Computer Vision I

Tim Luchterhand, Paul Nykiel (Group 17)

21. Mai 2018

1 Filter Algebra

1.1

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

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");
```

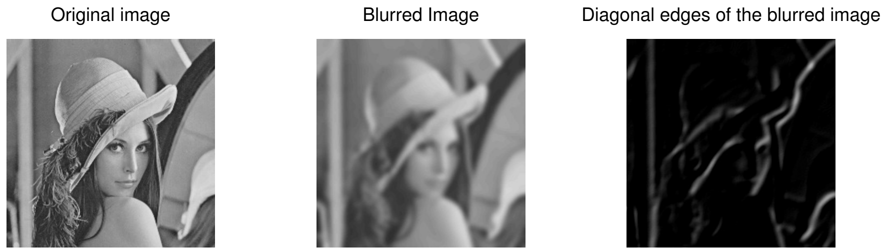

Abbildung 1: Output of the Matlab script

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

```
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");
```

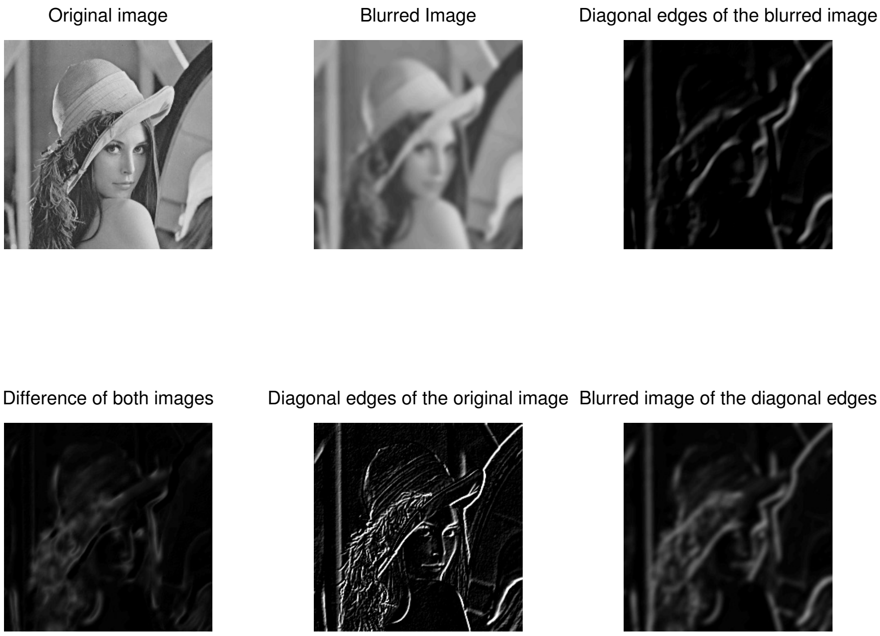

Abbildung 2: Output of the second part of the Matlab script

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

```
end;
S ./= sqrt(N);
```

2.2

The uth 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");
```

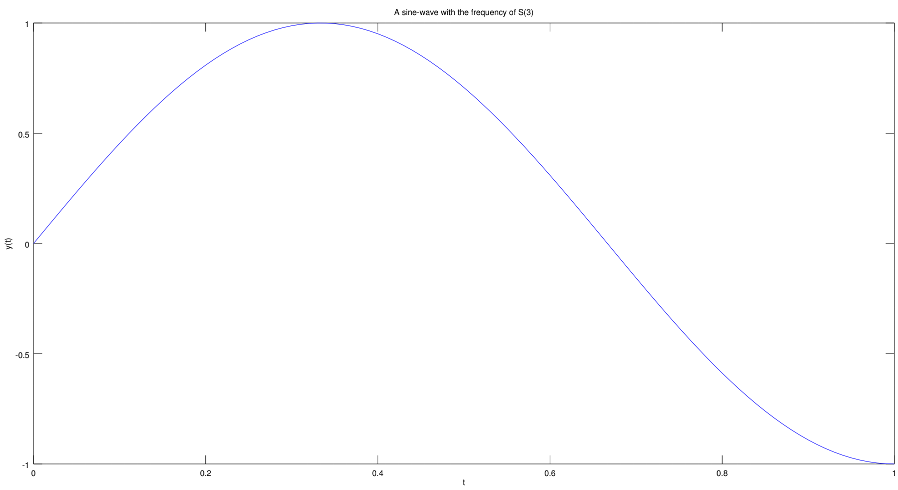

Abbildung 3: Output of the second part of the Matlab script

2.3

```
%% Part c
sr = zeros(size(ti));
for u=1:N
```

```
    for t=0:(N-1)
        sr(u) += 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");
```

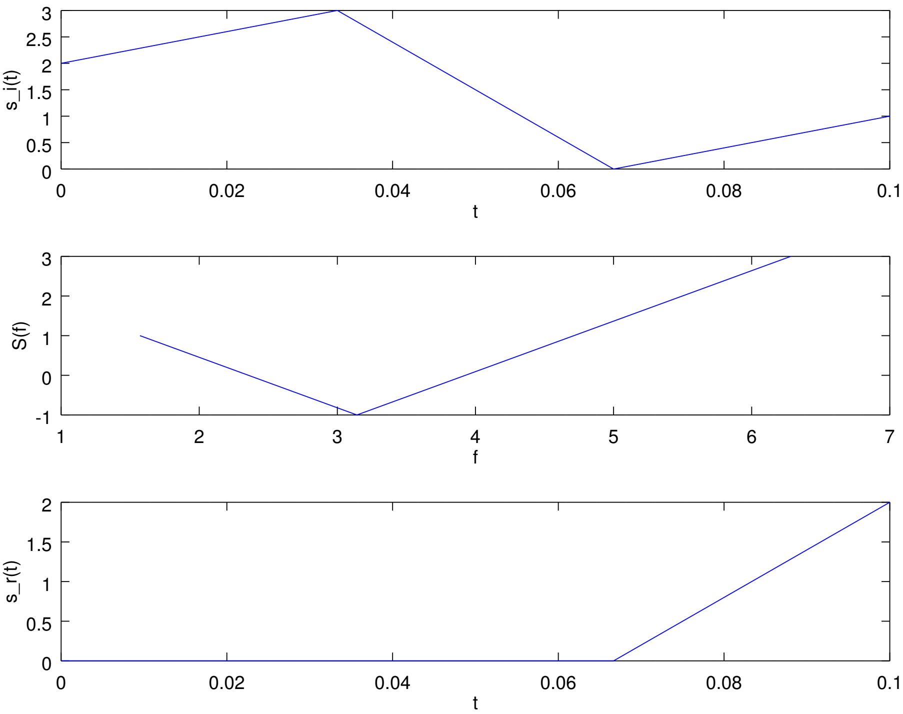

Abbildung 4: Output of the last part of the Matlab script

3 Fourier Transform for Image Quality Assessment

3.1