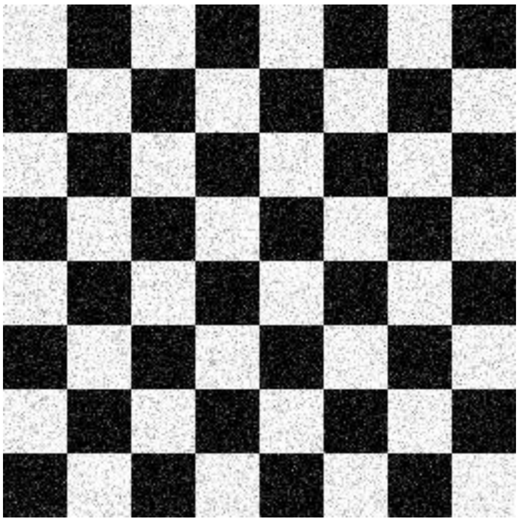
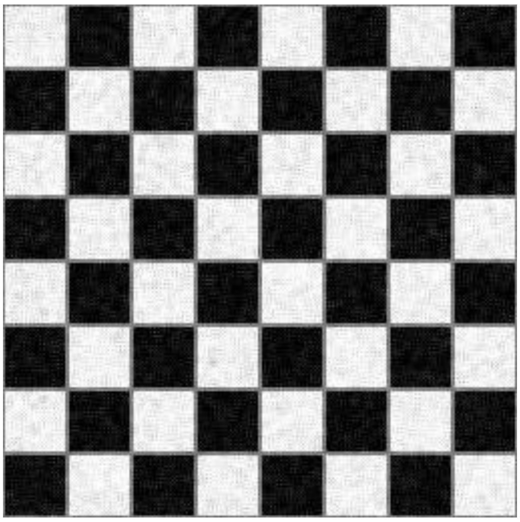
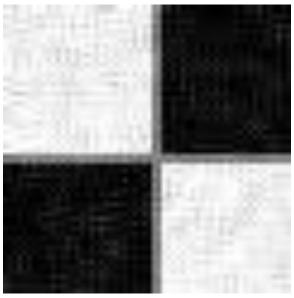
Chessboard and 1/2[1 0 1] filter



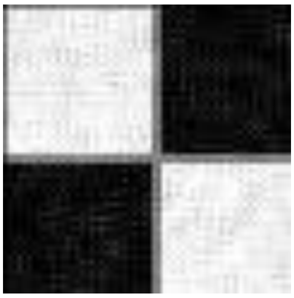
Chessboard with gaussian noise, v=0.02



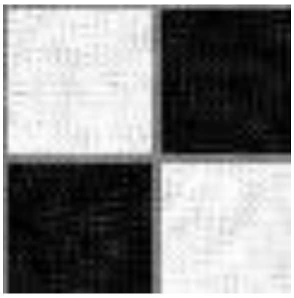
Chessboard with filter 0.5[1 0 1], periodic pad



Zoomed picture corner edges with extrapolation padding, picture edges have same color as pixels around them since extrapolation adds pixels with similar color and the filter has similar pixels when outside the border



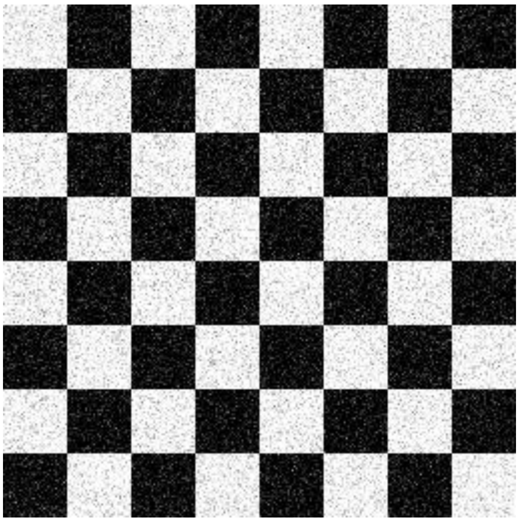
Zoomed picture corner edges with zero padding, picture edges are gray next to white since gray is average between white and black, picture edges next to black remain black



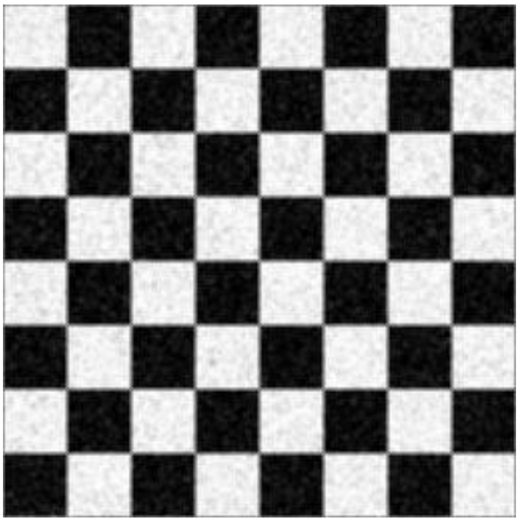
Zoomed picture corner edges with periodic padding, picture edges are gray due to periodic jumps from black to white at padded pixels

For gaussian noise we see that the [1 0 1] averaging filter does reduce the noise and less color jumps are seen, and they are smaller, however, the filter also blurres the image causing it to be fuzzy and blurred next to the edges, also the filtering causes a new gray edge between black and white areas that did not exist earlier due to average of white and black, the padding method has affect on picture edges, when periodic a gray edge is created, when zero padding, black edge remains black, but white is changed to gray, with extrapolation the picture edges after filtering remains similar to edges before filtering

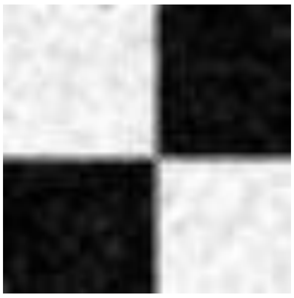
Chessboard and 1/3[1 1 1] filter

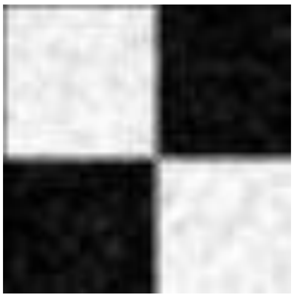


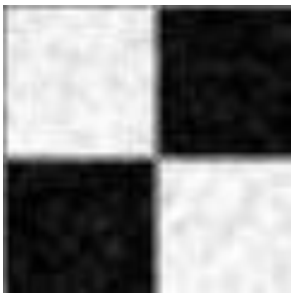
Chessboard with gaussian noise, v=0.02



Chessboard with filter 1/3[1 1 1], periodic pad





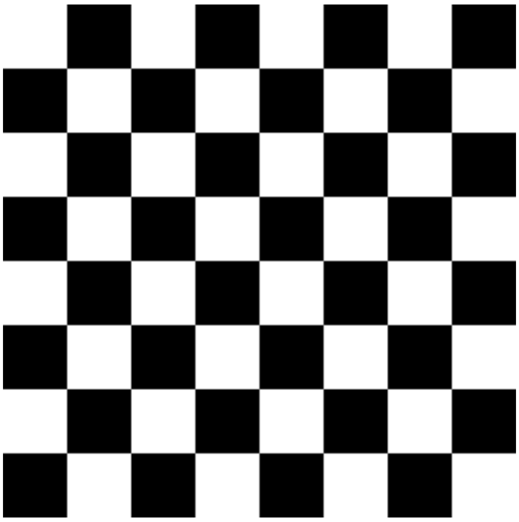


Zoomed picture corner edges with extrapolation padding, picture edges have same color as pixels around them since extrapolation adds pixels with similar color and the filter has similar pixels when outside the border

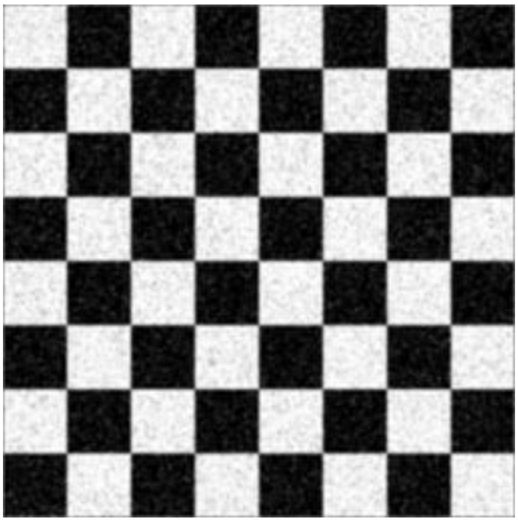
Zoomed picture corner edges with zero padding, picture edges are gray next to white since gray is average between white and black, picture edges next to black remain black, the gray is brighter then in [1 0 1] filter due to only 1/3 weight of padded pixel

Zoomed picture corner edges with periodic padding, picture edges are gray due to periodic jumps from black to white at padded pixels, however the gray level is closer to image color then in [1 0 1] since the affect of padding is only 1/3

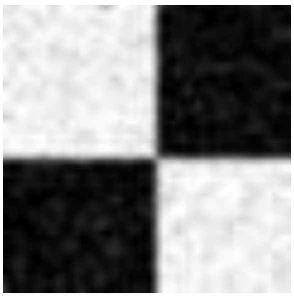
For gaussian noise we see that the [1 1 1] averaging filter does reduce the noise and less color jumps are seen, and they are smaller, however, the filter also blurres the image causing it to be fuzzy and blurred next to the edges, also the filtering causes a new gray edge between black and white areas that did not exist earlier due to average of white and black, the gray edges are closer to real value since the effect of wrong pixel is now 1/3 and not ½ as with the previous filter, the padding method has affect on picture edges, when periodic a gray edge is created, when zero padding, black edge remains black, but white is changed to gray, with extrapolation the picture edges after filtering remains similar to edges before filtering



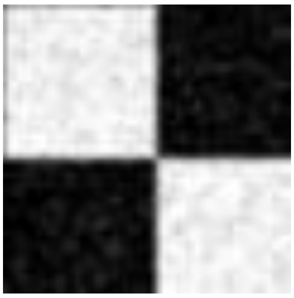
Original Chessboard



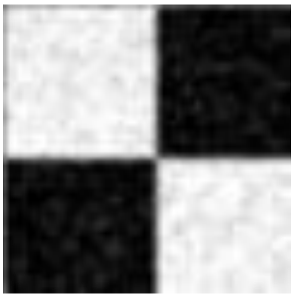
Chessboard after noise with filter 1/4[1 2 1], periodic pad



Zoomed picture corner edges with extrapolation padding, picture edges have same color as pixels around them since extrapolation adds pixels with similar color and the filter has similar pixels when outside the border



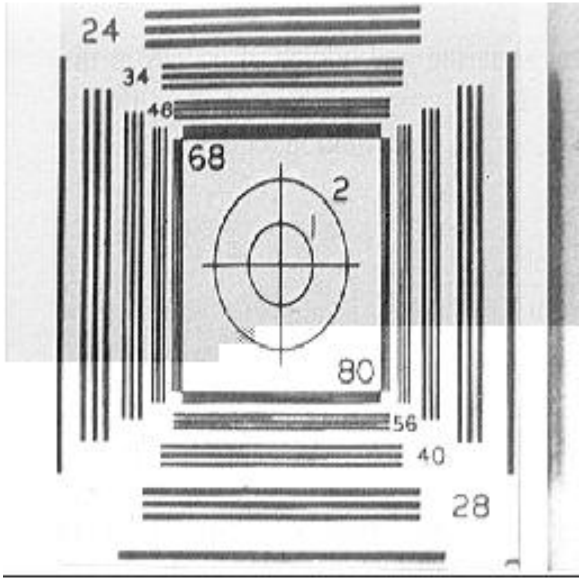
Zoomed picture corner edges with zero padding, picture edges are gray next to white since gray is average between white and black, picture edges next to black remain black, the gray is brighter then in [1 1 1] filter due to only 1/4 weight of padded pixel



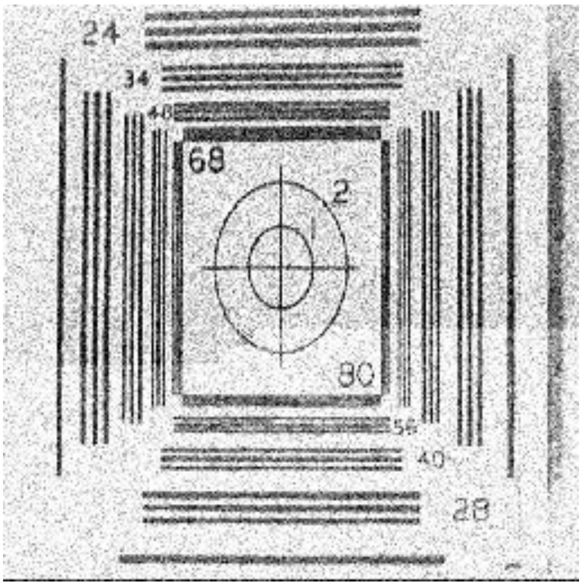
Zoomed picture corner edges with periodic padding, picture edges are gray due to periodic jumps from black to white at padded pixels, however the gray level is closer to image color then in [1 1 1] since the affect of padding is only 1/4

For gaussian noise we see that the [1 2 1] averaging filter does reduce the noise and less color jumps are seen, and they are smaller, however, the filter also blurres the image causing it to be fuzzy and blurred next to the edges, white is light gray and black is dark gray, also the filtering causes a new gray edge between black and white areas that did not exist earlier due to average of white and black, the gray edges are closer to real value since the effect of wrong pixel is now 1/4 and not 1/3 as with the previous filter, the padding method has affect on picture edges, when periodic a gray edge is created, when zero padding, black edge remains black, but white is changed to gray, with extrapolation the picture edges after filtering remains similar to edges before filtering

The pattern image



Original pattern image

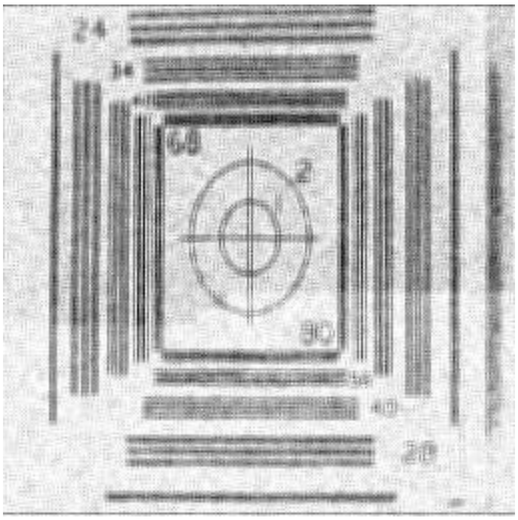


Pattern image with gaussian noise, m=0, v=0.02

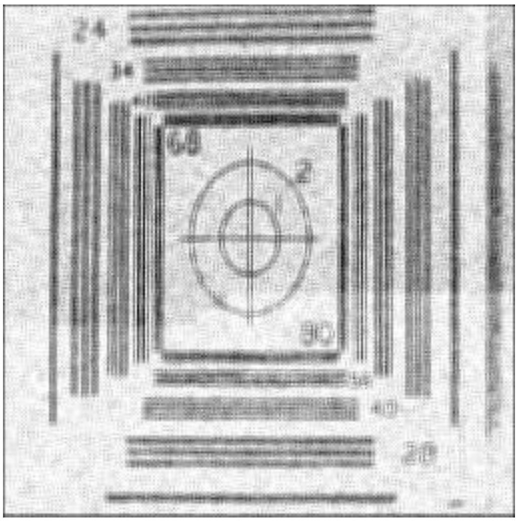
Commands:

I = rgb2gray(imread('pattern','jpg'));
N = imnoise(I,'gaussian',0,0.02);

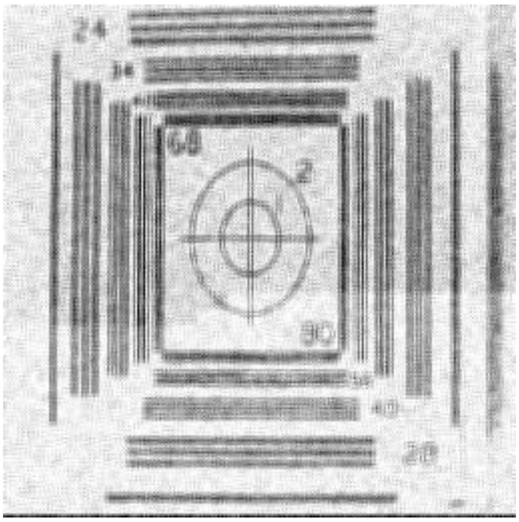
Filtering with ½*[1 0 1]



with periodically padding



with zero padding



With extrapolation padding

The filtering causes picture to be much more blurred compared to original, this is very prominent when looking at the numbers, with noise the numbers can still be read, but after filtering, the numbers are much less clear, especially in the bottom left area, in addition the filtering causes separate lines to look is if they where connected into one thicker line

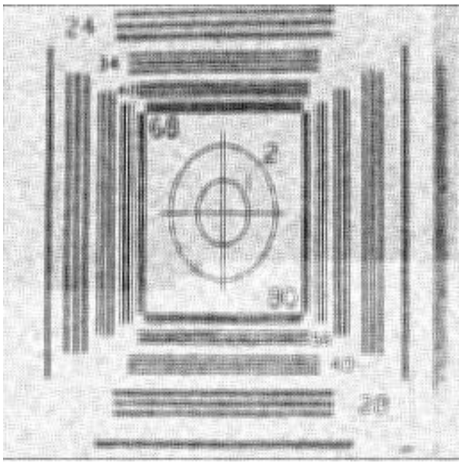
About the padding,

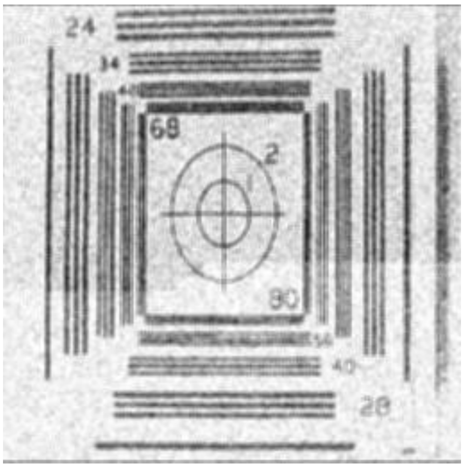
the zero padding creates a gray frame in the picture, this is as result of averaging with black padded pixels

the periodical padding causes the black line in the bottom to move to the top while the bottom becomes whiter

with extrapolation padding, the frame pixels remains similar to frame of original picture

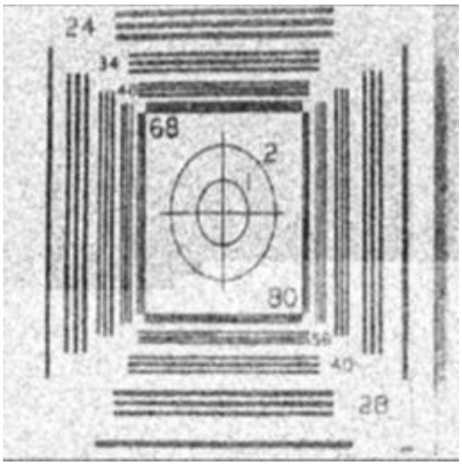
Different filters on pattern (periodical pad)





with filter [1 0 1]/2

with filter [1 1 1]/3



with filter [1 2 1]/4

comparing with same padding since the different paddings effects with the different filters is almost the same as with the [1 0 1] filter and affects only the frame

we can see that the [1 0 1] filter causes the most blurred image and is the most different from the original, this can be viewed by trying to read the numbers and by looking at close lines that looks like one thick line, one reason for that is that when multiplying the fourier coefficients, high frequency values remains high (cos(pi))=1 for frequency N/2 and noise has high frequency

the [1 1 1] filter is a bit better and we can see that the numbers are a bit more clear while less lines are combined to a thicker line (the lines next to the 20), if we multiplied the fourier coefficients, frequency N/2 would multiply by $\frac{1+2\cos(\pi)}{3} = -\frac{1}{3}$ so high frequency of noise becomes smaller

the [1 2 1] filter seems better then the [1 1 1] filter, the 56 number is a bit clearer and the lines next to 40 are more separated, also the background seems smoother, if we look at fourier multiplications for this filter we see that high frequency of N/2 is multiplied by $\frac{1+\cos(\pi)}{2} = 0$ and so the high frequency noise has less effect

multiply fourier coefficients by $\sigma_n = \frac{1+\cos(2\pi \cdot n)}{N}$ pictures

original

with noise

multiplied fourier

we can see her as well as in the [1 0 1] filter case (they are equivalent with periodic padding) that the new coefficients causes the image to be blurred but the noise seems to have less effect, as seen earlier, for high N/2 frequency the multiplication only change sign and so noise which usually has high frequency is not totally eliminated as may be with other multiplication options

original

noisy

multiplied fourier

ideal filter

original

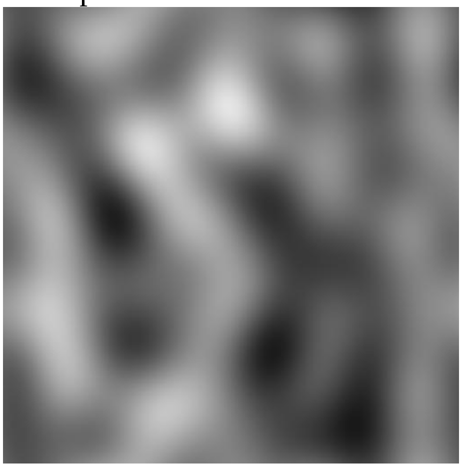
noisy

multiplied d=100

multiplied d=50

multiplied d=20

multiplied d=5

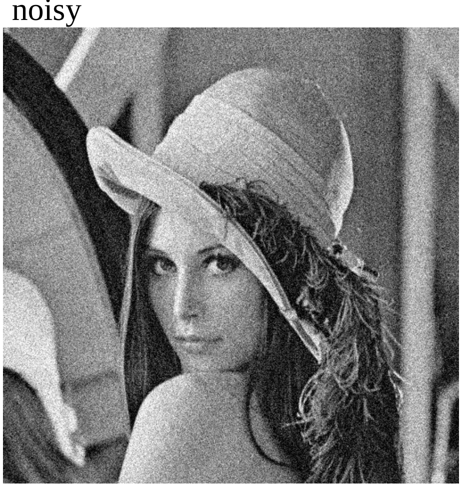


we can see that the larger the passing radius, the picture is more understandable and recognizable, however the bigger the radius, the noise is less removed and so with D=100, the picture is recognizable and close to real image but the noise has almost the same effect as without filtering and some small details disappear, the smaller D is, the noise has less effect but the picture is blurred and it seems that large object has sort of a shadow of themselves around them and are tripled (like the hat), under a certain level, as seen with D=5, the image almost unrecognizable

fourier multiply with $\sigma_n = e^{-\frac{\alpha n}{N}}$

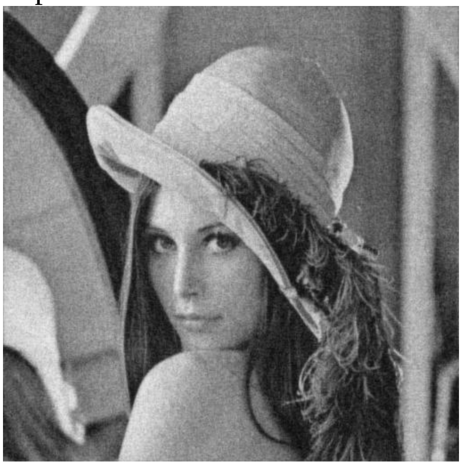
original

noisy



alpha=0.5

alpha=3



alpha=8

alpha=15

it seems that for small alpha, the image remains almost the same as the noisy image so the multiplication has no effect, the more we increase alpha we see less noise and less jumps in pixel colors, however the bigger alpha we also see less details and less edges information, for alpha=15 we almost see no noise and the picture looks smooth, however it looks too smooth and we also loose the picture details, for instance the hat looks like something in a hat shape but without the small shadow lines

butterworth $\sigma_{nx,ny} = \frac{1}{1+\left(\frac{D(nx,ny)}{D0}\right)^{2n}}$

D0=15, n=1

D0=15, n=5

D0=15, n=10

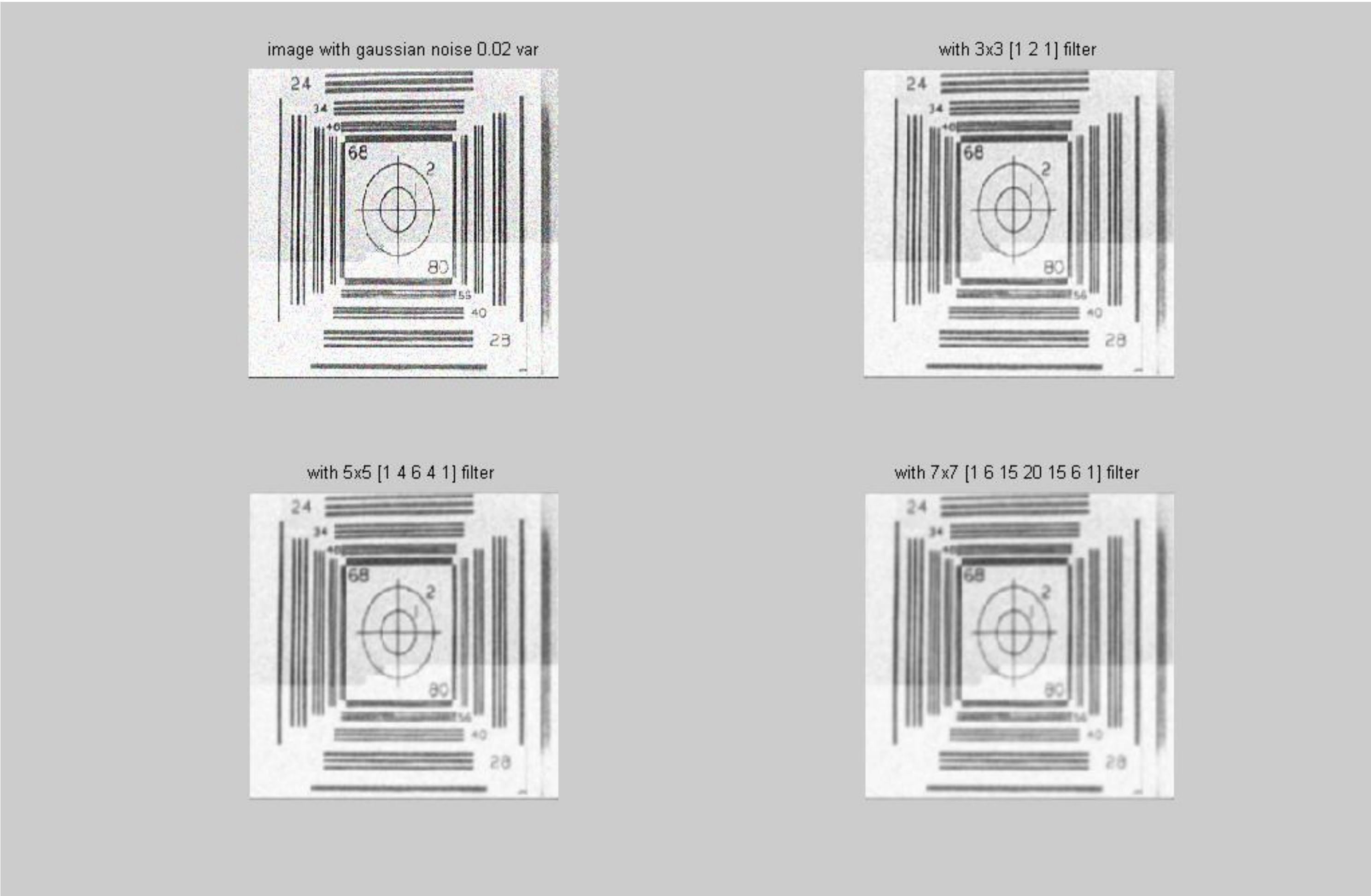
D0=70, n=1

D0=70, n=5

D0=70, n=10

we can see that the smaller D0 is, n has larger effect, and for D0=15, large n causes effect similar to that of ideal filter when the image is very blurred and large shapes appear to have a shadow, with small D0 and small n the effect is like with the exponent with large alpha, when D0 is larger, n has less effect, with smaller n the image is close to the noisy image and larger n causes it to be smoother and blurred

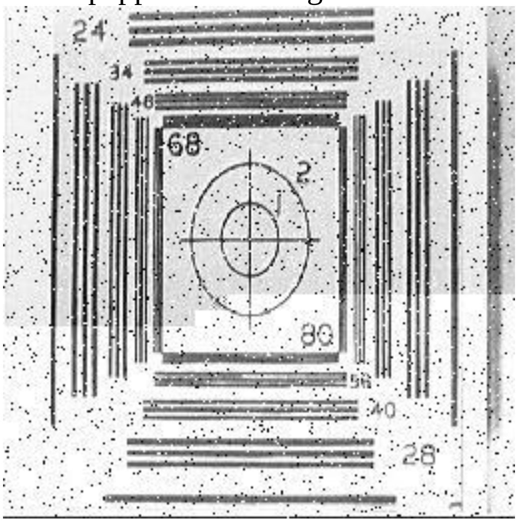
averaging with different filter size, the [1 2 1]/4 extention filters, periodic filling



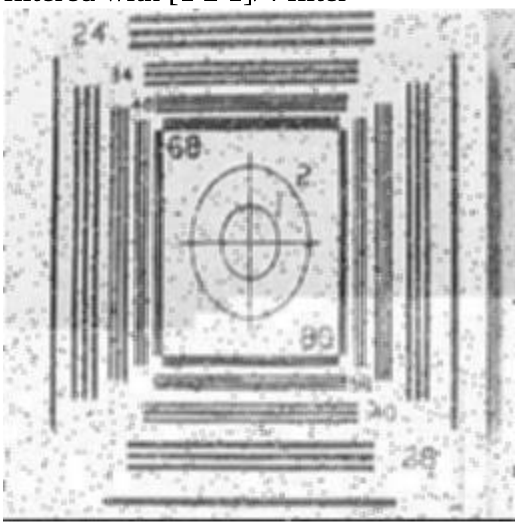
the bigger the averaging window, the more noise is reduced, however the picture becomes flatter and small differences disappear, the 7x7 window causes the picture to be very blurred and edges are not sharp, the edges are smoothened and therefore close lines look like one bigger line, in the 3x3 this affect also happens but it is less intense, when looking at the numbers the effect is also shown and the bigger the averaging window the harder it is to read the numbers, however the bigger the window, the surfaces are smoother and so the noise in the 7x7 window is almost unseen

averaging filter over salt & pepper noise

salt & pepper noise image



filtered with [1 2 1]/4 filter



in the case of salt & pepper noise we can see that the filtering, besides the usual blurring of edges, also smoothens the noise and therefore instead of having single pixels of noise we get small spots of noise that are larger then a pixel size, this happens since after averaging, every noise pixel has effect on all pixels in the window and so it is spread to a larger portion of the image

averaging color images - RGB

the color images are available on the disk with colors,

image with noise
flower_noise.jpg

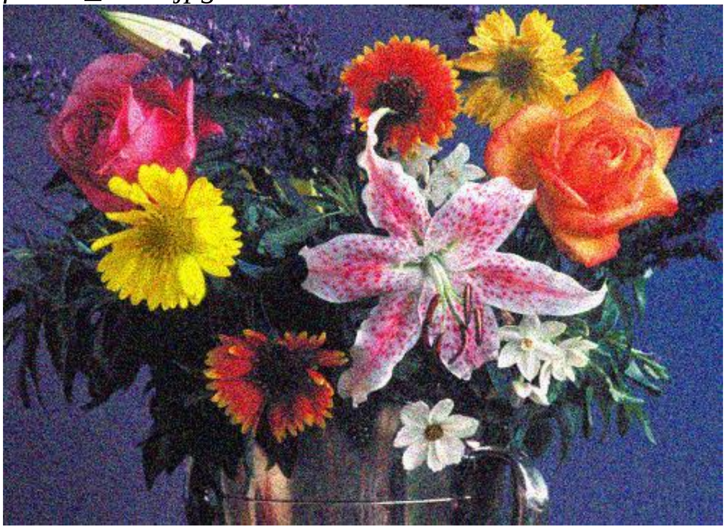


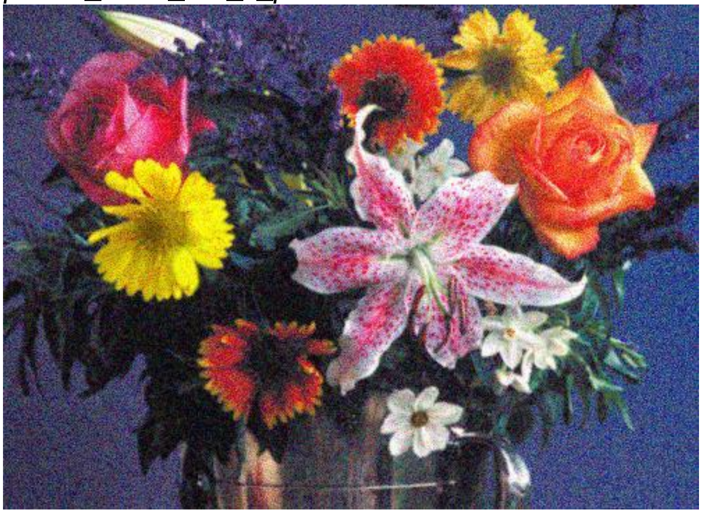
image with noise, averaged each of the r g b components with [1 2 1]/4 filter
flower_noise_rgbfilt.jpg

when averaging an rgb color image at each of it's components, the result is a bit like using filters on gray level images, noise seems to reduce and picture is more blurred, this happens since if there is noise, the noise has value in each component and so by averaging each component, the noise is reduced for the component averaged, moreover the blurring occurs since again, when near an edge some color components will have different values and so the pixels around the edge in these components will have values between colors in one edge side and the other edge side

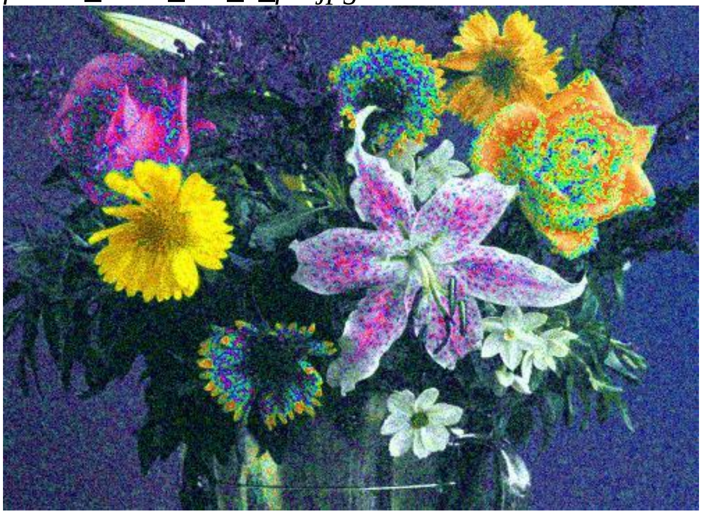
averaging color images - HSV

the color images are available on the disk with colors

filtered using [1 2 1]/4 on v component
flower_noise_hsv_v_filt.



filtered using [1 2 1]/4 on h component
flower_noise_hsv_h_filt.jpg



when activating filter on HSV components we see that the filter does not have the same effect as in the case of gray level images, when filtering the S or V components filtering causes minor changes and does not remove noise, when averaging over the H component the image is having different colors then the original image had, for instance the red flower becomes green and orange, this happens because small changes to hue which are the average result cause a different color, averaging 2 different colors creates a new color that is not any of the original colors

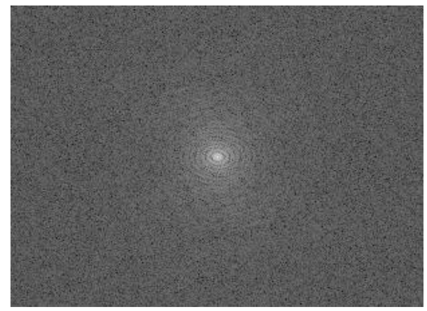
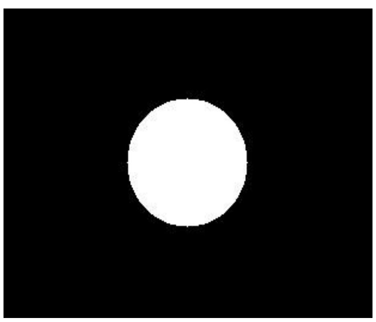
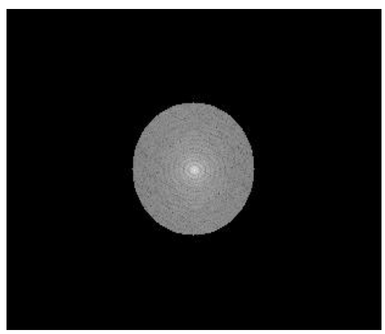
what I have learned from the assignment

from this assignment I have learned several things, using the averaging filters I learned that they do reduce Gaussian noise but at the expense of causing the image to be blurred and making edges smoother, the smoothing of edges is shown especially in the case of the pattern image where thin spaces between lines becomes as one thick line, using different averaging weights it shows that the smoothing of edges can be a little differently when in the [1 2 1]/4 filter, the chessboard edges appears better then in the [1 0 1]/2 case, I also saw the different effect of padding when the extrapolation padding seemed to cause the least picture edges different while zero padding produces a gray line in picture edges and periodic padding may cause gray lines and may not, depending on the opposite picture edge, I also tested and saw that the difference between filtering with periodical padding and multiplying the fourier coefficients by the values as calculated in class for the same filter produces very similar values, up to 10^-9 due to the number representation in the matlab,
doing filtering in fourier space, I found that for ideal filter, the bigger the radius, the image noise stays almost the same, and the smaller the radius, the noise reduces but the image starts having sort of shadows of large objects, the exponent functionlooks a bit better then the ideal filter but again when n is very large the picture is blurred and less details can be noticed, the butterworth filter allows for something in between the ideal and the exponent function, depending on the parameters
I also noticed that filtering by average on HSV color picture over the hue value can cause results that does not reduce the noise but cause the picture to have different colors since averaging two colors has no special meaning and the average color can be a different color

Frequency Domain Filtering

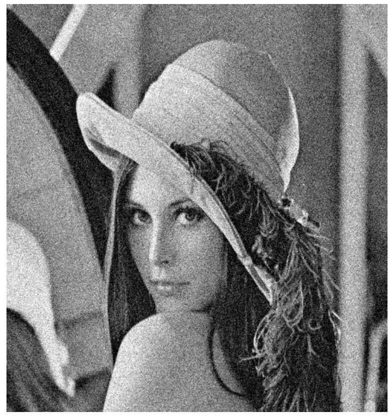
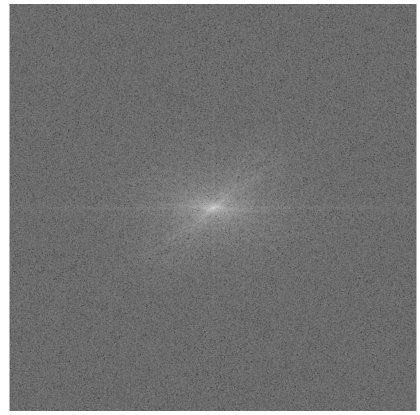
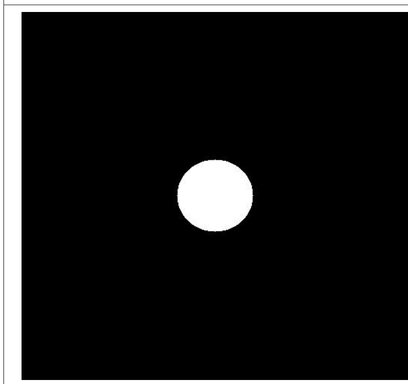
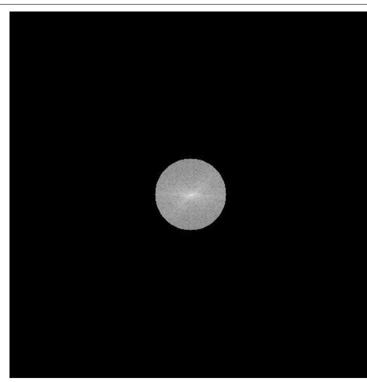
Ideal Low Pass Filter

Additive Gaussian noise ~(0, 0.01)

Original Image	Noisy Image	Frequency Space
		
Filter	Applied Filter	Image Space Result
		

Artifacts in the resulting image are highly visible. Also, the edges on the boundaries and on the surface of the coin suffer from degradation.

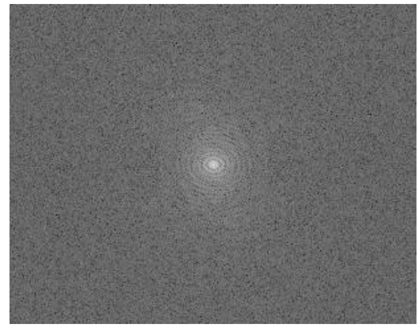
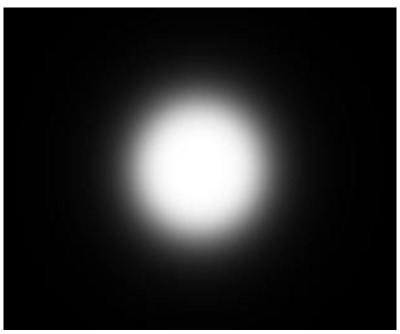
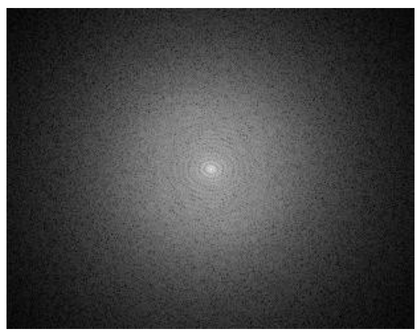
Additive Gaussian noise ~(0, 0.01)

Original Image	Noisy Image	Frequency Space
		
Filter	Applied Filter	Image Space Result
		

Artifacts in the resulting image are highly visible. The detail loss at the edges seems to be less severe than in the coins image.

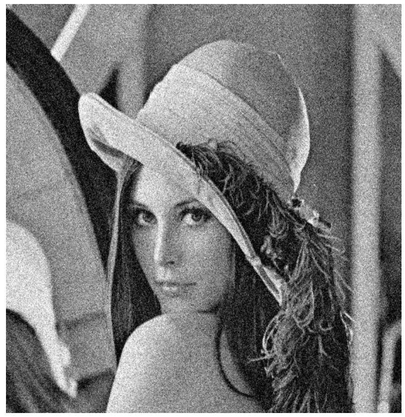
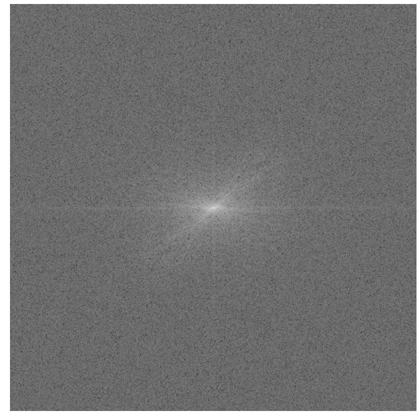
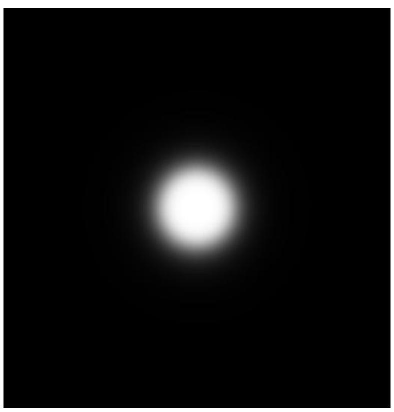
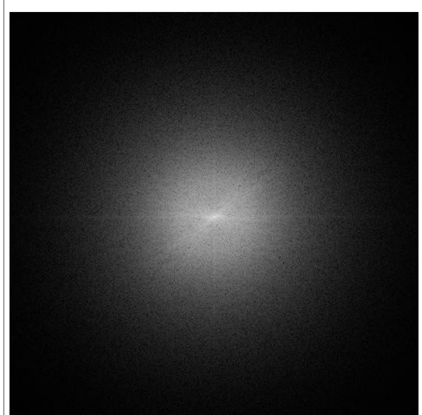
Butterworth Low Pass Filter

Additive Gaussian noise ~(0, 0.01)

Original Image	Noisy Image	Frequency Space
		
Filter	Applied Filter	Image Space Result
		

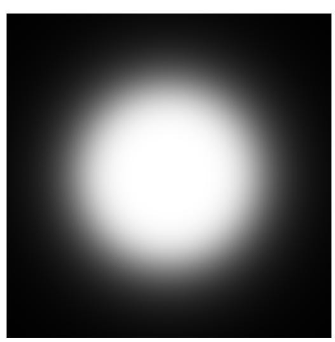
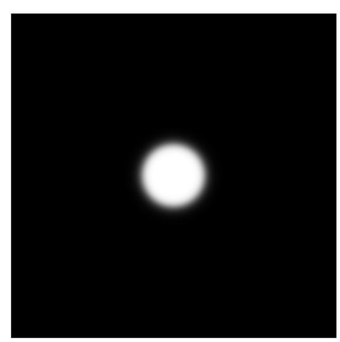
On first sight, this result doesn't seem to be better than the ideal low pass filter. A more careful examination shows that the artifacts on the table top surface are weaker, with no visible sharpness loss.

Gaussian noise ~(0, 0.01)

Original Image	Noisy Image	Frequency Space
		
Filter	Applied Filter	Image Space Result
		

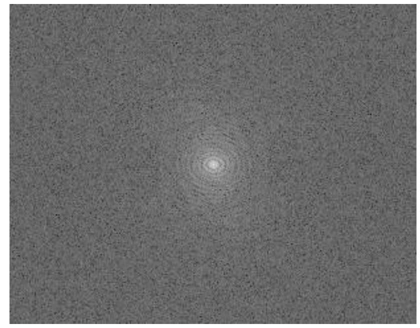
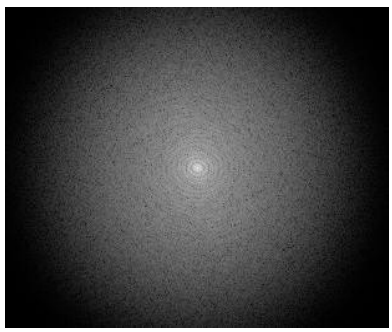
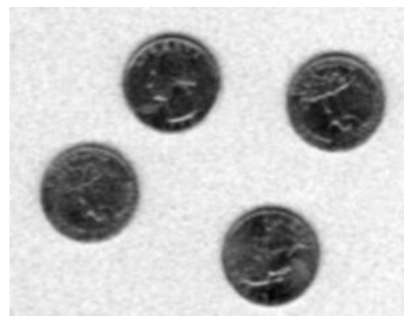
In this image the artifacts are weaker than in the ideal low pass filter result as well.

The Butterworth filter has two parameters. One of them controls the scale, and the other controls the decay rate. Here are a few examples of how the parameters influence the filter:

Base	Increasing the scale	Increasing the decay rate
		

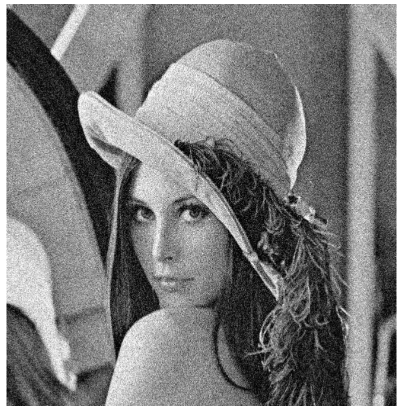
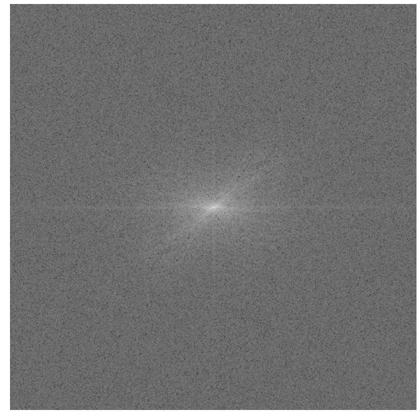
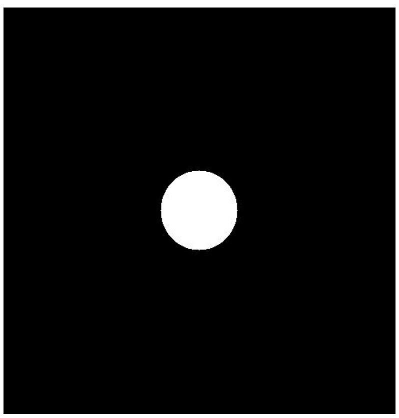
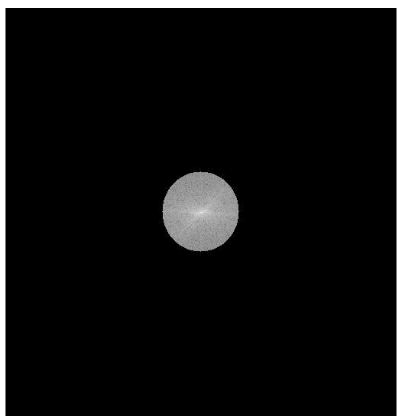
Gaussian Low Pass Filter

Additive Gaussian noise ~(0, 0.01)

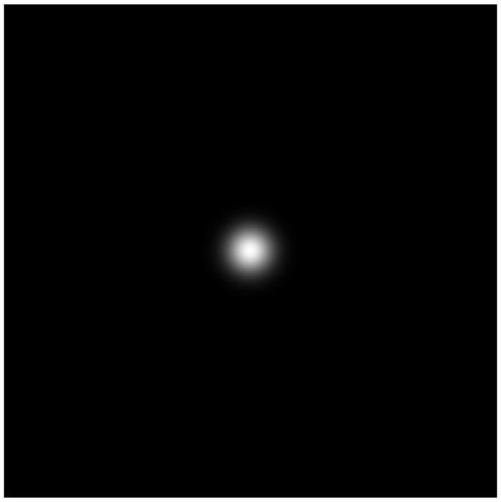
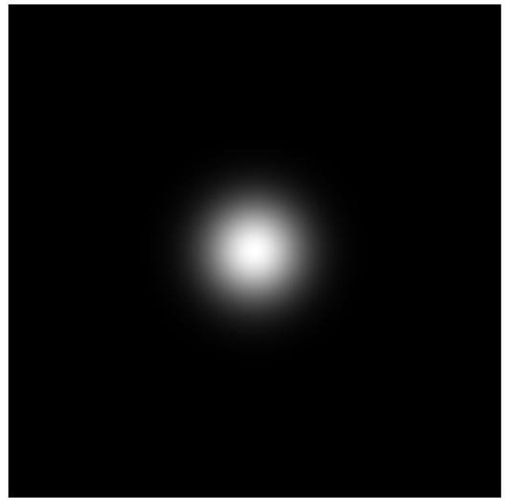
Original Image	Noisy Image	Frequency Space
		
Filter	Applied Filter	Image Space Result
		

The results of this filter are on par with the Butterworth filter. An interesting observation is the difference in local brightness between the results of the Gaussian and the Butterworth filter. This happens, of course, due to the different decay functions.

Additive Gaussian noise ~(0, 0.01)

Original Image	Noisy Image	Frequency Space
		
Filter	Applied Filter	Image Space Result
		

Now, let us see the difference between the results of Gaussian filter ~ (0, 25) and Gaussian filter ~ (0, 50). The image is the same as was used above – lenna with additive Gaussian noise ~ (0, 0.01)

Gaussian ~ (0, 25^2)	Gaussian ~ (0, 50^2)
	
	

In the left-most image we can see that the noise has been cancelled, together with most of the edge information of the image.

To sum up the frequency filter tests, the simple filters which suppress high frequencies do not produce good results on images where edges and other high frequency features provide important visual information. That information is being lost together with the noise.

נתבונן בתמונה RGB ונבצע עליה את הפילטר על כל צבע בנפרד

ולאחר הפילטר:

הפילטר לא הצליח להתמודד טוב עם תמונת צבע כאשר נעשה על כל חלק הפעולה, יתכן ובתמונות אחרות הפילטר יעבוד יותר טוב