# Computer Vision I

Tim Luchterhand, Paul Nykiel (Group 17)

28. Mai 2018

## 1 Filter Algebra

### 1.1

Without loss of generality choose an image $I \in [0, 255]$ with size $3 \times 3$.

The maximum value for $I_{22}$ is achieved by weighting all positive values in the filter with 255 and all negativ values with 0. This results in image (1). With this image the maximum output value is $2 \cdot 255 + 2 \cdot 255 + 1 \cdot 255 = 1275$.

$$I_{max} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 255 \\ 0 & 255 & 255 \end{pmatrix} \tag{1}$$

The minimum value for $I_{22}$ is achieved by weighting all positive values in the filter with 0 and all negativ values with 255. This results in image (2). With this image the maximum output value is $-2 \cdot 255 + (-2) \cdot 255 + (-1) \cdot 255 = -1275$.

$$I_{max} = \begin{pmatrix} 255 & 255 & 0 \\ 255 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{2}$$

### 1.2

$$\begin{aligned} I &= (1\ 2) \\ H &= (1\ -1) \end{aligned}$$
$$\alpha = 255$$

With these values calculate the convolution (the values are continued with 0):

$$\begin{aligned}
(\alpha \cdot I) * H &= (255\ 255) * (1\ \ -1) = (255\ 0\ 0) \\
\alpha \cdot (I * H) &= 255 \cdot (1\ 1\ 0) = (255\ 255\ 0) \\
&\Rightarrow (\alpha \cdot I) * H \alpha \cdot (I * H)
\end{aligned}$$

As shown by the contradiction above the linearity does not hold for clamped values.

### 1.3

Matlab Code:

```
I = imread("lena.tif");
sigma = 3;
G = fspecial("gaussian", 2*ceil(2*sigma)+1, sigma);
H = [-1 -2 0; -2 0 2; 0 2 1];

T1 = imfilter(I, G, 'replicate', 'conv');
R1 = imfilter(T1, H, 'replicate', 'conv');

figure();
subplot(1,3,1);
imshow(I);
title("Original image");
subplot(1,3,2);
imshow(T1);
title("Blurred Image");
subplot(1,3,3);
imshow(R1);
title("Diagonal edges of the blurred image");

print("lenaEdge1.eps", "-depsc");
```

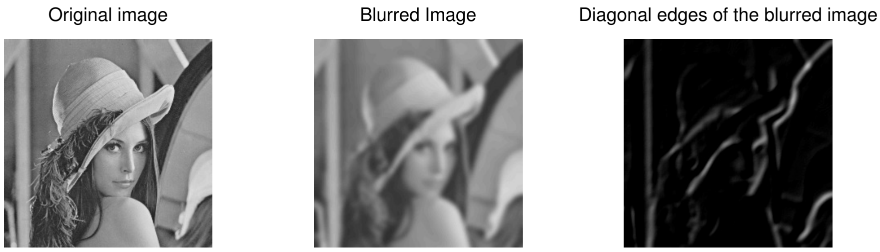


Abbildung 1: Output of the Matlab script

The filter highlights strong edges in the diagonal axis (north-west to south-east) by calculating an approximitation of the derivative in this direction.

### 1.4

```
T2 = imfilter(I, H, 'replicate', 'conv');
R2 = imfilter(T2, G, 'replicate', 'conv');

figure();
subplot(2,3,1);
imshow(I);
title("Original image");
subplot(2,3,2);
imshow(T1);
title("Blurred Image");
subplot(2,3,3);
imshow(R1);
title("Diagonal edges of the blurred image");
subplot(2,3,4);
imshow(abs(R2-R1));
title("Difference of both images");
subplot(2,3,5);
imshow(T2);
title("Diagonal edges of the original image");
subplot(2,3,6);
```

```
imshow(R2);
title("Blurred image of the diagonal edges");

print("lenaEdge2.eps", "-depsc");
```

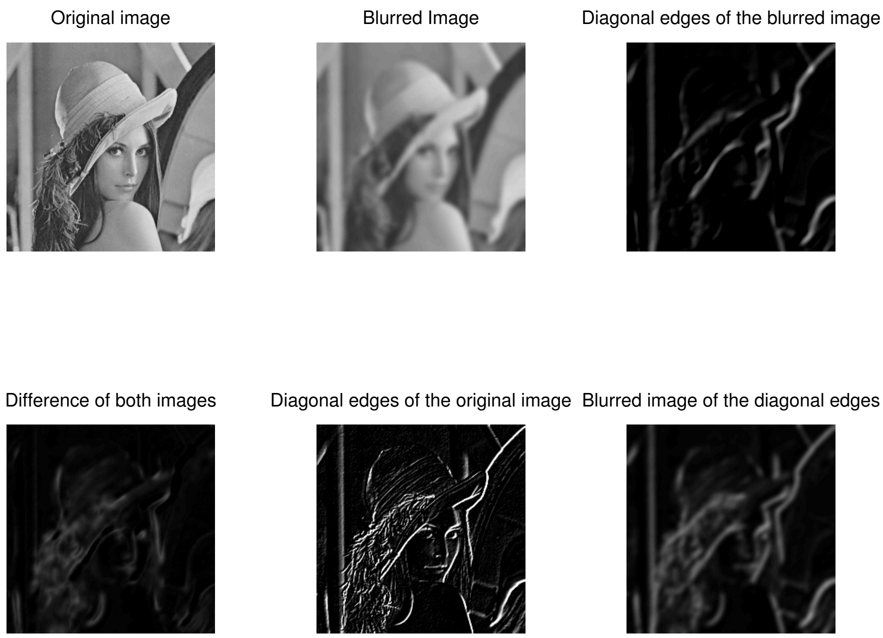


Abbildung 2: Output of the second part of the Matlab script

The second image looks slighly more detailed. When using clamped images the convolution producrt is no longer commutative.

# 2 Discrete Fourier Transform

## 2.1

```
%% Part a
N = 4;
ti = [0 1/30 2/30 3/30];
si = [2 3 0 1];

S = zeros(size(ti));
f = zeros(size(ti));

```

```
for u=0:(N-1)
    for t=0:(N-1)
        S(u+1) += si(t+1) * exp(-2*pi*i*u/N*t);
    end;
    f(u+1) = 2*pi*u/N;
end;
S ./= sqrt(N);
```

### 2.2

The $u$th coefficient (starting at zero) corresponds to a frequency of

$$f(u) = 2 \cdot \pi \cdot \frac{u}{N} = \frac{\pi \cdot u}{2}$$

```
%% Part b
x = 0:0.01:1;
y = sin(2*pi*x*1/N*3);
figure();
plot(x,y);
title("A sine-wave with the frequency of S(3)");
xlabel("t");
ylabel("y(t)");
print("frequency.eps", "-depsc");
```

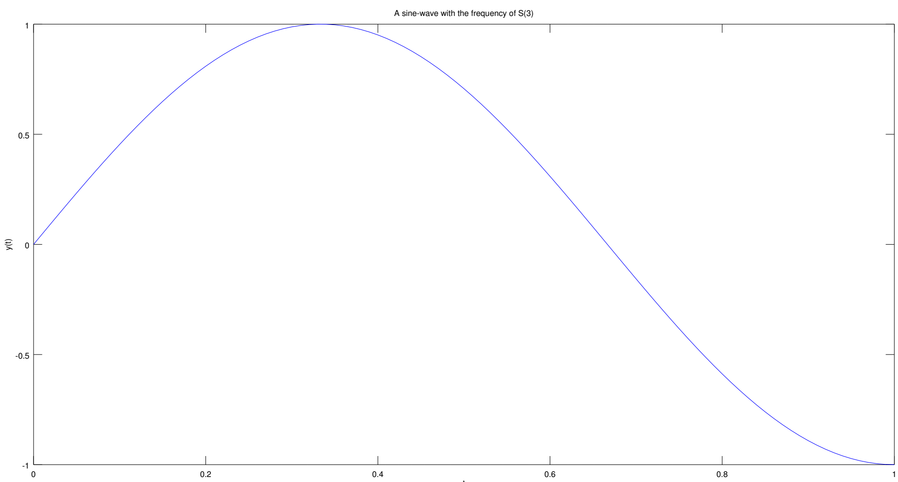


Abbildung 3: Output of the second part of the Matlab script

### 2.3

```
%% Part c
sr = zeros(size(ti));
for u=0:N-1
    for t=0:(N-1)
        sr(u+1) += S(t+1) * exp(2*pi*i*u/N*t);
    end;
end;
sr ./= sqrt(N);
```

### 2.4

```
%% Part d
figure();
subplot(3,1,1);
plot(ti,si);
xlabel("t");
ylabel("s_i(t)");
subplot(3,1,2);
plot(f,S);
xlabel("f");
ylabel("S(f)");
subplot(3,1,3);
plot(ti,sr);
xlabel("t");
ylabel("s_r(t)");
print("transforms.eps", "-depsc");
```

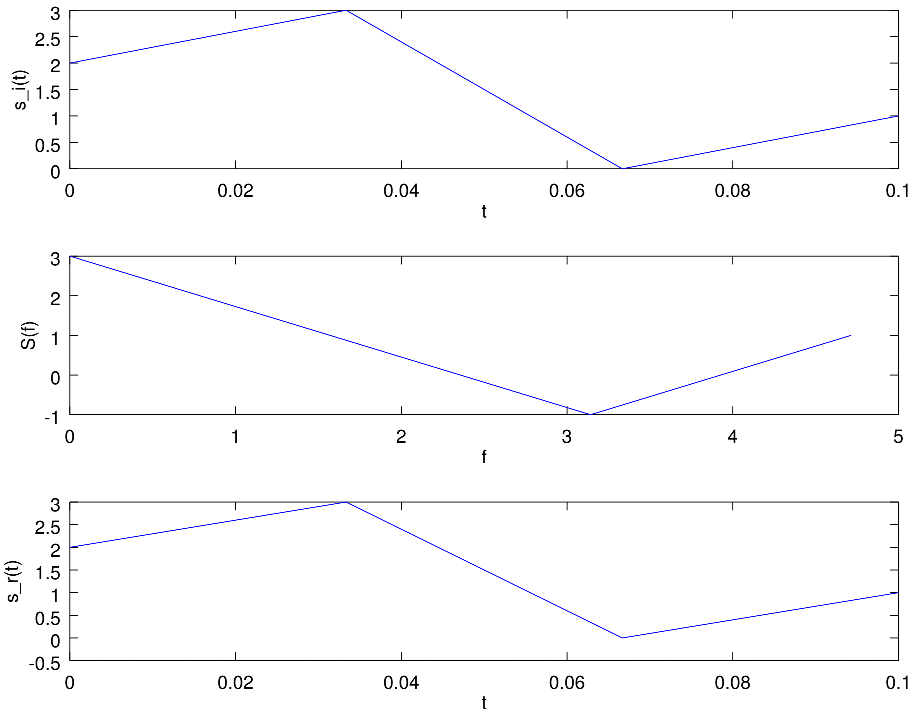


Abbildung 4: Output of the last part of the Matlab script

# 3 Fourier Transform for Image Quality Assessment

## 3.1

```
%% Part a
I1 = double(imread("flower01.png"))/255;
I2 = double(imread("flower02.png"))/255;

F1 = fftshift(fft2(I1));
F2 = fftshift(fft2(I2));

F1 = abs(F1);
F2 = abs(F2);
```

## 3.2

```
%% Part b
```

```
sigma = size(I1,1);
G = fspecial('gaussian', sigma, sigma/2);
G = padarray(G,(size(F1)-size(G))/2,0, 'replicate');
G /= max(max(G));
H = ones(size(G)) - G;

H1 = F1 .* H;
H2 = F2 .* H;
```

### 3.3

```
%% Part c
H1 .*= H1;
H2 .*= H2;

E1 = sum(sum(H1));
E2 = sum(sum(H2));
```

### 3.4

```
%% Part d
figure();
subplot(3,3,1);
imshow(I1);
title("flower01.png");
subplot(3,3,2);
imshow(F1);
title("Fourier transform of flower01.png");
subplot(3,3,3);
imshow(H1);
title("Filtered fourier transform of flower01.png");


subplot(3,3,4);
imshow(I2);
title("flower02.png");
subplot(3,3,5);
imshow(F2);
title("Fourier transform of flower02.png");
subplot(3,3,6);
imshow(H2);
title("Filtered fourier transform of flower02.png");

subplot(3,3,7);
```

```
imshow(G);
title("G");
subplot(3,3,8);
imshow(H);
title("H");
subplot(3,3,9);
bar([E1 E2]);
title("Energy of both images");

print("Quality.eps","-depsc");
```

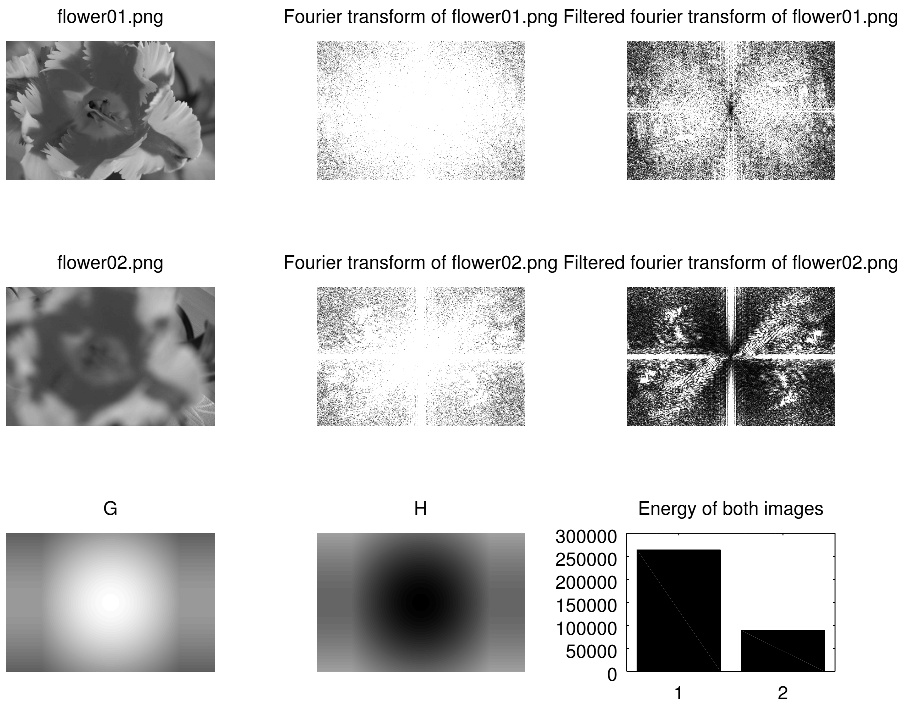


Abbildung 5: Output of the Matlab script