Image Processing
lab 2

Klaas Kliffen   Jan Kramer

December 3, 2015

Exercise 1 − Fourier spectrum

a. The functions build in functions fft2 and fftshift are used to create the fourier spectrum
image centered around the DC component of the fourier transform. Some extra code is
added for calculating the average of the image, which will be explained in more detail in
part c of this exercise. To be able to view the spectrum, the values need to be scaled. So
the log is taken of each value to increase the constrast.

```
1  % Read the image
2  x = imread('../images/characters.tif');
3  % Get image width and heigt
4  [width, height] = size(x);
5  % Perform the Fourier transform
6  spectrum = fftshift(fft2(x));
7  % Calculate the avarge of the image
8  % It can be found by taking the dc component (center of the image)
9  % And dividing it by the number of pixels
10 avg_fourier = abs(spectrum(width/2+1,height/2+1))/(width*height)
11 avg_mean = mean(mean(x))
12
13 % calculate the magnitude
14 spectrum = abs(spectrum);
15
16 % take the log value for better scaling in octave
17 logspectrum = log2(spectrum);
18 % Take note of max and min of the spectrum for image scaling
19 maxs = max(max(logspectrum));
20 mins = min(min(logspectrum));
21 % Scale the image for output to file
22 spectrumimg = uint8(floor((logspectrum - mins) / (maxs-mins) * 256));
23 imwrite(spectrumimg, 'spectrum.png');
24 % Show the log image as a figure
25 figure, imshow(logspectrum,[]), colormap gray
```
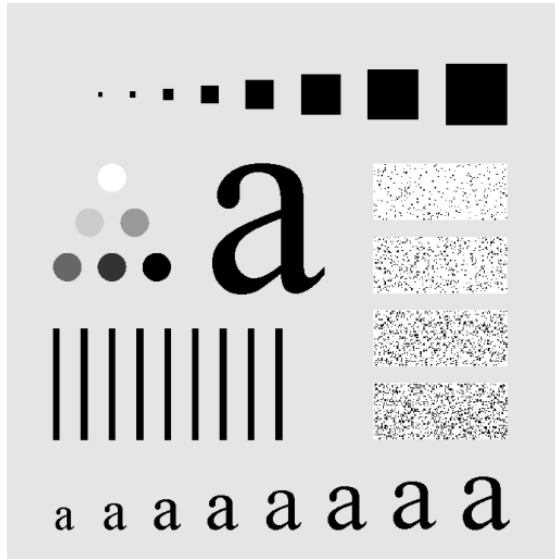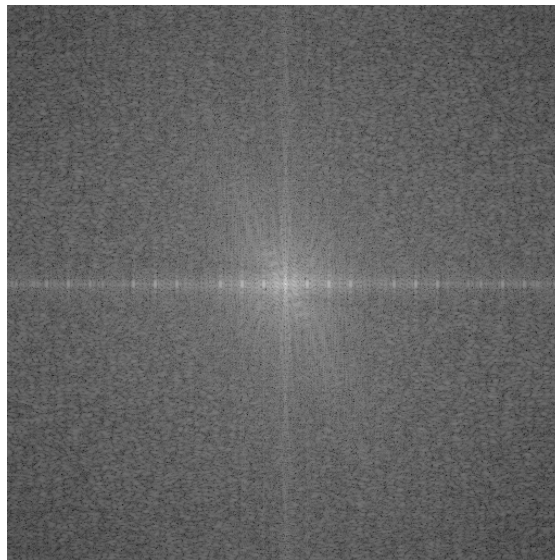
Figure 1: Original image

Figure 2: Fourier spectrum of figure 1

b.

c. The maginitude of the center of the image is the DC component. For images, this is the sum of the grey levels of all pixels. Dividing this by the number of pixels gives the average grey level. In this case: 207.31 for figure 1. This value is equal to the result achieved by using `mean` two times.

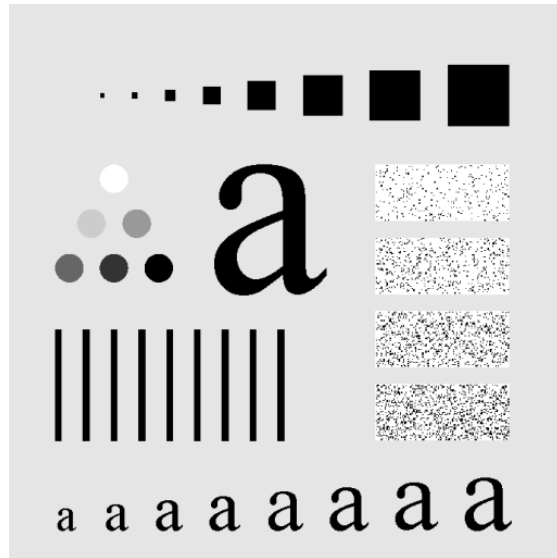# Exercise 2 – Highpass filtering in the frequency domain

a.

b.

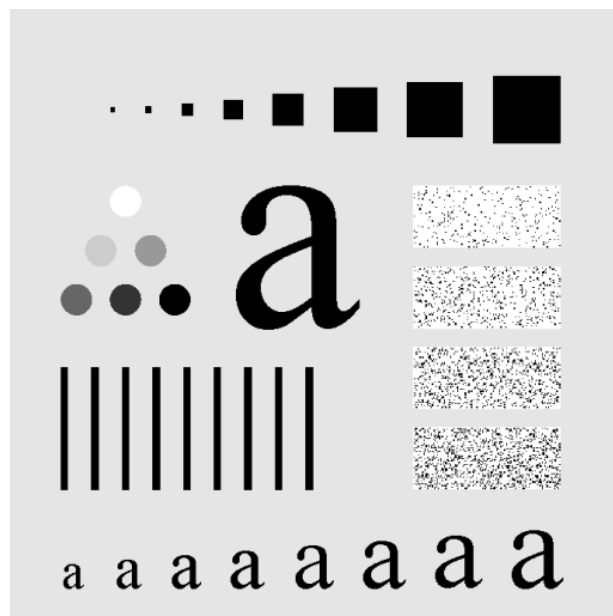Figure 3: Original image

Figure 4: Gausian filtering on figure 3

c.

Exercise 3 – Median filtering

a. The `IPmedian` function takes the distorted image and a value k as its input parameters. The for each pixel in the image, a window is created with width and height of $2k + 2$. When encountering a boundary, the windowsize is decreased to fit the image. A submatrix is then used to represent all the pixels in the window. From this submatrix, the median is calculated and used for the output image.

```matlab
1  % Median filter width a 2k+1 x 2k+1 window
2  function [out] = IPmedian(img,k)
3  % get the image size
4  [width, height] = size(img)
5  %create the output image
6  dest = uint8(zeros(size(img)));
7
8  % loop over all pixels
9  for x = 1 : width
10   for y = 1 : height
11     % determine the edge of the window
12     % the size of the window shrinks at the boundaries of the image
13     startx = max(x-k,1);
14     starty = max(y-k,1);
15     endx = min(x+k,width);
16     endy = min(y+k,height);
17     % get the submatrix
18     submat = img(startx:endx,starty:endy);
19     [w,h] = size(submat);
20     % convert it to a vector to be able to calculute the median
21     submat = reshape(submat, 1, w*h);
22     % take the median and store it
23     dest(x,y) = median(submat);
24   end
25  end
26  % output the image
27  out = dest;
```
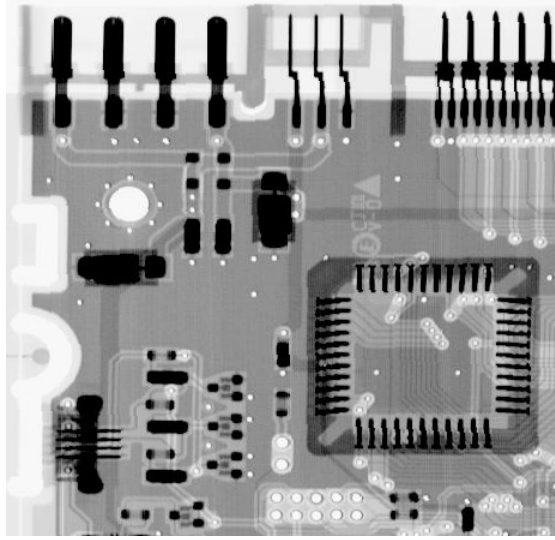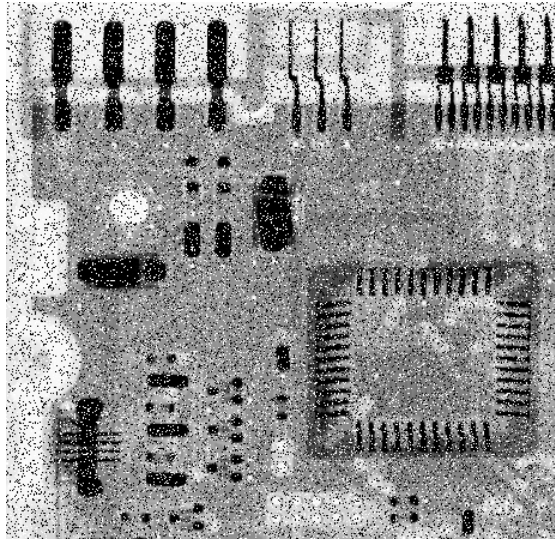
Figure 5: Original image of the circuitboard

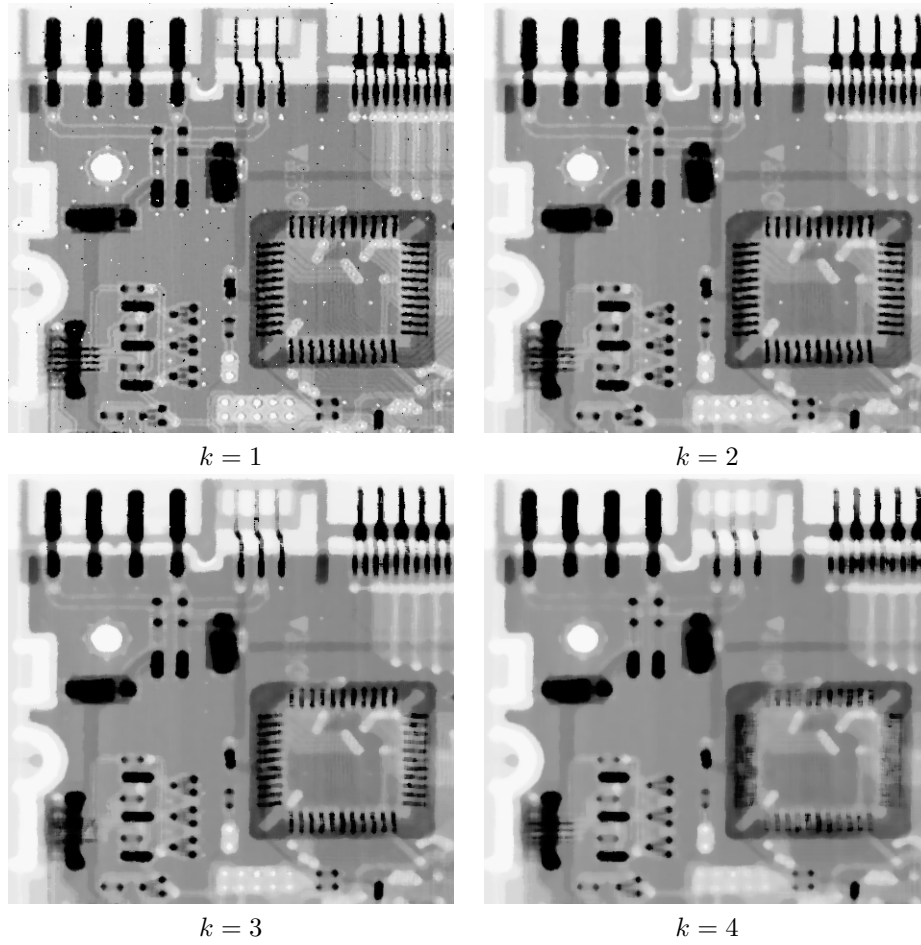Figure 6: Figure 5 with salt & pepper noise with $Pa = Pb = 0.2$

b.

$k = 1$

$k = 2$

$k = 3$

$k = 4$

Figure 7: Using median filtering with different window values k on figure 6

c. In the image created by filtering with $k = 1$, some noise is still present. This is equal to the image 5.10b in the book. Using $k = 2$ removes all noise particles. Increasing k further reduces the sharpness of the image as can be seen in the two lower images in 7

Task distribution

ex1	design	implementation	answers questions	writing report
Klaas	50%	100%	50%	50%
Jan	50%	0%	50%	50%

ex2	design	implementation	answers questions	writing report
Klaas	50%	0%	50%	50%
Jan	50%	100%	50%	50%

ex3	design	implementation	answers questions	writing report
Klaas	50%	100%	50%	50%
Jan	50%	0%	50%	50%