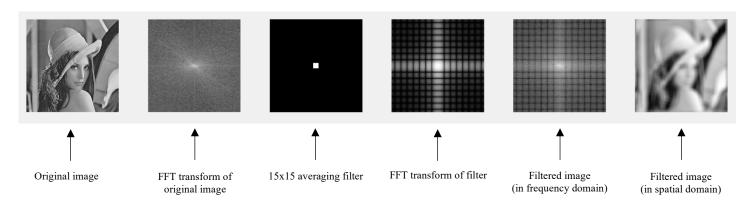
Haruka Kido EE456: Digital Image Processing Assignment 3

- 1. Filters applied to "lena_gray.jpg" image in the **frequency domain**:
 - a. 15x15 Average filter:
 - Source code:

```
clear;
       A = imread('lena_gray.jpg');
2 -
3 -
       subplot(1, 6, 1);
       imshow(A); %show original image
4 -
5
6 -
       [r, c] = size(A);
       %fft of original image
       A_fft = fftshift(fft2(A));
8 -
       subplot(1, 6, 2);
       imshow(log(1 + abs(A_fft)), []); %show fft transform of original image
10 -
11
       %15x15 average filter to filter image in frequency domain
12
13 -
       sizeofavgmask = 15
       half_sizeofavgmask = floor(sizeofavgmask/2);
15
       %avgfilterA = fspecial('average', sizeofavgmask);
       %B = imfilter(A, avgfilterA);
16
       mask = zeros(r, c);
17 -
       %dimensions of white-centered box in filter set to 1
18
       mask(r/2 - half\_sizeofavgmask: r/2 + half\_sizeofavgmask, c/2 - half\_sizeofavgmask: c/2 + half\_sizeofavgmask) = 1;
19 -
20 -
       subplot(1, 6, 3);
21 -
       imshow(mask); %show filter
22
       %fft of filter
23
24 -
       fft_mask = fftshift(fft2(mask));
       subplot(1, 6, 4);
25
26
       imshow(log(1 + abs(fft_mask)), []); %show fft transform of filter
27
28
       %application of filter to image
29
       %multiplies transformed image and transformed filter element by element through .* operator
30
       filtered_img_freq = A_fft .* fft_mask;
31 -
       subplot(1, 6, 5);
32
       imshow(log(1 + abs(filtered_img_freq)), []); %show filtered image in frequency domain
34
35
       %puts filtered image in freq domain to filtered image in spatial domain
       filtered_img_spatial = ifft2(filtered_img_freq);
37 -
       subplot(1, 6, 6);
       imshow(log(1 + abs(fftshift(filtered_img_spatial))), []); %show filtered image in spatial domain
38 -
```

- Original image and its FFT transform
- o Filter and its FFT transform
- o Filtered image in frequency domain
- o Filtered image in spatial domain

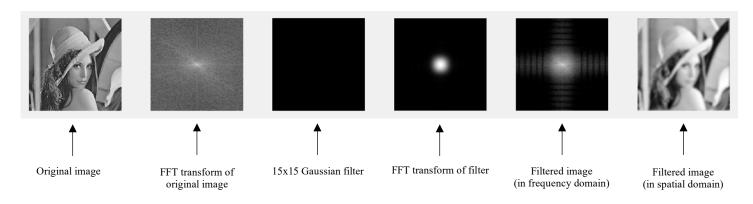


b. 15x15 Gaussian filter:

Source code:

```
clear;
       A = imread('lena_gray.jpg');
 2 -
 3 -
       subplot(1, 6, 1);
imshow(A); %show original image
 4 -
 5
 6 -
       [r, c] = size(A);
       %fft of original image
 7
 8 -
       A_fft = fftshift(fft2(A));
 9 -
       subplot(1, 6, 2);
10 -
       imshow(log(1 + abs(A_fft)), []); %show fft transform of original image
11
       %15x15 gaussian filter to filter image in frequency domain
12
13 -
       sizegaussfilt = 15
14 -
       half_sizegaussfilt = floor(sizegaussfilt/2);
15 -
16 -
       gaussfilt = fspecial("gaussian", sizegaussfilt, sigma);
17 -
       mask = zeros(r, c);
18
       %dimensions of white-centered box in filter set to 1 \,
       mask(r/2 - half_sizegaussfilt:r/2 + half_sizegaussfilt, c/2 - half_sizegaussfilt:c/2 + half_sizegaussfilt) = gaussfilt;
19 -
20 -
       subplot(1, 6, 3);
21 -
       imshow(mask); %show filter
22
23
       %fft of filter
24 -
       fft_mask = fftshift(fft2(mask));
25 -
       subplot(1, 6, 4);
       imshow(log(1 + abs(fft_mask)), []); %show fft transform of filter
26 -
27
28
       %application of filter to image
       \%multiplies transformed image and transformed filter element by element through .* operator
29
30 -
       filtered_img_freq = A_fft .* fft_mask;
31 -
       subplot(1, 6, 5);
32 -
       imshow(log(1 + abs(filtered_img_freq)), []); %show filtered image in frequency domain
33
34
35
       %puts filtered image in freq domain to filtered image in spatial domain
36 -
       filtered_img_spatial = ifft2(filtered_img_freq);
37 -
       subplot(1, 6, 6);
        imshow(log(1 + abs(fftshift(filtered_img_spatial))), []); %show filtered image in spatial domain
38 -
```

- Original image and its FFT transform
- o Filter and its FFT transform
- o Filtered image in frequency domain
- o Filtered image in spatial domain

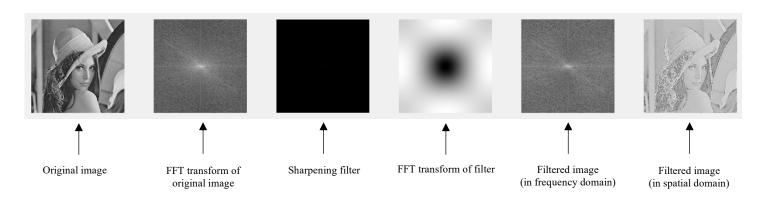


c. Any Sharpening filter:

o Source code:

```
1 -
       clear:
 2 -
       A = imread('lena_gray.jpg');
3 -
       subplot(1, 6, 1);
 4 -
       imshow(A); %show original image
 5
 6
       [r, c] = size(A);
       %fft of original image
 7
 8 -
       A_fft = fftshift(fft2(A));
 9 -
       subplot(1, 6, 2);
10
       imshow(log(1 + abs(A_fft)), []); %show fft transform of original image
11
12
       %15x15 gaussian filter to filter image in frequency domain
13 -
       sizefilt 💂 3
14 -
       half_sizefilt = floor(sizefilt/2);
15
16
       %high-boost filtering with constant > 1 as constant = 1.1
17 -
       SHARP = [-1.1 -1.1 -1.1;
18
                -1.1 11 -1.1;
19
                -1.1 -1.1 -1.1];
       %B = conv2(A, SHARP, 'valid'); %convolution
20
21
       C = uint8(B);
22
23
       mask = zeros(r, c);
24
       %dimensions of white-centered box in filter set to 1
       mask(r/2 - half\_sizefilt:r/2 + half\_sizefilt, c/2 - half\_sizefilt:c/2 + half\_sizefilt) = SHARP;
25 -
26 -
       subplot(1, 6, 3);
27 -
       imshow(mask); %show filter
28
29
       %fft of filter
30 -
       fft_mask = fftshift(fft2(mask));
31 -
       subplot(1, 6, 4);
32 -
       imshow(log(1 + abs(fft_mask)), []); %show fft transform of filter
33
34
       %application of filter to image
       %multiplies transformed image and transformed filter element by element through .* operator
35
36 -
       filtered_img_freq = A_fft .* fft_mask;
37 -
       subplot(1, 6, 5);
38 -
       imshow(log(1 + abs(filtered_img_freq)), []); %show filtered image in frequency domain
39
40
41
       %puts filtered image in freq domain to filtered image in spatial domain
42 -
       filtered_img_spatial = ifft2(filtered_img_freq);
43 -
       subplot(1, 6, 6);
       imshow(log(1 + abs(fftshift(filtered_img_spatial))), []); %show filtered image in spatial domain
```

- o Original image and its FFT transform
- o Filter and its FFT transform
- o Filtered image in frequency domain
- o Filtered image in spatial domain



- 2. Procedure in MATLAB to reduce the periodic noise on the background of "periodicNoiseCar.png" image:
 - Source code:

```
clear;
       A = imread('periodicNoisecar.png');
 3
4 -
5 -
       r = size(A, 1);
       c = size(A, 2);
 6
       subplot(1, 4, 1);
       imshow(A); %show original image
 8
 9
       %fft of original image
10 -
       A_{fft} = log(1 + abs(fftshift(fft2(A))));
11 -
       subplot(1, 4, 2);
       imshow(A_fft, []); %show fft transform of original image
12 -
13
       %filter that gives 0 outputs in locations of noise frequency components
14
15 -
       f = ones(r, c);
16 -
       f(62:66, 68:72) = 0;
17 -
       f(104:108, 68:72) = 0;
18 -
       f(186:190, 70:74) = 0;
19 -
       f(228:232, 72:76) = 0;
       f(60:64, 128:132) = 0;
20 -
21 -
       f(100:104, 128:132) = 0;
f(180:184, 128:132) = 0;
22 -
23 -
       f(222:226, 130:134) = 0;
24
25
       %application of filter to image
26
       %multiplies transformed image and transformed filter element by element through .* operator
27 -
       subplot(1, 4, 3);
28 -
       imshow(log(1 + abs(fftshift(fft2(A)) .* f)), []); %show filtered image in frequency domain
29
30
       %inverse fft
       %puts filtered image in freq domain to filtered image in spatial domain
31
       subplot(1, 4, 4);
32 -
        imshow(log(1 + abs(ifft2(fftshift(fft2(A)) .*f))), []); %show filtered image in spatial domain
33 -
```

- Original image and its FFT transform
- o Filtered image in frequency domain
- o Filtered image in spatial domain

