_hf,j+half_wf)=image_ori(i,j);
    end
end

% replicate boundary
for i=1:height_i+half_hf*2
    for j=1:width_i+half_wf*2
        if j<=half_wf&&i>=half_hf+1&&i<=height_i+half_hf
            image_temp1(i,j)=image_temp1(i,half_wf+1);
        elseif j>=width_i+half_wf+1&&i>=half_hf+1&&i<=height_i+half_hf
            image_temp1(i,j)=image_temp1(i,width_i+half_wf);
        elseif i<=half_hf&&j>=half_wf+1&&j<=width_i+half_wf
            image_temp1(i,j)=image_temp1(half_hf+1,j);
        elseif i>=height_i+half_hf+1&&j>=half_wf+1&&j<=width_i+half_wf
            image_temp1(i,j)=image_temp1(height_i+half_hf,j);
        elseif j<=half_wf&&i<=half_hf
            image_temp1(i,j)=image_temp1(half_hf+1,half_wf+1);
        elseif j<=half_wf&&i>=height_i+half_hf+1
            image_temp1(i,j)=image_temp1(height_i+half_hf,half_wf+1);
        elseif i<=half_hf&&j>=width_i+half_wf+1
            image_temp1(i,j)=image_temp1(half_hf+1,width_i+half_wf);
        elseif j>=width_i+half_wf+1&&i>=height_i+half_hf+1
            image_temp1(i,j)=image_temp1(height_i+half_hf,width_i+half_wf);
        end
    end
end

% filtering
```

```
image_temp2=image_temp1;
for i=1+half_hf:height_i+half_hf
      for j=1+half_wf:width_i+half_wf
            image_temp2(i,j)=sum(sum(filter.*image_temp1(i-half_hf:i+half_hf,j-
half_wf:j+half_wf)));
      end
end

% cut
image_fil=zeros(height_i,width_i);
for i=1:height_i
      for j=1:width_i
            image_fil(i,j)=image_temp2(i+half_hf,j+half_wf);
      end
end

end
```

(3) nms.m

```
function image=nms(image_ori,direction)
% width and height of original image
width_i=size(image_ori,2);
height_i=size(image_ori,1);

image=zeros(height_i,width_i);
for i=2:height_i-1
    for j=2:width_i-1
        if image_ori(i,j)~=0
            % horizontal, vertical and the two diagonals?
            % 90 degree
            if direction(i,j)<=-67.5||direction(i,j)>=67.5
                temp1=image_ori(i-1,j);
                temp2=image_ori(i+1,j);
            % -45 degree
            elseif direction(i,j)>-67.5&&direction(i,j)<-22.5
                temp1=image_ori(i-1,j-1);
                temp2=image_ori(i+1,j+1);
            % 0 degree
            elseif direction(i,j)>=-22.5&&direction(i,j)<=22.5
                temp1=image_ori(i,j-1);
                temp2=image_ori(i,j+1);
            % 45 degree
            elseif direction(i,j)>22.5&&direction(i,j)<67.5
                temp1=image_ori(i-1,j+1);
                temp2=image_ori(i+1,j-1);
            end

            if image_ori(i,j)>=temp1&&image_ori(i,j)>=temp2
                image(i,j)=image_ori(i,j);
            end
        end
    end
end

end
```