

# Computer Vision I

Tim Luchterhand, Paul Nykiel (Group 17)

5. Juni 2018

## 1 Histogram Calculation

1. Matlab-Funktion:

```
1 function H = myHistogram(im)
2     H = zeros(1,256);
3     [width, height] = size(im);
4     for x = 1:width
5         for y = 1:height
6             intensity = im(x,y);
7             assert(intensity >= 0 & intensity <= 255, "Not
              all matrix elements are between 0 and 255");
8             H(intensity+1) = H(intensity+1) + 1;
9         end
10    end
11    H ./= 1/(width*height);
12 end
```

2. Generate the plots:

```
1 % Read the images
2 fruitsA = imread('images/fruitsA.png');
3 fruitsB = imread('images/fruitsB.png');
4
5 % Calculate the histograms
6 histA = myHistogram(fruitsA);
7 histB = myHistogram(fruitsB);
8
9 % Plot the histograms
10 figure();
11
12 subplot(2,2,1);
```

```

13 imshow(fruitsA);
14 title('fruitsA.png');
15
16 subplot(2,2,2);
17 plot(0:255, histA);
18 axis([0 255 0 0.04]);
19 title('Histogramm for fruitsA.png');
20 xlabel('Intensity');
21 ylabel('Probability');
22
23 subplot(2,2,3);
24 imshow(fruitsB);
25 title('fruitsB.png');
26
27 subplot(2,2,4);
28 plot(0:255, histB);
29 axis([0 255 0 0.05]);
30 title('Histogramm for fruitsB.png');
31 xlabel('Intensity');
32 ylabel('Probability');
33
34 print('Histograms', '-depsc')

```

3.

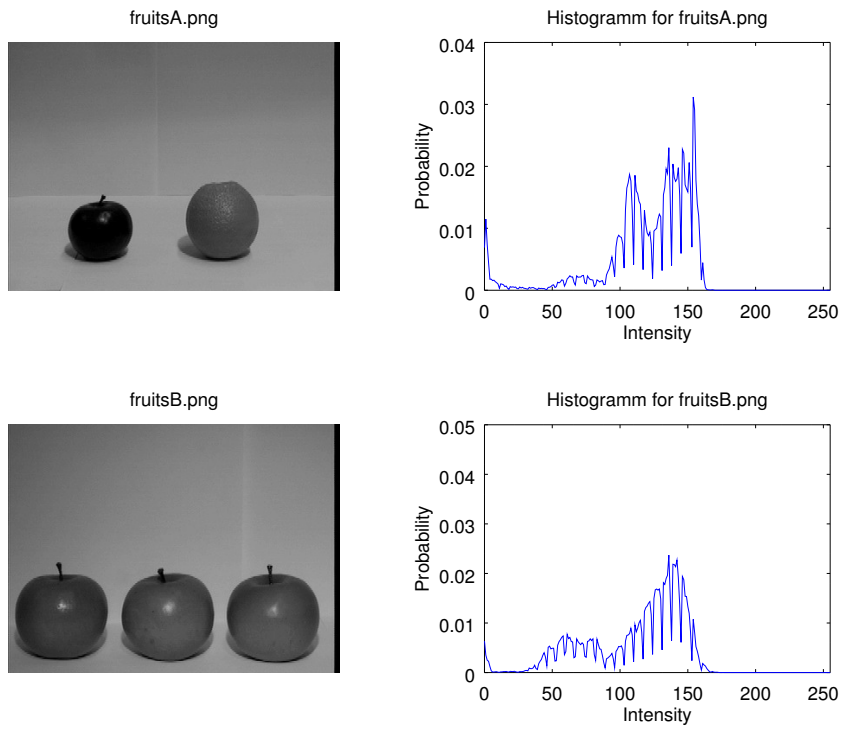


Abbildung 1: Plot of the histograms

4.

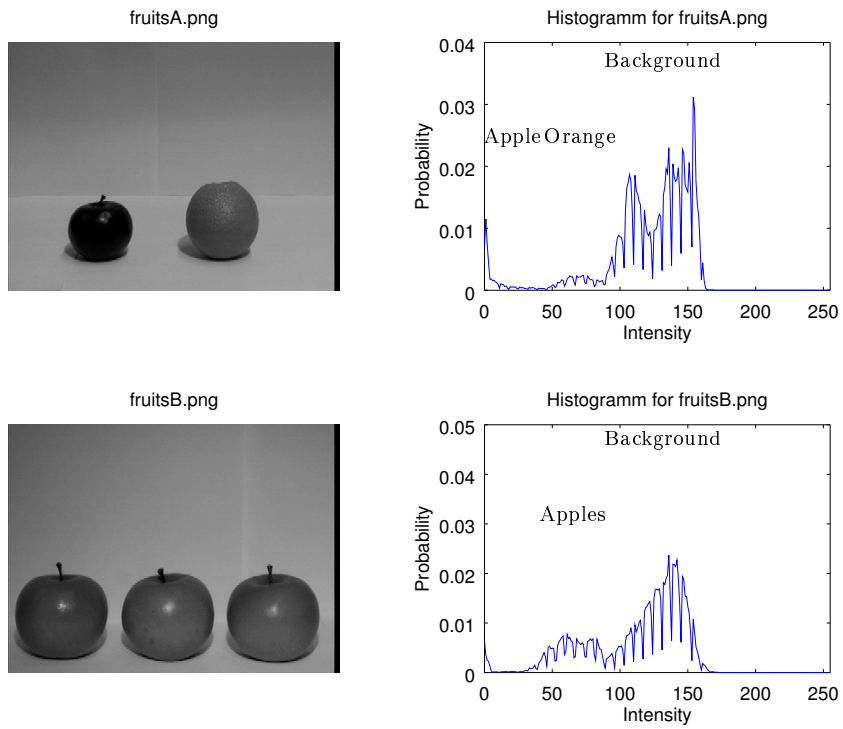


Abbildung 2: Plot of the histograms

5.

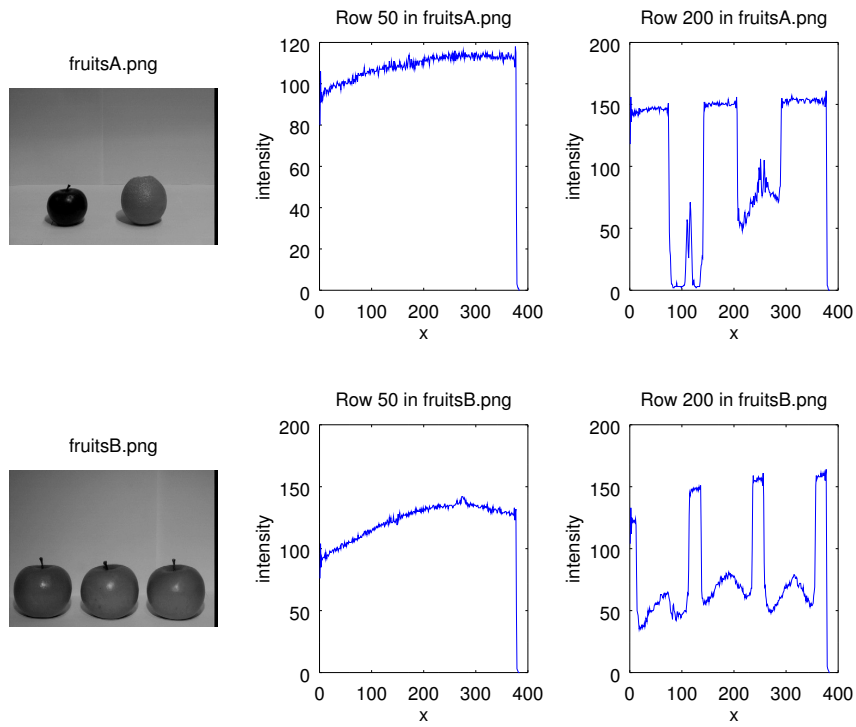


Abbildung 3: Plot of the rows

## 2 Local weighting

1.

$$\begin{pmatrix} \cdot & \cdot & \cdot \\ 1 \cdot 1 + 1 \cdot 1 & 1 \cdot 1 & -1 \cdot 1 + -1 \cdot 1 + 1 \cdot 1 & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{pmatrix} = \begin{pmatrix} \cdot & \cdot & \cdot & \cdot \\ 2 & 1 & -1 & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{pmatrix}$$

2. Negative derivative in **red**, positive in **green**, zero in **blue**.

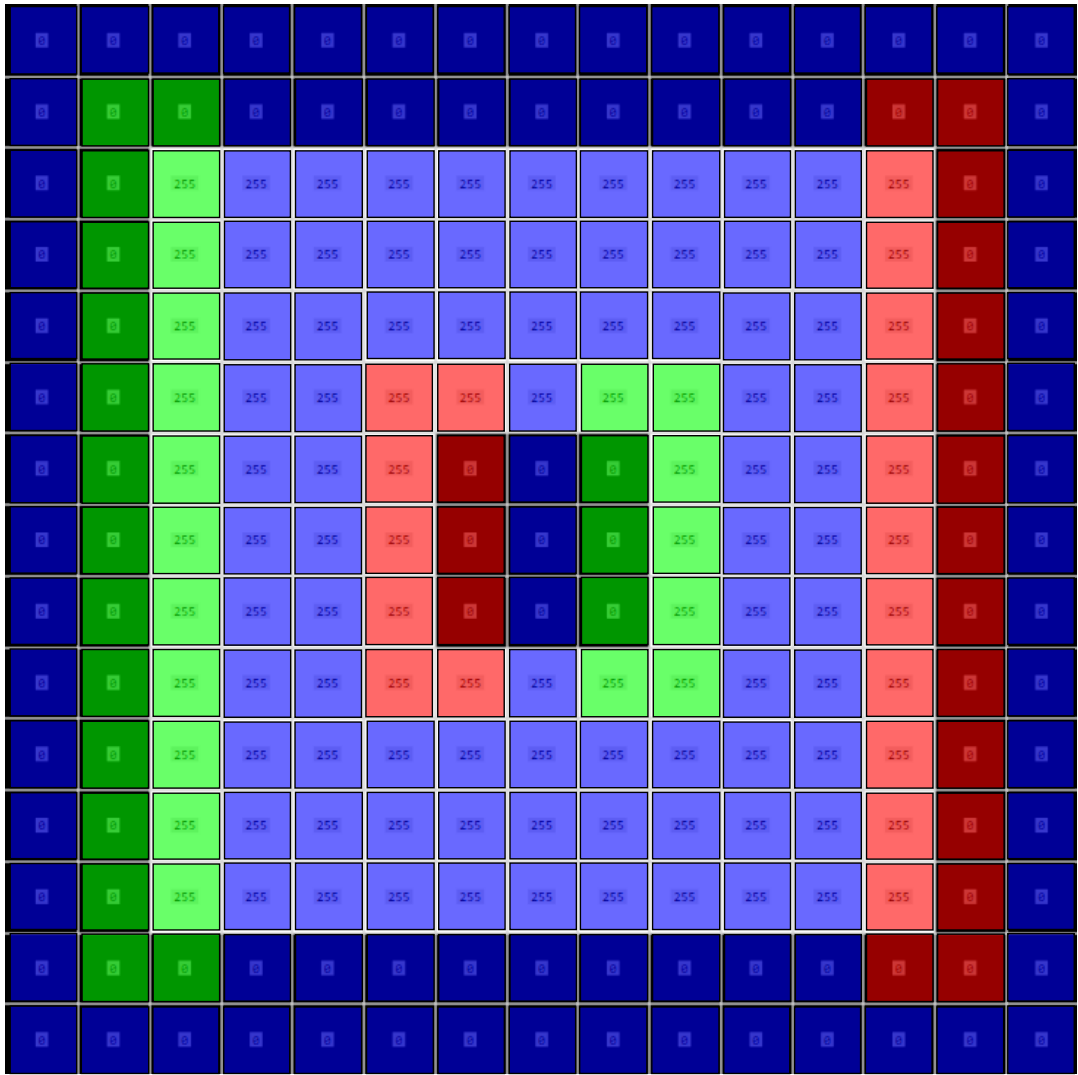


Abbildung 4: Local derivative using  $G_x$

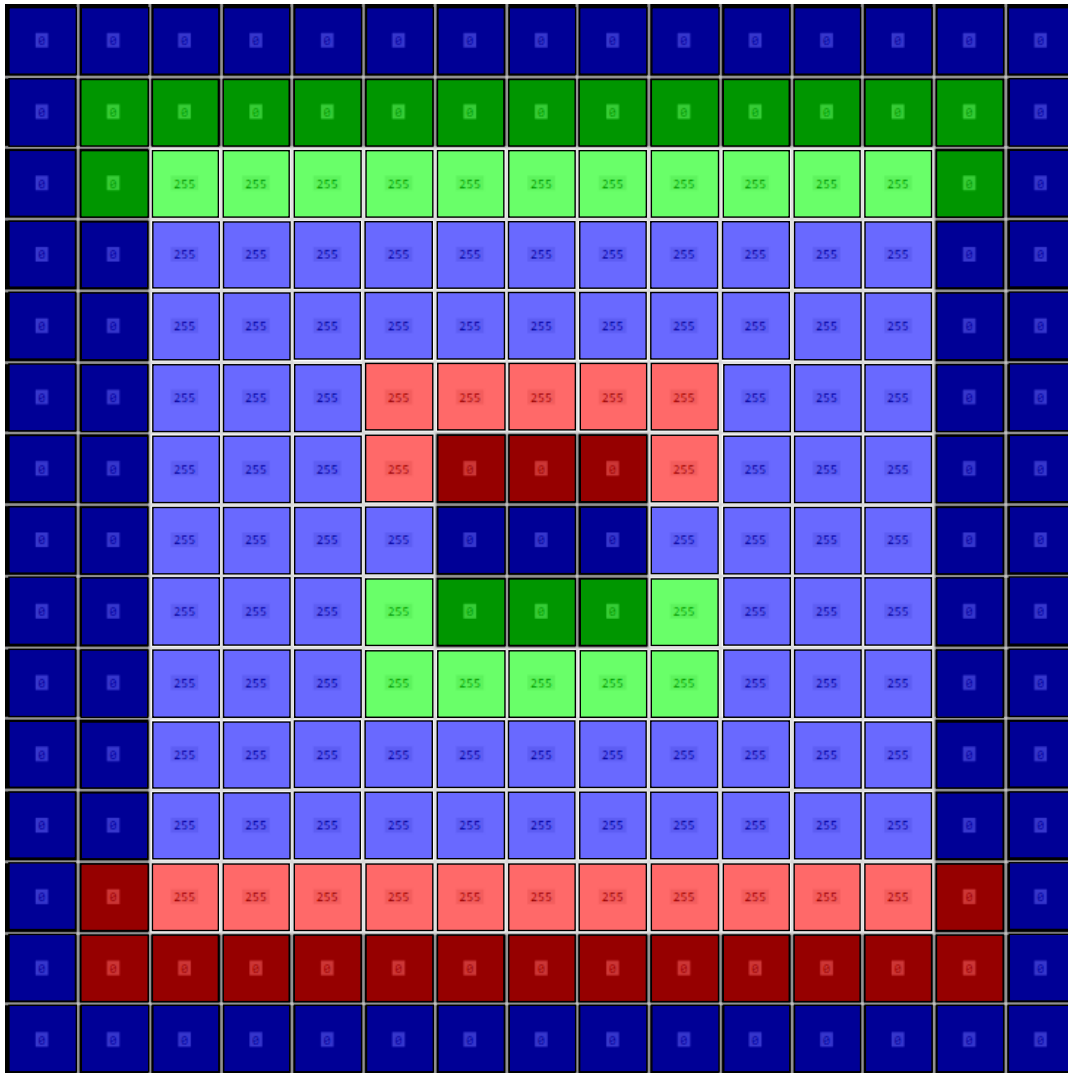


Abbildung 5: Local derivative using  $G_y$

3.

```

1 lena = imread('images/lena.tif');
2 lenaNoise = imread('images/lenaNoise.tif');
3
4 B = ones(3,3) / 9;
5
6 subplot(2,2,1);
7 imshow(lena);

```

```

8  title( 'lena.tif' );
9
10 subplot(2,2,2);
11 imshow( imfilter( lena ,B) );
12 title( 'Filtered lena.tif' );
13
14 subplot(2,2,3);
15 imshow( lenaNoise );
16 title( 'lenaNoise.tif' );
17
18 subplot(2,2,4);
19 imshow( imfilter( lenaNoise ,B) );
20 title( 'Filtered lenaNoise.tif' );
21
22 print( 'BoxFilter ', '-dep' );

```

This filter can be used to smooth an image and therefor reduce noise. It is sort of a blurring filter.

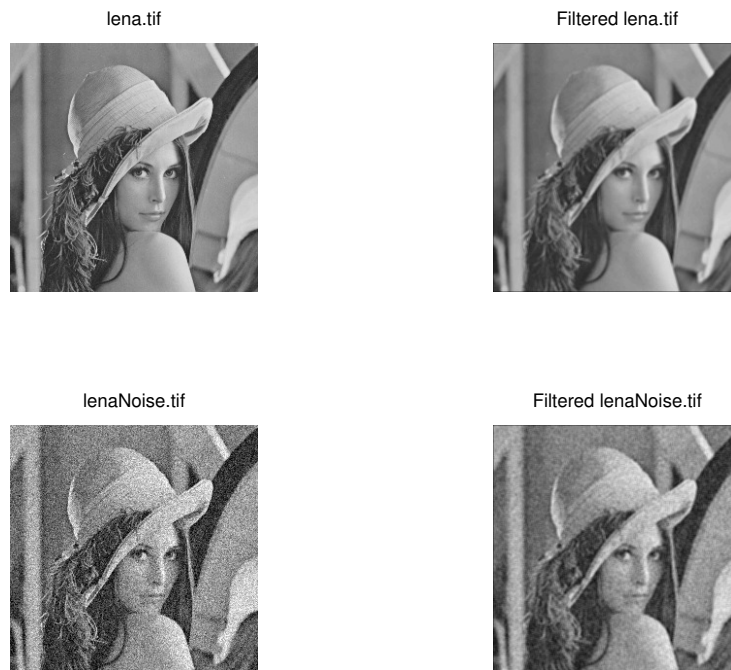


Abbildung 6: Original and filtered images