Homework X Writeup

Instructions

- Describe any interesting decisions you made to write your algorithm.
- Show and discuss the results of your algorithm.
- Feel free to include code snippets, images, and equations.
- Use as many pages as you need, but err on the short side If you feel you only need to write a short amount to meet the brief, th
- **Please make this document anonymous.**

Files...

1. my_imfilter.m

   input : image, filter
   output : after filter image

   1) Pad the input image with zeros.
   2) Support grayscale and color images.
   3) Support arbitrary shaped odd-dimension filters
   (e.g., $7x9$ filters but not $4x5$ filters).
   4) Return an error message for even filters, as their output is undefined.
   5) Return an identical image with an identity filter.
   6) Return a filtered image which is the same resolution as the input image.
   Used in hw2_test_filtering and gen_hybrid_image.

2. hw2_test_filtering.m

   Nothing to modify. This is to verify that my_imfilter is fuctioning properly.

3. gen_hybrid_image.m

   input : image1, image2, cutoff_frequency
   output : three images [hybrid_image,low_frequencies,high_frequencies]
   Just use what filters to make high pass filter and low pass filter.
   Used in vis_hybrid_image and hw2.

4. vis_hybrid_image.m

   Nothing to modify. Adjust the scale and padding.

5. hw2.m

   Nothing to modify. Using gen_hybrid_image and vis_hybrid_image, apply high pass filter and low pass filter, make hybrid image, and store each image.

In code.

1. my_imfilter.m

```
function output = my_imfilter(image, filter)
\%%%%%%%%%%%%%%%%
\% Your code here

double_image = im2double(image);
[im,in,ik] = size(double_image);
[fm,fn] = size(filter);

\%if size of filter even, catch error.
if rem(fm,2)~=1
    msg = 'filter size error1.';
    error(msg);
elseif rem(fn,2)~=1
    msg = 'fliter size error2.';
    error(msg);
end

\%boundaries make zero.
cal_image = double_image;
lrzero = zeros(im,fix(fn/2),3);
cal_image = [lrzero cal_image lrzero];
udzero = zeros(fix(fm/2),in+fn-1,3);
cal_image = [udzero;cal_image;udzero];

\%calculate filter*image.
startm = fix(fm/2)+1;
endm = startm+im-1;
startn = fix(fn/2)+1;
endn = startn+in-1;
h = zeros(im,in,ik);
for i = startm:endm
    for j = startn:endn
        sum = zeros(1,ik);
        for a = 1:ik
            temp = sum(a);
            for k = -fix(fm/2):fix(fm/2)
                for l = -fix(fn/2):fix(fn/2)
                    increase = filter(k+fix(fm/2)+1,l+fix
                        (fn/2)+1).*cal_image(i-k,j-l,a);
                    sum(a) = temp + increase;
                    temp = sum(a);
```

```
                end
            end
            h(i-fix(fm/2),j-fix(fn/2),a) = sum(a);
        end
    end
end

output = h;
%%%%%%%%%%%%%%%%%
```

Initially, if the filter was even in size, the error was detected.

Second, since the data can be lost when applying filter, assuming the filter size is 2m+1, 2n+1, we made each m up and down, and each n left and right.

Finally, the equation for the resolution is as follows, so the process was applied to all parts of the image, not to the Boundaries.

convolution :

$$h[m,n] = \sum_{k,l} f[k,l]I[m-k,n-l]$$

2. hw2_test_filtering.m

Nothing to modify. So the code will be omitted.

3. gen_hybrid_image.m

```
function [hybrid_image,low_frequencies,high_frequencies]
   = gen_hybrid_image( image1, image2, cutoff_frequency )
\%%%%%%%%%%%%%%%%%%%%%%%%
\% Remove the high frequencies from image1 by blurring it
   . The amount of
\% blur that works best will vary with different image
   pairs
\%%%%%%%%%%%%%%%%%%%%%%%%
filter = fspecial('Gaussian', cutoff_frequency*4+1,
   cutoff_frequency);
image1 = im2double(image1);
low_frequencies = my_imfilter(image1, filter);

\%%%%%%%%%%%%%%%%%%%%%%%%
```

```
11 \% Remove the low frequencies from image2. The easiest
      way to do this is to
12 \% subtract a blurred version of image2 from the original
       version of image2.
13 \% This will give you an image centered at zero with
      negative values.
14 \%%%%%%%%%%%%%%%%%%%%%%%%
15 image2 = im2double(image2);
16 high_frequencies = image2 - my_imfilter(image2, filter);
17
18 \%%%%%%%%%%%%%%%%%%%%%%%%
19 \% Combine the high frequencies and low frequencies
20 \%%%%%%%%%%%%%%%%%%%%%%%%
21 hybrid_image = imfuse(low_frequencies,high_frequencies,'
      blend','Scaling','joint');
```

The low-pass filter was applied to the Gausian filter because it had to be put in blur, and the high-pass filter was made to subtract the low-pass filter from the original because only the high frequencies must remain.

4. vis_hybrid_image.m

Nothing to modify. So the code will be omitted.

5. hw2.m

Nothing to modify. So the code will be omitted.

Result...

1. my_imfilter.m

   Nothing output image.

2. hw2_test_filtering.m

[output]
(1) identity_image.jpg

```
identity_filter = [0 0 0; 0 1 0; 0 0 0];
identity_image = my_imfilter(test_image, identity_filter)
    ;
```

Figure 1: identity_image.

(2) blur_image.jpg

```
blur_filter = [1 1 1; 1 1 1; 1 1 1];
blur_filter = blur_filter / sum(sum(blur_filter));
blur_image = my_imfilter(test_image, blur_filter);
```

Figure 2: blur_image.

(3) large_blur_image.jpg

```
large_1d_blur_filter = fspecial('Gaussian', [25 1], 10);
large_blur_image = my_imfilter(test_image,
   large_1d_blur_filter);
large_blur_image = my_imfilter(large_blur_image,
   large_1d_blur_filter');
```

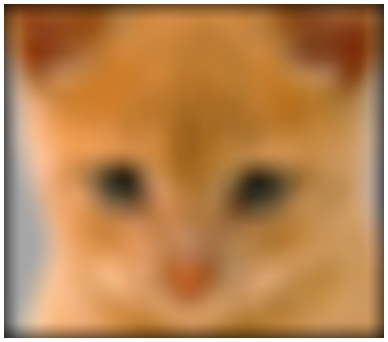

Figure 3: large_blur_image.

(4) sobel_image.jpg

```
sobel_filter = [-1 0 1; -2 0 2; -1 0 1];
sobel_image = my_imfilter(test_image, sobel_filter);
```

Figure 4: sobel_image.

(5) laplacian_image.jpg

```
laplacian_filter = [0 1 0; 1 -4 1; 0 1 0];
laplacian_image = my_imfilter(test_image,
   laplacian_filter);
```

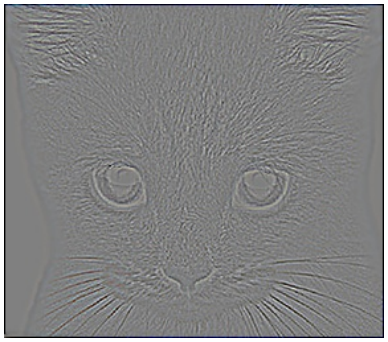

Figure 5: sobel_image.

(6) high_pass_image.jpg

```
high_pass_image = test_image - blur_image;
```

Figure 6: high_pass_image.

3. gen_hybrid_image

Nothing output image.

4. vis_hybrid_image

Nothing output image.

5. hw2

low_frequencies, high_frequencies, and hybrid_image is made through gen_hybrid_image, and hybrid_image_scales through vis_hybrid_image.

[output]
(1) low_frequencies.jpg

```
filter = fspecial('Gaussian', cutoff_frequency*4+1,
   cutoff_frequency);
low_frequencies = my_imfilter(image1, filter);
```

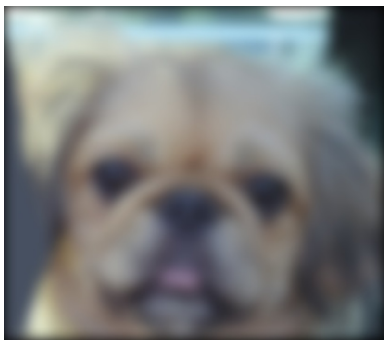

Figure 7: low_frequencies.

(2) high_frequencies.jpg

```
high_frequencies = image2 - my_imfilter(image2, filter);
```

Figure 8: high_frequencies.

(3) hybrid_image.jpg

```
hybrid_image = imfuse(low_frequencies,high_frequencies,'
   blend','Scaling','joint');
```

Figure 9: hybrid_image.

(4) hybrid_image_scales.jpg

```
scales = 5;
scale_factor = 0.5;
padding = 5;

original_height = size(hybrid_image,1);
num_colors = size(hybrid_image,3);
output = hybrid_image;
cur_image = hybrid_image;

for i = 2:scales
    output = cat(2, output, ones(original_height, padding
       , num_colors));
    cur_image = imresize(cur_image, scale_factor, '
       bilinear');
    tmp = cat(1,ones(original_height - size(cur_image,1),
        size(cur_image,2), num_colors), cur_image);
    output = cat(2, output, tmp);
end
```

Figure 10: hybrid_image_scales.