ECG 782 HW 2

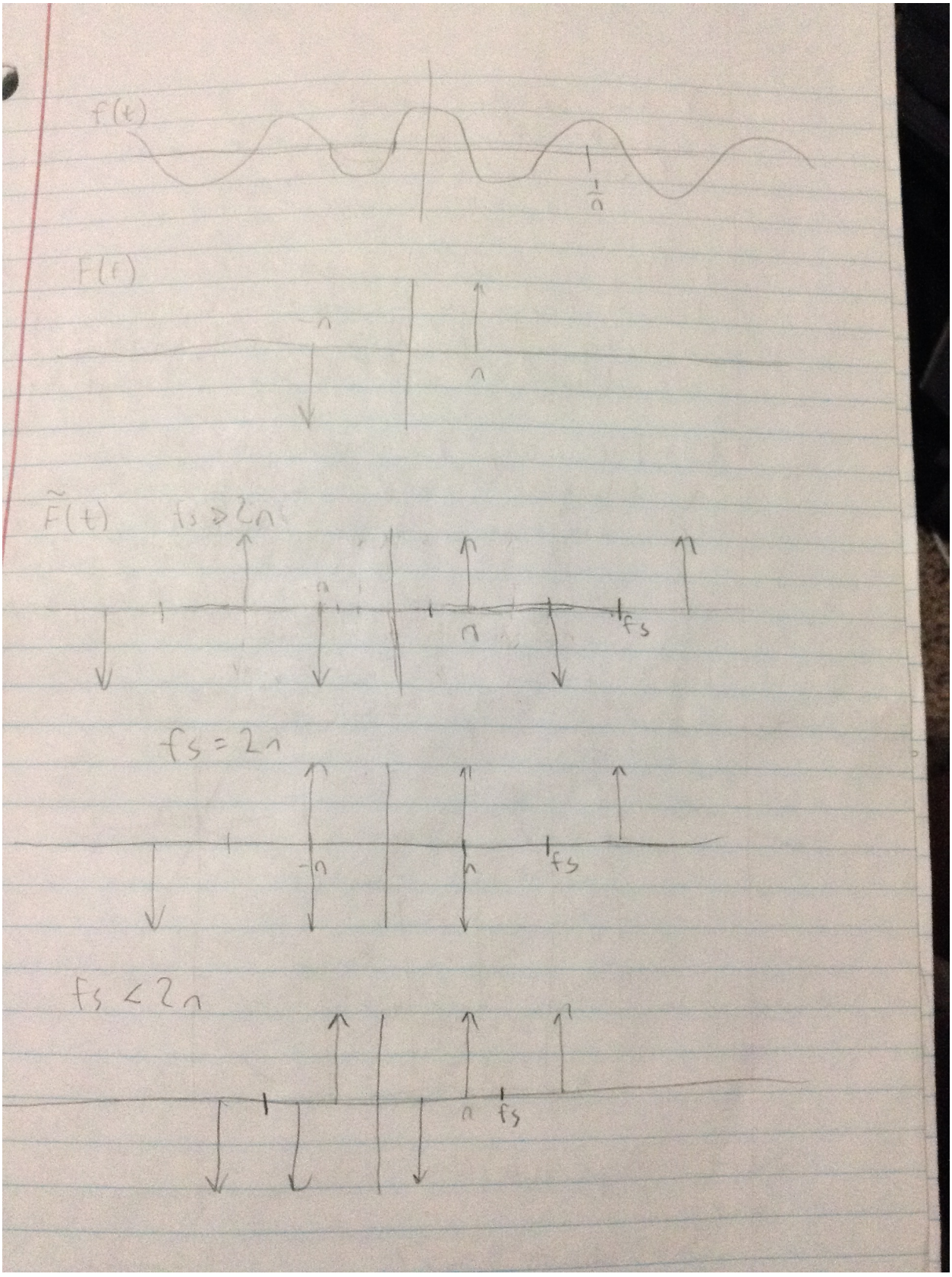
Carlo Lopez-Tello

2015-10-7

1 Problem 4.4: Features of cosine

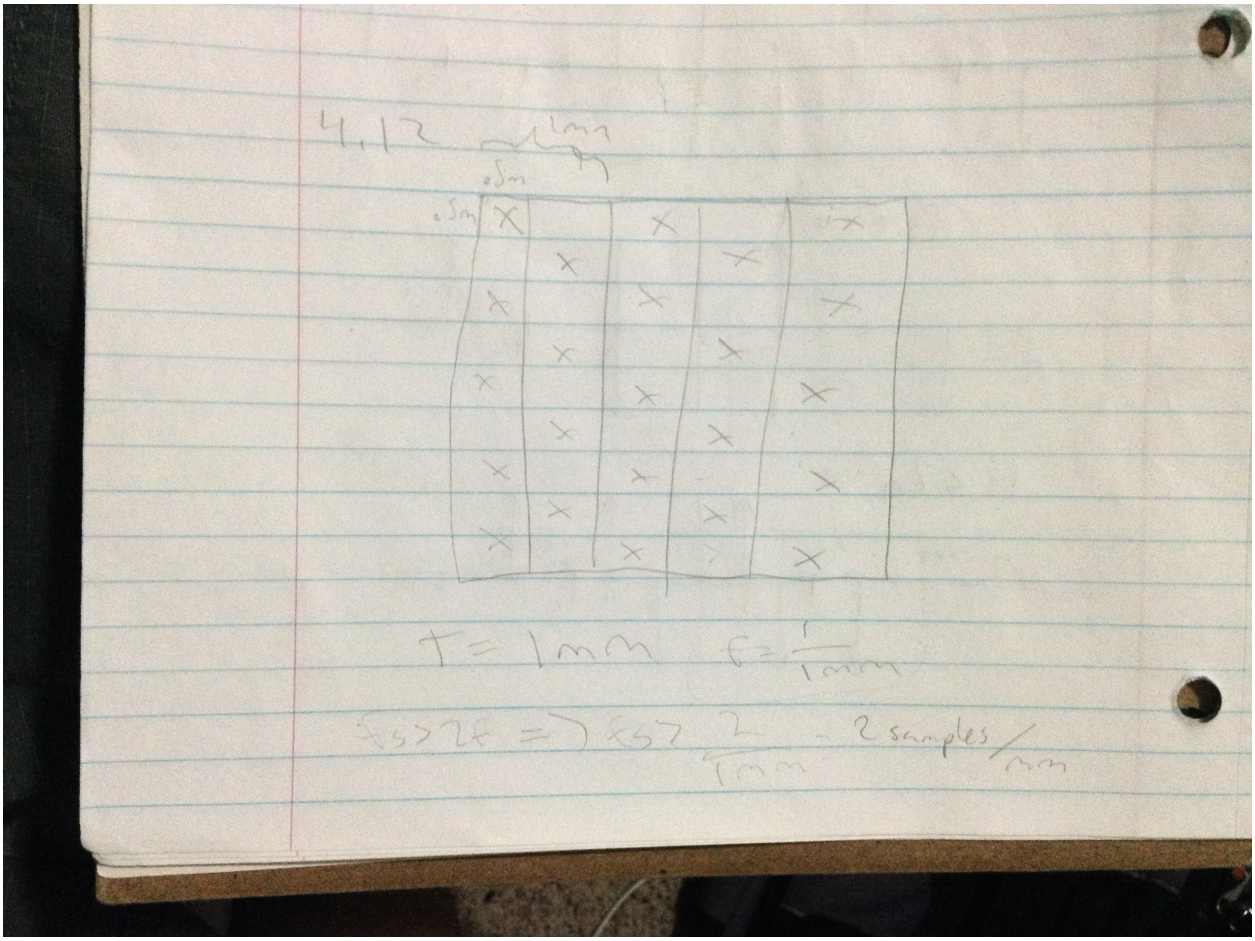
$$f(t) = cos(2\pi nt)$$

The period of $f(t)$ is $T = \frac{2\pi}{2\pi n} = \frac{1}{n}$.
The frequency is $f = \frac{1}{T} = n$.
The Fourier transform of $f(t)$, $F(f)$, will look like two deltas, one positive at $f = n$ and one negative at $f = -n$. If $f(t)$ is sampled at a higher sampling frequency than $fs > 2n$ then there will be copies of $F(f)$ every fs. The copies will be far enough apart that they will not interfere with one another. If $f(t)$ is sampled at $fs = 2n$ the copies will be too close and cancel each other out. If $f(t)$ is sampled at $fs < 2n$ the copies will be too close and interfere with each other. It will cause the sampled version of $f(t)$ to have more frequency components. The extra frequency components will be at $fs - n$ or $fs - n$ depending on fs.


f(t)
1/n
F(f)
n
n
F̃(t) fs ≥ 2n
n
n
fs
fs = 2n
-n
n
fs
fs < 2n
n
fs

2 Problem 4.12: Sampling of checkerboard

The period of the checkerboard image is $T = 1mm$. This implies that the image should be sampled at a rate $fs > \frac{2}{1mm}$.



3 Problem 4.32: What is the form of the spatial domain Gaussian high pass filter

A high pass filter can be made using a low pass.

$$H_{HP} = 1 - H_{LP}$$

A Gaussian low pass filter has the following form

$$H_{LP} = e^{-\frac{\mu^2+v^2}{2\sigma^2}}$$

therefore

$$H_{HP} = 1 - e^{-\frac{\mu^2+v^2}{2\sigma^2}}$$

we can apply the inverse Fourier transform to get the spatial domain version

$$h_{HP} = \delta(t,z) - 2\pi\sigma^2 e^{-2\pi^2\sigma^2(t^2-z^2)}$$

4 Problem 4.39: Applying high frequency emphasis and histogram equalization

The order of operations matters, because histogram equalization is a non linear operation. Therefore, the results will differ depending on which operation is applied first. High frequency emphasis should be applied first, since it reduces image contrast. If one were to apply histogram equalization first it would most likely be undone by high frequency emphasis.

5 Problem 4.43: Cleaning up images

1.The best way to remove bright spots would be median filtering.
2.To sharpen an image one should do high frequency filtering and add the edges to the original image.
3.To improve contrast one can apply histogram equalization or gamma correction.
4.If the average intensities vary between images one should first find the desired average intensity I_{av}. Then one can find the difference between the average intensity of an image and the desired intensity $I_{df} = I_{av} - I_{im}$. To adjust the intensity of a specific image all on has to do is add the difference $I_{df}$to every pixel in the image. This procedure works because if one adds a constant to every pixel the average of all the pixels increases by that constant.

6 Problem 6: Filtering in spacial domain

6.1 Image smoothing

original image

image filtered with 7x7 mean filter

image filtered with 7x7 gaussian filter with .5 sigma

image filtered with 7x7 gaussian filter with 3 sigma

The mean filter produces the blurriest image followed by the wider Gaussian filter.

6.2 Edge enhancement

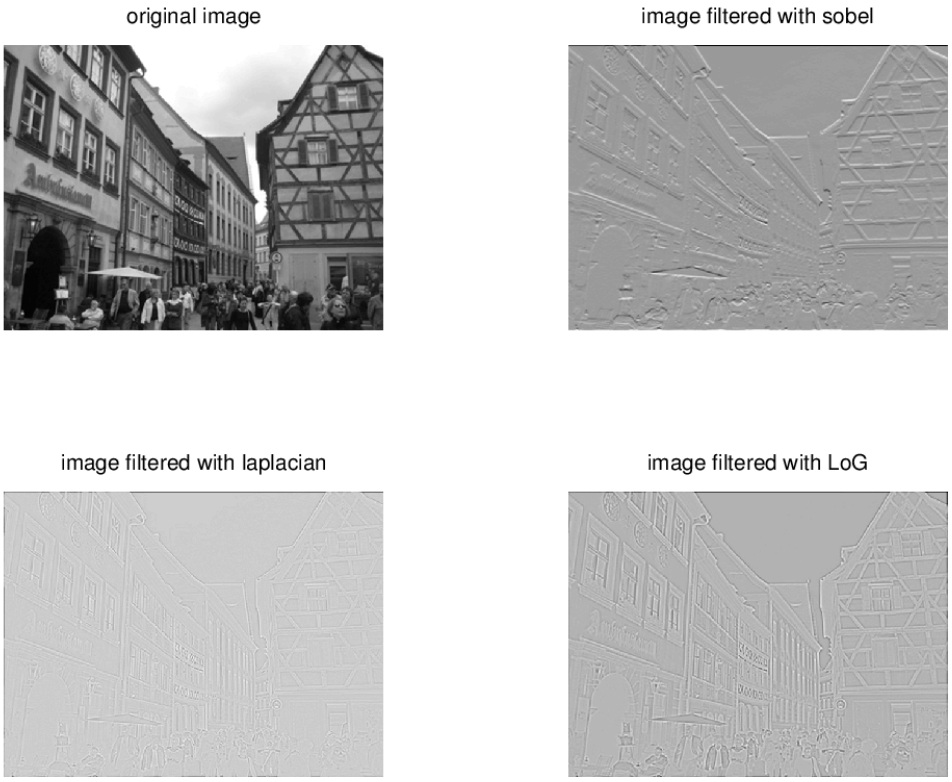


Figure 1: output of filters

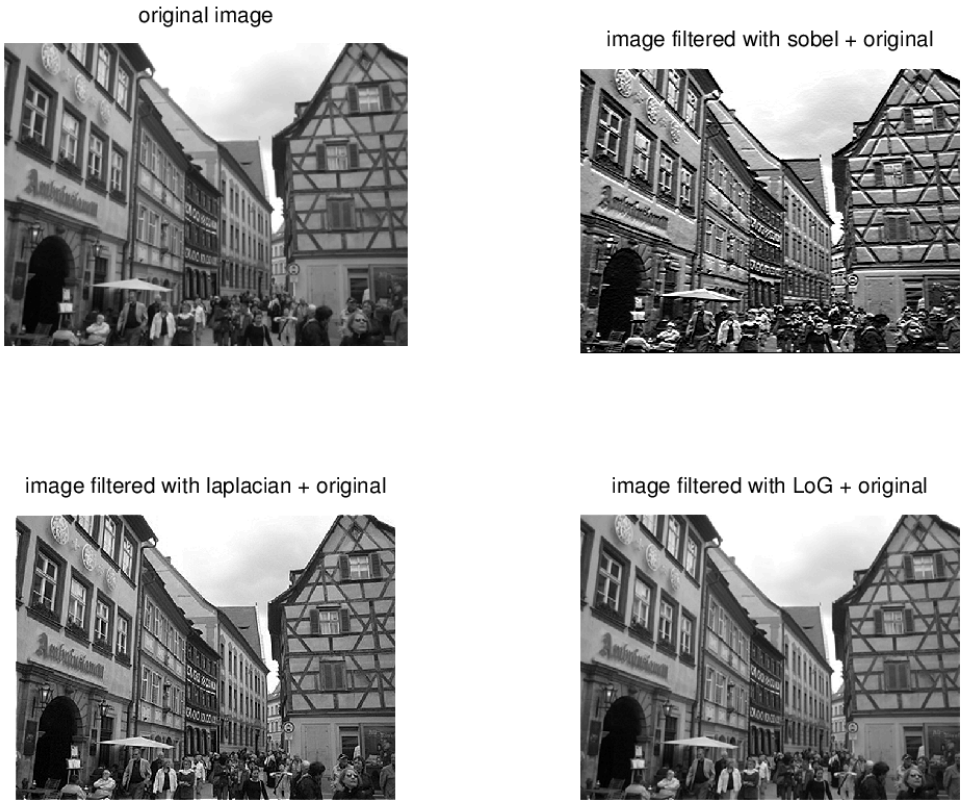


Figure 2: enhanced images

The sobel filter produced the thickest edges, but distorted the image. The laplacian accentuated the edges without distorting the image too much. The LoG sharpened the image without distorting it.

7 Problem 7: Filtering in frequency domain

7.1 Plot the Fourier transform of 'city.jpg'

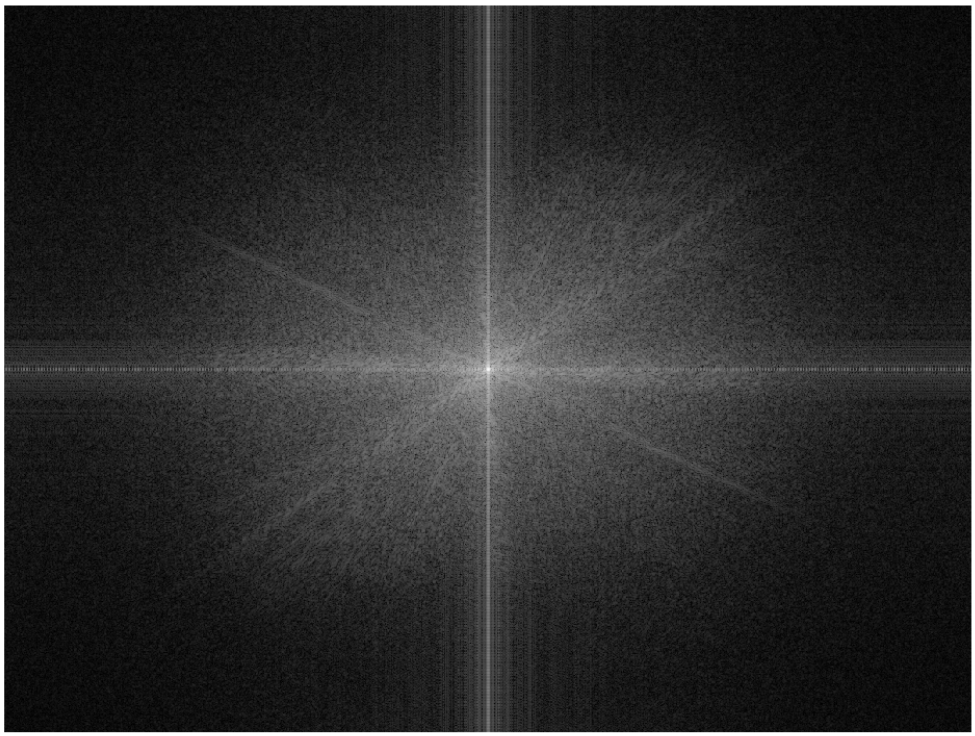


Figure 3: Fourier transform of 'city.jpg'

7.2 Comparison between freq domain and spatial domain filtering for smoothing

Figure 4: image filtered with 7x7 mean filter

Figure 5: image filtered with 3x3, .5 sigma Gaussian filter

Figure 6: image filtered with 3x3, 3 sigma Gaussian filter

There is a significant decrease of performance (5 to 10 times slower) when filtering in the frequency domain.

filter	time for spacial dom	time for freq dom
mean filter	.016	.157
.5 sigma Gaussian filter	.033	.197
3 sigma Gaussian filter	.038	.157

Table 1: time comparison for filtering in space vs freq domains

7.3 Comparison between freq domain and spatial domain filtering for edge enhancement

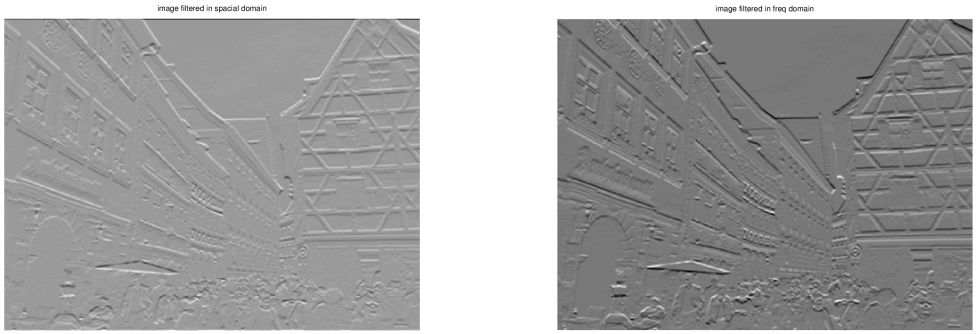


Figure 7: image filtered with sobel filter

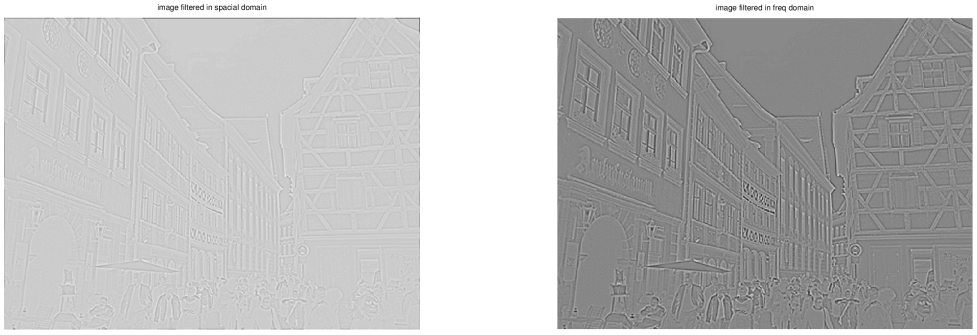


Figure 8: image filtered with laplacian filter

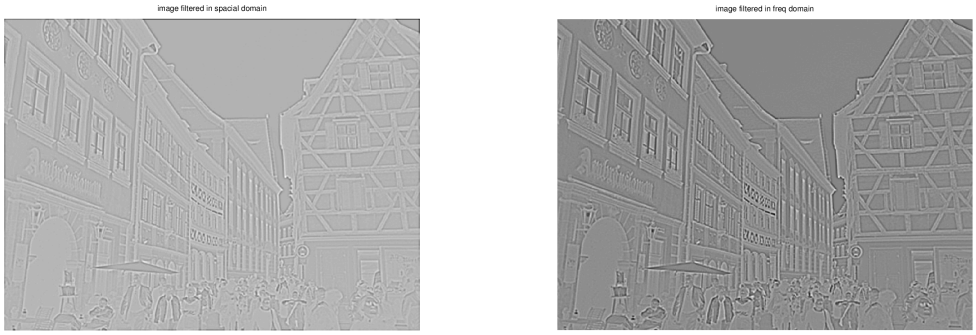


Figure 9: image filtered with LoG filter

There is a significant decrease of performance (25 to 40 times slower) when filtering in the frequency domain.

filter	time for spacial dom	time for freq dom
sobel	.006	.169
laplacian	.006	.223
LoG	.012	.313

Table 2: time comparison for filtering in space vs freq domains

7.4 Ideal low pass and Gaussian

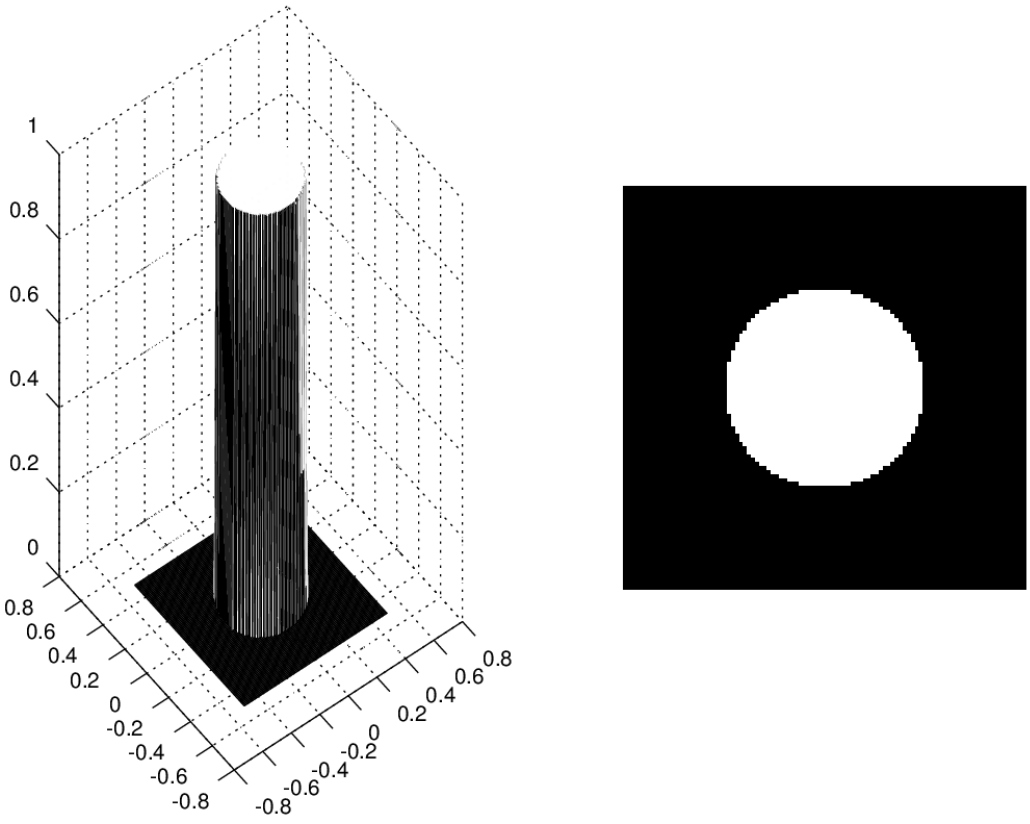


Figure 10: ideal low pass in freq domain

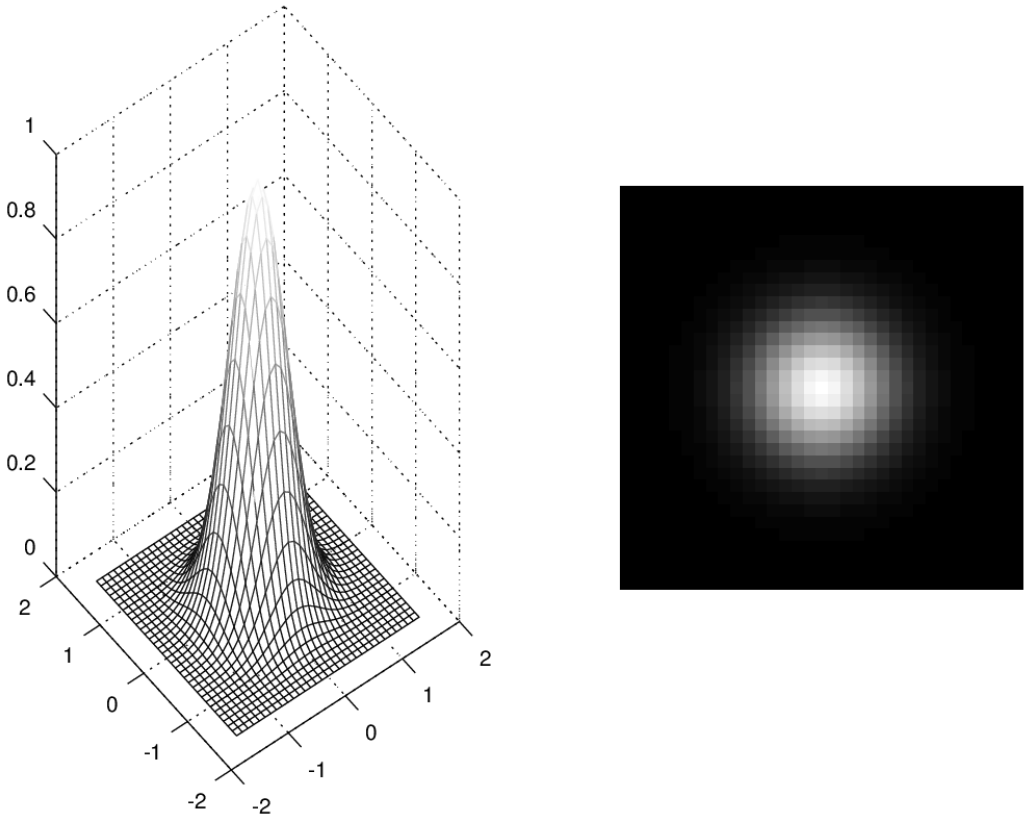


Figure 11: Gaussian filter with .4 sigma in freq domain

7.5 Rectangular filter

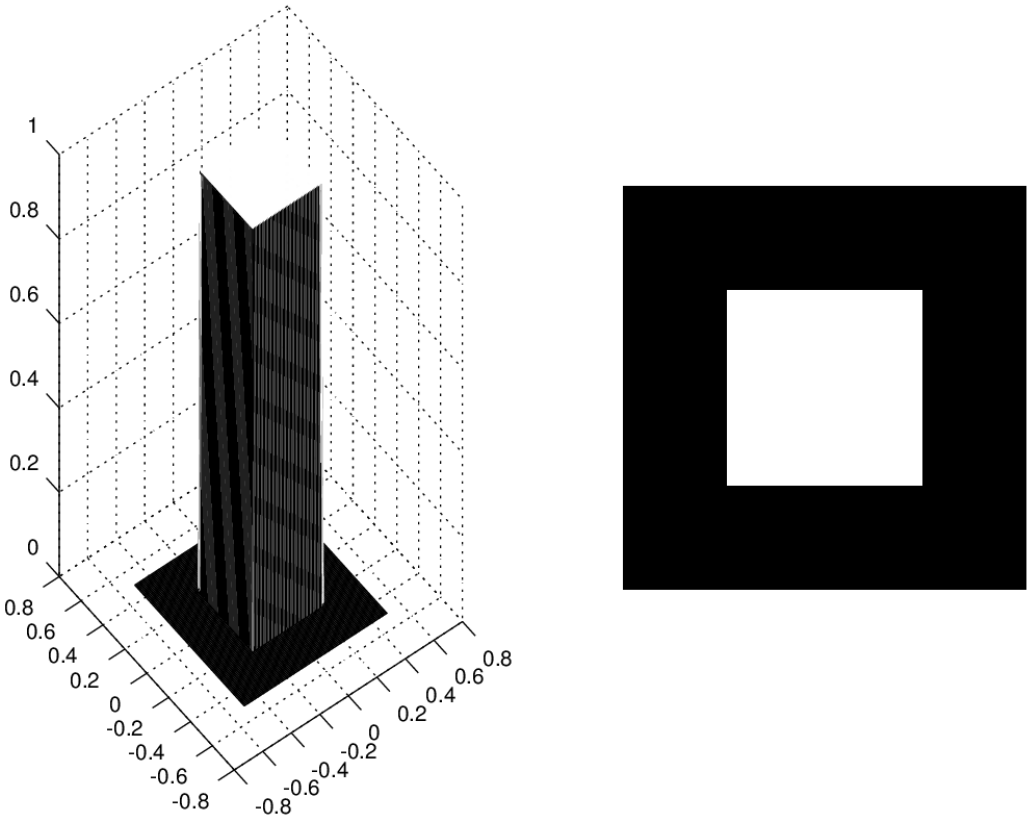


Figure 12: rectangular filter in freq domain

8 Code

8.1 Q6

Code for smoothing

```
%image = double(imread('city.jpg'))/255;
image = double(rgb2gray(imread('city.jpg')))/255;
figure(1)
subplot(2,2,1);
imshow(image);
title('original image');

filter = fspecial('average',7);
filteredImage = imfilter(image,filter);
subplot(2,2,2);
imshow(filteredImage);
title('image filtered with 7x7 mean filter');

filter = fspecial('gaussian',7,.5);
filteredImage = imfilter(image,filter);
subplot(2,2,3);
imshow(filteredImage);
title('image filtered with 7x7 gaussian filter with .5 sigma');

filter = fspecial('gaussian',7,3);
filteredImage = imfilter(image,filter);
subplot(2,2,4);
imshow(filteredImage);
title('image filtered with 7x7 gaussian filter with 3 sigma');

print(figure(1),'partA.png');
```

Code for sharpening

```
%image = double(imread('city.jpg'))/255;

image = double(rgb2gray(imread('city.jpg')))/255;

figure(1)
subplot(2,2,1);
imshow(image);
title('original image');

figure(2)
subplot(2,2,1);
imshow(image);
title('original image');

filter = fspecial('sobel');
filteredImage = imfilter(image,filter);
filteredImage = filteredImage - mean(mean(filteredImage));
figure(1)
subplot(2,2,2);
imshow(normalizeImage(imfilter(image,filter)));
title('image filtered with sobel');

figure(2)
subplot(2,2,2);
enhancedImage = image-filteredImage;
imshow(enhancedImage);
title('image filtered with sobel + original');


filter = fspecial('laplacian');
filteredImage = imfilter(image,filter);
filteredImage = filteredImage-mean(mean(filteredImage));
figure(1)
subplot(2,2,3);
imshow(normalizeImage(imfilter(image,filter)));
title('image filtered with laplacian');

figure(2)
subplot(2,2,3);
enhancedImage = image-filteredImage;
imshow(enhancedImage);
title('image filtered with laplacian + original');

filter = fspecial('log',5,.75);
filteredImage = imfilter(image,filter);
filteredImage = filteredImage-mean(mean(filteredImage));
figure(1)
subplot(2,2,4);
imshow(normalizeImage(imfilter(image,filter)));
title('image filtered with LoG');

figure(2)
```

```
subplot(2,2,4);
enhancedImage = image-filteredImage;
imshow(enhancedImage);
title('image filtered with LoG + original');

print(figure(1),'partB1.png');
print(figure(2),'partB2.png');
```

Utility Code

```
function [normalizedImage] = normalizeImage(image)
        MI = min(min(image));
        MA = max(max(image));
        normalizedImage = (image-MI)./(MA-MI);
end
```

8.2 Q7

Code to plot Fourier transform

```
image = double(rgb2gray(imread('city.jpg')))/255;
figure(1);
si = size(image);
row = si(1,1);
col = si(1,2);
Fimage = fft2(image,row*2,col*2);
Fimage = normalizeImage(log10(1+abs(Fimage)));
Fimage = fftshift(Fimage);
imshow(Fimage);

print(figure(1),'partA.png');
```

Code for smoothing

```
image = double(rgb2gray(imread('city.jpg')))/255;

filter = fspecial('average',7);
figure(1);
filterName = '7x7 mean filter';
spaceAndFreqFilter(image,filter,filterName);
print(figure(1),'partB1.png');

filter = fspecial('gaussian',7,.5);
figure(1);
filterName = '3x3 gaussian filter with .5 sigma';
spaceAndFreqFilter(image,filter,filterName);
print(figure(1),'partB2.png');

filter = fspecial('gaussian',7,3);
figure(1);
filterName = '3x3 gaussian filter with 3 sigma';
spaceAndFreqFilter(image,filter,filterName);
print(figure(1),'partB3.png');
```

Code for sharpening

```
image = double(rgb2gray(imread('city.jpg')))/255;

filter = fspecial('sobel');
figure(1);
filterName = 'sobel filter';
spaceAndFreqFilter(image,filter,filterName);
print(figure(1),'partC1.png');

filter = fspecial('laplacian');
figure(1);
filterName = 'laplacian filter';
spaceAndFreqFilter(image,filter,filterName);
print(figure(1),'partC2.png');

filter = fspecial('log',5,.75);
figure(1);
filterName = 'LoG filter';
spaceAndFreqFilter(image,filter,filterName);
print(figure(1),'partC3.png');
```

Utility Code

```
function [] = spaceAndFreqFilter(image, filter, filterName)
        tic();
        filteredImage = imfilter(image, filter);
        [filterName ' time domain']
        toc()

        subplot(1,2,1);
        imshow(normalizeImage(filteredImage));
        title(['image filtered in spacial domain']);


        si = size(image);
        row = si(1,1);
        col = si(1,2);

        tic();
        Fimage = fft2(image,row*2,col*2);
        Ffilter = fft2(filter,row*2,col*2);
        FfilteredImage = Fimage .* Ffilter;
        filteredImage = real(ifft2(FfilteredImage));
        filteredImage = filteredImage(5:row,5:col,:);
        [filterName ' freq domain']
        toc()

        subplot(1,2,2);
        imshow(normalizeImage(filteredImage));
        title(['image filtered in freq domain']);

end
```

Code for plotting filters

```
function [] = cylinderplot(h)
        r = .25*h;
        [X,Y] = meshgrid(-r*2:.01:r*2,-r*2:.01:r*2);
        Z = h.*(X.^2 + Y.^2 < r.^2);
        figure(1)
        subplot(1,2,1);
        mesh(X,Y,Z);
        subplot(1,2,2);
        imshow(Z);
        print(figure(1),'partD1.png');
end
```

```
function [] = gaussianplot(sigma)
        [X,Y] = meshgrid(-sigma*4:.1:sigma*4,-sigma*4:.1:sigma*4);
        Z = exp(-(X.^2 + Y.^2)/(2*sigma.^2))/(2*pi*sigma.^2);
        figure(1)
        subplot(1,2,1);
        mesh(X,Y,Z);
        subplot(1,2,2);
        imshow(Z);
        print(figure(1),'partD2.png');
end
```

```
function [] = rectangleplot(h)
        r = .25*h;
        [X,Y] = meshgrid(-r*2:.01:r*2,-r*2:.01:r*2);
        Z = h.*(abs(X) < r).*(abs(Y) < r);
        figure(1)
        subplot(1,2,1);
        mesh(X,Y,Z);
        subplot(1,2,2);
        imshow(Z);
        print(figure(1),'partE.png');
end
```