EEE 178: Homework 2

Curtis Muntz

February 13, 2014

- Problem 2
  - Part 10

    As you can see in the images attached on the following pages, the gaussian filter was slightly effective in removing the gaussian noise, and the median filter was very effective in removing the salt & pepper noise. Both filters were very ineffective at removing the other form of noise. See Table 1

    All of the functions that I wrote were effective in producing identical results as the built in functions, but ran exponentially slower than their counterparts.

    See the attached pages for the MATLAB code used to complete the homework and self-written functions.

Table 1: Filter Effectiveness On Noise Type

Filter Type	Noise Type	Results
Gaussian	Gaussian Noise	Sort of effective
Median	Salt & Pepper Noise	Very effective
Gaussian	Salt & Pepper Noise	Not effective
Median	Gaussian Noise	Not effective

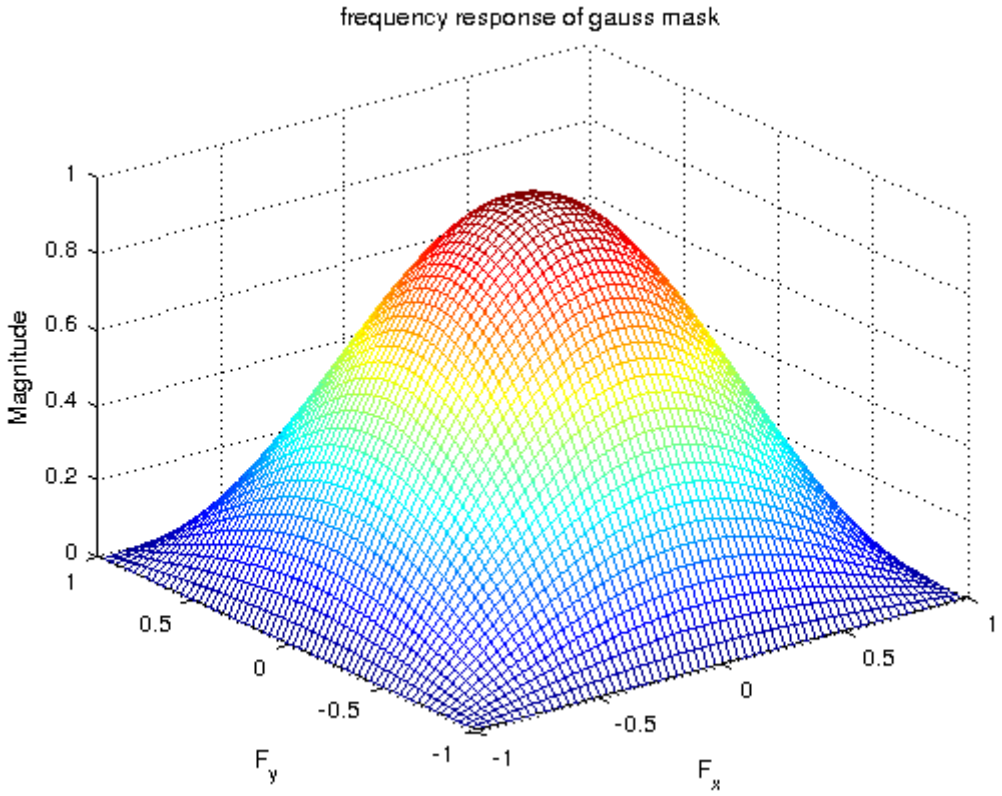
```
clear all;
clc;
clf
close all;

filetype='.jpg';
%-----------------------------------------------%
% problem 1                                     %
%-----------------------------------------------%

%--------%
% part 1 %
%--------%
%%construct 3x3 normalized gauss mask
%default response to mimic matlab's built in function
dimensions = 3;
sigma = 0.5;
h = gaussfilter(dimensions,sigma)

%--------%
% part 2 %
%--------%
%%built in matlab function, compare to my function
H = fspecial('gaussian',dimensions,sigma)
sprintf('comapring my gauss filter vs builtin: h-H:')
gauss_difference = H-h %proves my mask and built in masks are identical
                       %because it = 0

%--------%
% part 3 %
%--------%
%%construct a 300x300 gauss mask
h2=fspecial('gaussian',300,sigma);

%--------%
% part 4 %
%--------%
%%plot frequency respons of said mask
figure('name', 'frequency response of gaauss mask');
freqz2(h2)
title('frequency response of gauss mask');
%--------%
% part 5 %
%--------%
%%display magnitude of FFT of gauss mask with DC component in the center
h2 = fft2(h2);
h2=fftshift(h2);
%h2 = sqrt(shifted);
figure('name','shifted fourier');
imshow(h2);
title('shiftedfft');
```

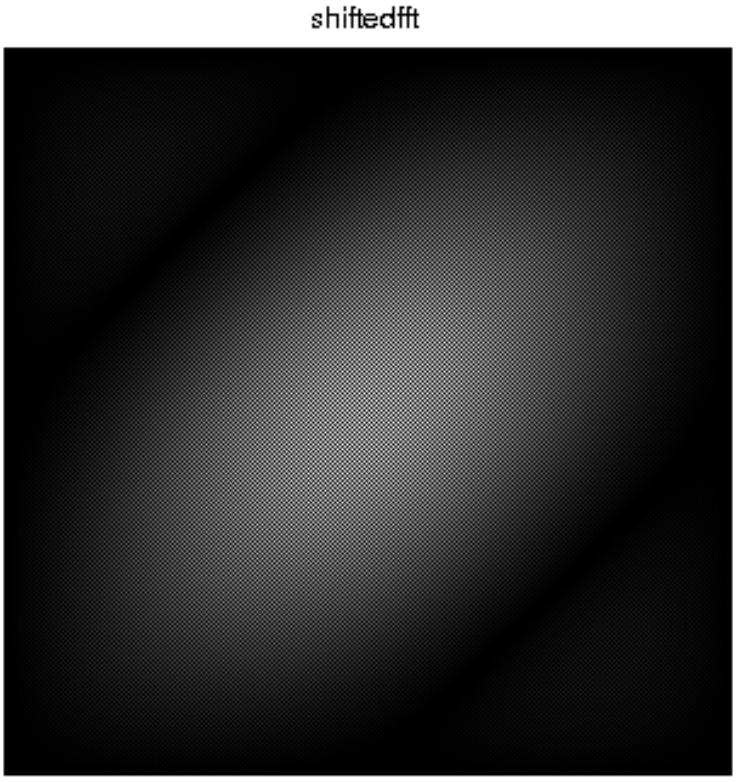
```
%-----------------------------------------------%
% problem 2                                      %
%-----------------------------------------------%

%--------%
% part 1 %
%--------%
%%read image (i develop on both windows and linux so this command
%%automatically determines what filepath to use)
image = './Carl_Friedrich_Gauss.jpg';
baseimage = imread(image);
I = imresize(baseimage, [400,400]);
I = rgb2gray(I);

figure('name', 'original image')
imshow(I);
title('original image');
%--------%
% part 2 %
%--------%
%%add gauss noise to image, call it image2
image2 = imnoise(I,'gaussian',0,.05);
figure('name', 'gauss noise added to img');
imshow(image2);
title('gauss noise added to img');

%--------%
% part 3 %
%--------%
%%add salt/pepper noise to image, call it image3
image3 = imnoise(I ,'salt & pepper',.05);
figure('name', 'salt & pepper added to img');
imshow(image3);
title('salt & pepper added to img');

%--------%
% part 4 %
%--------%
%%apply filtering in space domain (convolution)
%see convolution.m


%--------%
% part 5 %
%--------%
%%apply median filtering operation
%see medianfilter.m

%--------%
% part 6 %
%--------%
%%use convolution code to perform gauss filter on image2
FI = convolution(image2, h);
figure('name', 'gauss filtered image2');
```

```
imshow(FI);
title('gauss filtered image2');

% figure('name', 'gauss filter using builtin convolv');
% builtin = conv2(double(I),double(h));
% imshow(builtin);

%--------%
% part 7 %
%--------%
%%use convolution code to perform gauss filter on image3
G3 = convolution(image3, h);
figure('name', 'gauss filtered img3');
imshow(G3);
title('gauss filtered image3');

%--------%
% part 8 %
%--------%
%%use code to perform median filter on image2
med1= medianfilter(image2);
figure('name','median filtered image2');
imshow(med1);
title('median filtered image3');

%--------%
% part 9 %
%--------%
%%use code to perform median filter on image3
med2 = medianfilter(image3);
figure('name','median filtered image3');
imshow(med2);
title('median filtered image3');

%---------%
% part 10 %
%---------%
%%show and discuss results
```

```
h =

    0.0113    0.0838    0.0113
    0.0838    0.6193    0.0838
    0.0113    0.0838    0.0113


H =

    0.0113    0.0838    0.0113
    0.0838    0.6193    0.0838
    0.0113    0.0838    0.0113
```

```
ans =

comapring my gauss filter vs builtin: h-H:

gauss_difference =

     0     0     0
     0     0     0
     0     0     0

Warning: Displaying real part of complex input.
```




shiftedfft

original image

gauss noise added to img

salt & pepper added to img

gauss filtered image2

gauss filtered image3

median filtered image3

median filtered image3

Published with MATLAB® R2013b

```
function [h] = gaussfilter(dimensions,sigma)
 %center row, column is found by:
 %divide dimensions by 2, round up

 m=ceil(dimensions/2); %this gives center value

 %initialize 2D array
 h=zeros(dimensions,dimensions);
 %h(i,j) = e^(((i-m)^2+(j-m)^2)/(2sigma^2)))
 summed = 0;
 for (row=1:dimensions)
  for (col=1:dimensions)
   h(row,col) = exp(-(((col-m)^2+(row-m)^2)/(2*sigma^2)));
   summed = summed + h(row,col);
  end
    end
    %normalize!
 h = h/summed;
```

```
        Error using gaussfilter (line 5)
        Not enough input arguments.
```

Published with MATLAB® R2013b

```
function [filteredIMG] = convolution(image, mask)
 i = 1;
 j = 1;
 k = 1;
 l = 1;


 [maskM, maskN]=size(mask);
    [M,N] = size(image);

 if (mod(maskM,2) == 0)
  sprintf('error, i only support odd dimension masks currently')
  %easiest way to do this is to just add another row to make it odd,
        %although this might not be the 'best' option.
  return
    elseif (mod(maskN,2) == 0)
        sprintf('error, i only support odd dimension masks currently,')
  return
 end
 maskCenterRow=ceil(maskM/2);
 maskCenterCol=ceil(maskN/2);

 %initialize filtered image
 newIMG = zeros(size(image));
 filteredIMG = im2uint8(newIMG);

 %convolve!
    for i=1:M;
  for j=1:N;
   for k=1:maskM;
    for l=1:maskN;
                    if ((i+(k-maskCenterRow) > 0) &&...
                           (j+(l-maskCenterCol) > 0)) &&...
                           ((i+(k-maskCenterRow) <= M) &&...
                           (j+(l-maskCenterCol) <= N))
                       filteredIMG(i,j) =...
                           filteredIMG(i,j) + ...
                           (image(i+(k-maskCenterRow),...
                           j+(l-maskCenterCol))*mask(k,l));
                    end
                end
            end
        end
    end

        Error using convolution (line 8)
        Not enough input arguments.
```

Published with MATLAB® R2013b

```
function [MED] = medianfilter(image)
    [M,N] = size(image);
    center=2;
    z=1;
    values=zeros(1,9);
    tmp=zeros(size(image));
    MED=uint8(tmp);
    for i=1:M
     for j=1:N
      for k=1:3
       for l=1:3
        if (((i+(k-center)) > 0) && ((j+(l-center)) > 0) && ...
                              (i+(k-center) < M) && (j+(l-center) < N))
         values(z) = image((i+(k-center)),(j+(l-center)));
         z = z + 1;
        end
       end
      end
     MED(i,j) = median(values);
     clear values;
     z = 1;
     end
    end
```

```
    Error using medianfilter (line 2)
    Not enough input arguments.
```

Published with MATLAB® R2013b