Øving 2 – Mikkel Svagård

TDT4194, Visuell Databehandling

Teori:

1)

The convolution theorem allows for doing a spatial filtering operation in the frequency domain instead. By transforming the mask and the image to the frequeny domain we can do the operation there instead, then transforming back to the spatial domain again. ·

Wants to apply transformation h(x,y) * f(x,y) = g(x,y)

Rather transforming to frequency domain:

$\mathcal{F}\{h(x,y) * f(x,y)\}$ and doing the operation = H(u,v)F(u,v)

Now transforming back to frequency domain:

$\mathcal{F}^{-1}\{H(u,v)F(u,v)\} = g(x,y)$

2)

High and low-pass filters is filters that passes frequencies above/below a certain frequency known as the cut-off frequency. Frequencies not making it through this filter gets attenuated, as in losing its intensity gradually. High-pass filtering applied in the frequency domain can will achieve image sharpening, whilst low-pass achieve a smoothing effect.

3)

Starting with b, this is a clear low-pass filter as the low frequencies are centred in the middle. It heavily resembles the Gaussian Low-pass Filter.

The two other filters are high-pass filters, as the low frequencies are left out whilst the higher frequencies near the edges passes.

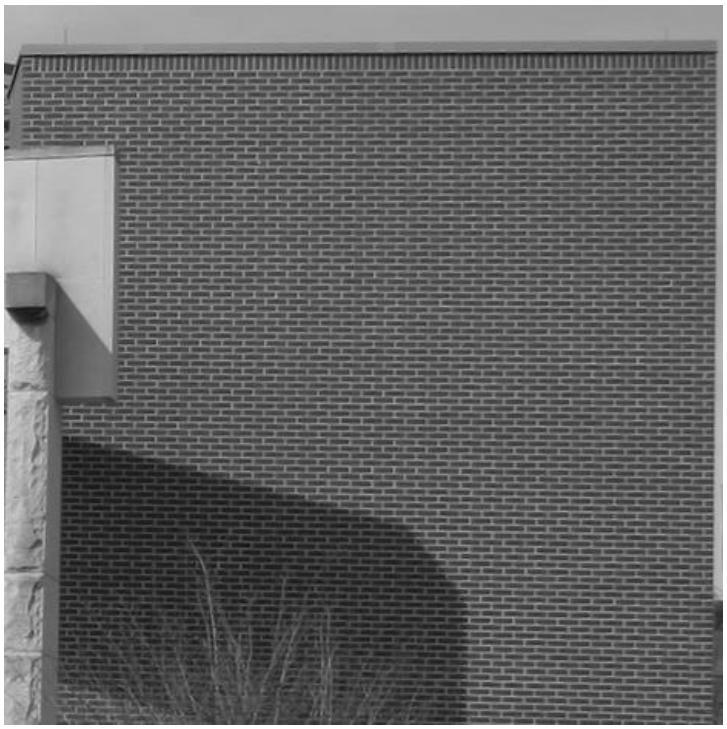
4)
Removing all the low frequencies will sharpen the image.
Removing all high frequencies will blur the image.

5)
Transforming a and b to the frequency domain gives two aligned dots horizontally on the image, mirrored around the centre. The distance to the centre will depend on the frequency of the wave of the picture, where b's dots will we further out than a's.
C will have an image of a similar manner, just with the dots plotted Diagonally, following the wave motion of the original picture.
As all pictures have waves going from black to white, the amplitude won't change.

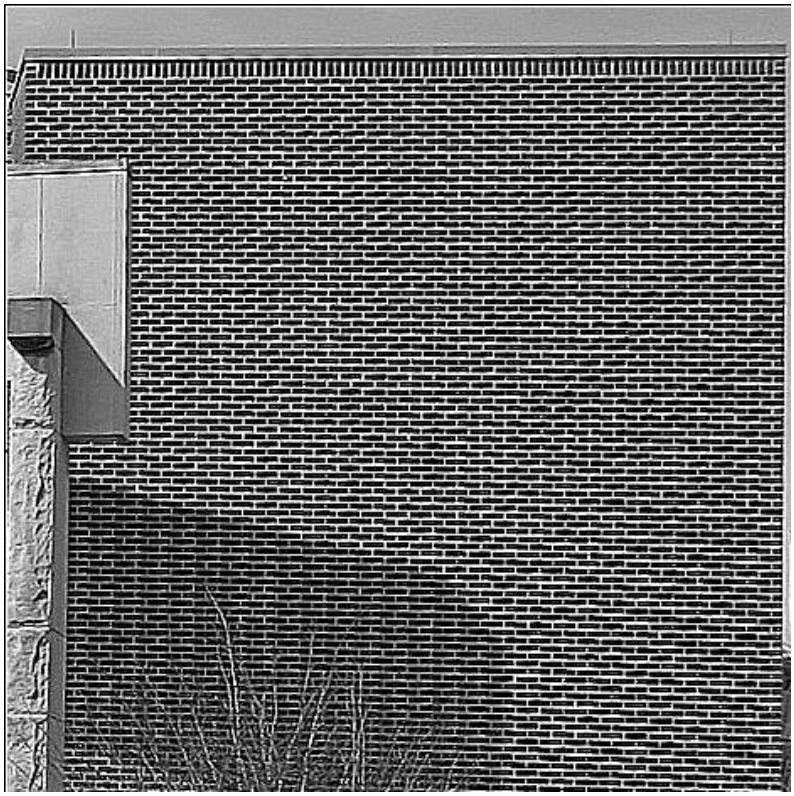
Programming:

Task 1a)

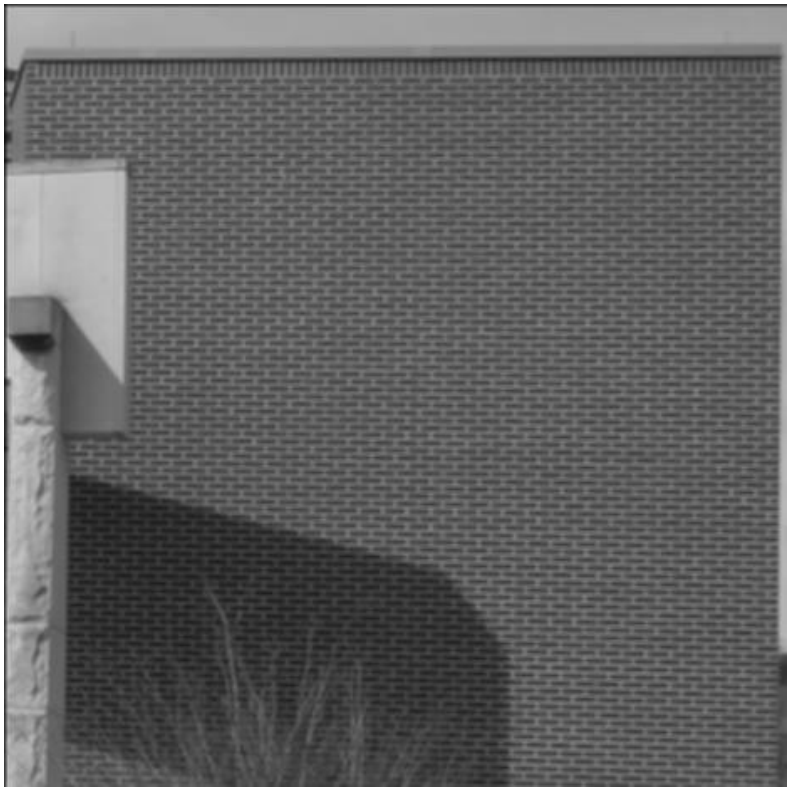
The original picture:



Highpass kernel:

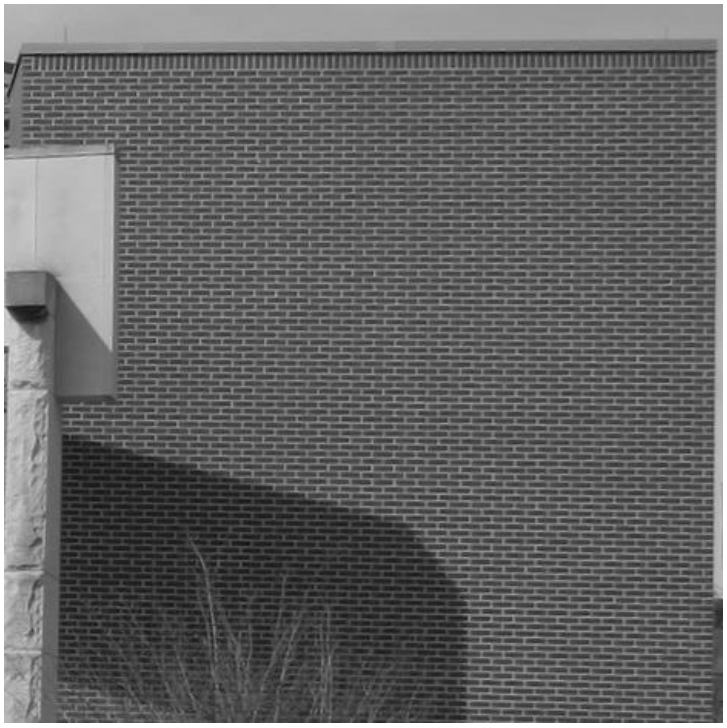


Lowpass Kernel:

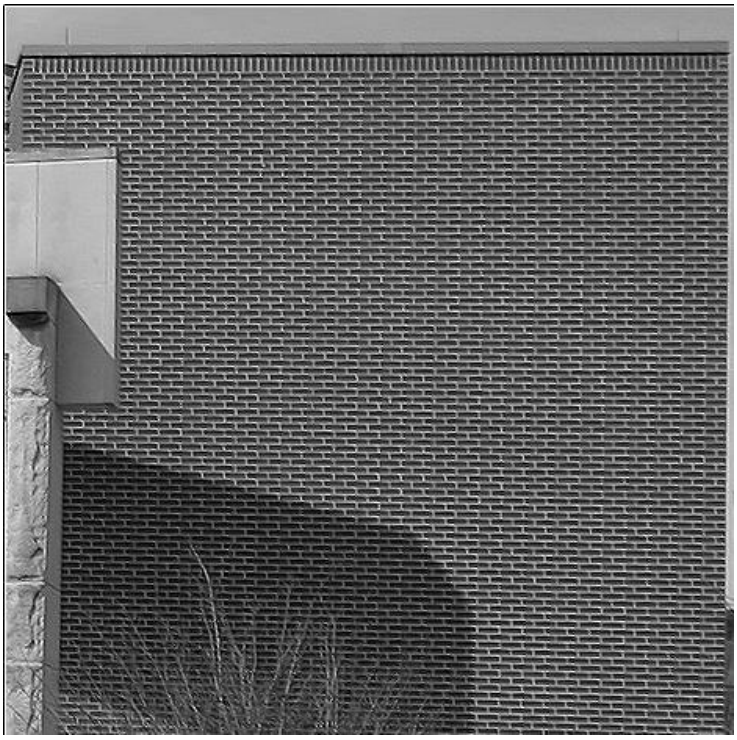


Task 1b)

Original:



Laplacian:

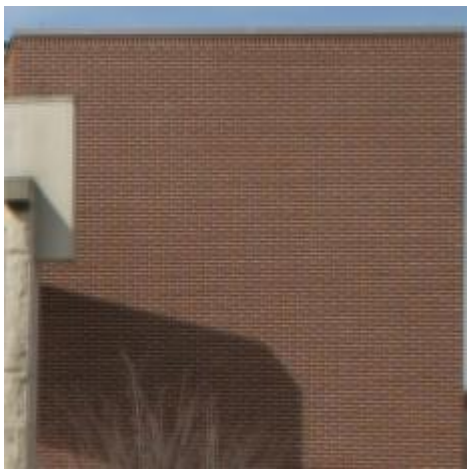


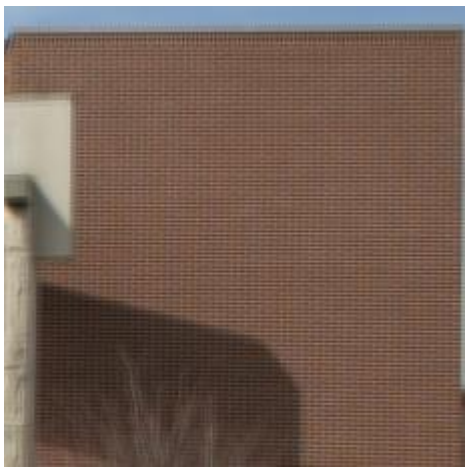
Task 2a)

Original (467*467)

Underpass (233*233)

Task 2b) The brick image above, smoothed with a 3x3, 5x5 and 7x7 Gaussian Kernel with σ = 1 before the downsampling.





Task 3) The exam question

As this is the first year we are using computers, I assume there will be some programming:

-Write a program that returns a Gaussian Kernel of NxN with σ=x when calling - *create_kernel(n,x)*
Further show how you can use built-in functions to transform it to the frequency domain.

Soluton:

```
def create_kernel(n,q=1):
    k = []
    c = int(n/2)
     v = 0
    for a in range(n):
        k.append([])
        for b in range(n):
            exp = (math.pow(a-c,2) + math.pow(a-c,2))/(2*q2)
            value =
1/((2*math.pi)*math.pow(q,2)*math.pow(math.e,exp))
            k[a].append(value)
            v += value
  return  [[(k[x][y]/s) for y in range(len(k))] for x in range(len(k))]
```

“Numpy allows us transforming this with the following function:”

```
kernel = create_kernel(3,1)
kernel_transformed = np.fft.fft2(kernel)
```