NTNU
Norwegian University of
Science and Technology

Digital Image Processing Assignment 1

Marius Blom

Sondre Jensen

2017

TDT4195 - Visual Computing Fundamentals

Table of Contents

List of Tables

LIST OF FIGURES

1 Theory Questions

1.1 Histogram equalisation

Histogram equalisation is a technique for adjusting image intensities to enhance contrast.

1.1.1 How can we see that an image has low contrast when looking at its histogram?

The frequencies are concentrated on the mid range intensities.

1.1.2 What effect does τ_{heq} have when applied on an image?

Improving the contrast by uniformly distributed difference between dark and light values.

1.1.3 What happens when histogram equalisation is applied multiple times on the same image in sequence?

Histogram equalisation enhance the contrast. Applying histogram equalisation multiplied times does not change anything since the intensities is already at the desired values.

1.2 Perform histogram equalisation

Table 1.1: Original image

12	3	1	9
3	0	4	3
2	6	15	2

Table 1.2: Histogram equalisation

n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
f_n	$\frac{1}{12}$	$\frac{1}{12}$	$\frac{2}{12}$	$\frac{3}{12}$	$\frac{1}{12}$	$\frac{0}{12}$	$\frac{1}{12}$	$\frac{0}{12}$	$\frac{0}{12}$	$\frac{1}{12}$	$\frac{0}{12}$	$\frac{0}{12}$	$\frac{1}{12}$	$\frac{0}{12}$	$\frac{0}{12}$	$\frac{1}{12}$
F_n	$\frac{1}{12}$	$\frac{2}{12}$	$\frac{4}{12}$	$\frac{7}{12}$	$\frac{8}{12}$	$\frac{8}{12}$	$\frac{9}{12}$	$\frac{9}{12}$	$\frac{9}{12}$	$\frac{10}{12}$	$\frac{10}{12}$	$\frac{10}{12}$	$\frac{11}{12}$	$\frac{11}{12}$	$\frac{11}{12}$	$\frac{12}{12}$
$F_n * 15$	$\frac{15}{12}$	$\frac{30}{12}$	$\frac{60}{12}$	$\frac{105}{12}$	$\frac{120}{12}$	$\frac{120}{12}$	$\frac{135}{12}$	$\frac{135}{12}$	$\frac{135}{12}$	$\frac{150}{12}$	$\frac{150}{12}$	$\frac{150}{12}$	$\frac{165}{12}$	$\frac{165}{12}$	$\frac{165}{12}$	$\frac{180}{12}$
floor	1	2	5	8	10	10	11	11	11	12	12	12	13	13	13	15

Table 1.3: Equalised image with different intensity on each pixel

13	8	2	12
8	1	10	8
5	11	15	5

1.3 Convolution operator being associative

The "Associative Laws" say that it doesn't matter how we group the numbers when we add or multiply. This property is beneficial for convolution since the grouping of the numbers does not affect the result.

1.4 A MxM kernel

$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

(a) left figure

$$\begin{bmatrix}1 & 1 & 1\\ 1 & -8 & 1\\ 1 & 1 & 1\end{bmatrix}$$

(b) right figure

Figure 1.1: Combined figure

1.4.1 Determine if the convolution kernel are separable by computing their rank

matrix a

$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} \overset{R3-R1}{\sim} \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 0 & 0 & 0\end{bmatrix}$$
$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} \overset{\frac{R2}{2}}{\sim} \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 1 & 2 & 1\\ 0 & 0 & 0\end{bmatrix}$$
$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} \overset{R2-R1}{\sim} \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix}$$

Which implies that $rank(a) = 1$. Hence the matrix is seperable. We thereby have that

$$MxM = \frac{1}{16}\begin{bmatrix}1 & 2 & 1\end{bmatrix}\begin{bmatrix}1\\ 2\\ 1\end{bmatrix} \Rightarrow \frac{1}{4}\begin{bmatrix}1 & 2 & 1\end{bmatrix}\frac{1}{4}\begin{bmatrix}1\\ 2\\ 1\end{bmatrix}$$

matrix b

$$\begin{bmatrix}1 & 1 & 1\\ 1 & -8 & 1\\ 1 & 1 & 1\end{bmatrix} \overset{R3-R1}{\sim} \begin{bmatrix}1 & 1 & 1\\ 1 & -8 & 1\\ 0 & 0 & 0\end{bmatrix}$$
$$\begin{bmatrix}1 & 1 & 1\\ 1 & -8 & 1\\ 1 & 1 & 1\end{bmatrix} \overset{R2-R1}{\sim} \begin{bmatrix}1 & 1 & 1\\ 0 & -9 & 0\\ 0 & 0 & 0\end{bmatrix}$$
$$\begin{bmatrix}1 & 1 & 1\\ 1 & -8 & 1\\ 1 & 1 & 1\end{bmatrix} \overset{\frac{R2}{-9}}{\sim} \begin{bmatrix}1 & 1 & 1\\ 0 & 1 & 0\\ 0 & 0 & 0\end{bmatrix}$$
$$\begin{bmatrix}1 & 1 & 1\\ 1 & -8 & 1\\ 1 & 1 & 1\end{bmatrix} \overset{R1-R2}{\sim} \begin{bmatrix}1 & 0 & 1\\ 0 & 1 & 0\\ 0 & 0 & 0\end{bmatrix}$$

The matrix is not separable since $rank(b) = 2$

1.4.2 Why is it helpful to have convolution kernels that are separable?

We are able to increase the computation speed by separating the matrix into vectors.

2 Greyscale Conversion

2.1 Implement two functions that manually convert a RGB colour image to a greyscale representation

$$grey_{i,j} = \frac{R_{i,j} + G_{i,j} + B_{i,j}}{3} \tag{2.1}$$

```
def average(rgb):
        for i in range(len(rgb)):
                for j in range(len(rgb[i])):
                        grey = 0
                        grey += rgb[i, j, 0]
                        grey += rgb[i, j, 1]
                        grey += rgb[i, j, 2]
                        avgColor = grey/3
                        rgb[i, j] = [avgColor, avgColor, avgColor]
        return rgb
```

$$grey_{i,j} = 0.2126R_{i,j} + 0.7152G_{i,j} + 0.0722B_{i,j} \tag{2.2}$$

```
def weightedAverage(rgb):
        for i in range(len(rgb)):
                for j in range(len(rgb[i])):
                        grey = 0
                        luminance = [0.2126, 0.7152, 0.0722]
                        grey += rgb[i, j, 0]*luminance[0]
                        grey += rgb[i, j, 1]*luminance[1]
                        grey += rgb[i, j, 2]*luminance[2]
                        rgb[i, j] = [grey, grey, grey]
        return rgb
```

2.2 Apply functions on colour images

We can see clearly from figure 2.1 that the averaging method lack some shades of grey. The averaging method is having trouble with the color red and the color green, since they seem to get the same shade of grey in the picture.

Figure 2.1: Averaging method, luminance-preserving method and original image

Figure 2.2: Averaging method, luminance-preserving method and original image

3 Intensity Transformations

3.1 Intensity transformation greyscale

```
#       intensity transformations
def intensity(grayscale):
#       loop through the colors
        for i in range(len(grayscale)):
                for j in range(len(grayscale[i])):
                #       save original image
                        p = grayscale[i, j]
                #       p_k is the highest value a pixel can have
                        p_k = 255
                #       use formula T(p)=p_k-p and update the colors
                        grayscale[i, j] = p_k - p
        return grayscale
```

3.1.1 Explain what this transformation does

The transformation inverts the colors of the image.

3.1.2 Apply the function on an image of your choice and show the result

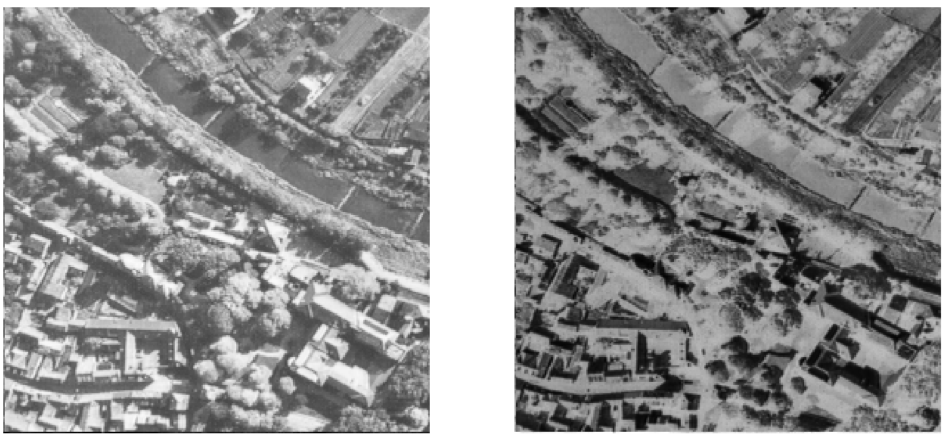

Figure 3.1: subplot of Intensity

3.2 Gamma transformation greyscale

```
#       gamma transformations
def gammaTransform(rgb,gammaValues):
        for i in range(len(rgb)):
                for j in range(len(rgb[i])):
                #       normalize the image
                        p = rgb[i, j]/255
                        rgb[i, j] = (p**gammaValues)*255
        return rgb
```

3.2.1 Explain what happens with the image intensity values when $\gamma > 1$ and $\gamma < 1$

The image gets lighter when $\gamma < 1$

3.2.2 Different γ values on a greyscale image

```
def subplotImage(filepath):
        _, ax = plt.subplots(1, 3, figsize=(16, 8))
#       store gamma values in array
        gammaValues = [.2, .5, .9]
#       read from array and create plot for each value
        for i in range(len(gammaValues)):
                image = misc.imread(filepath)
                gammaTransformed = gammaTransform(image,gammaValues[i])
                ax[i].imshow(gammaTransformed, cmap=plt.cm.gray)
                ax[i].set_axis_off()
        plt.show()
        return None
```

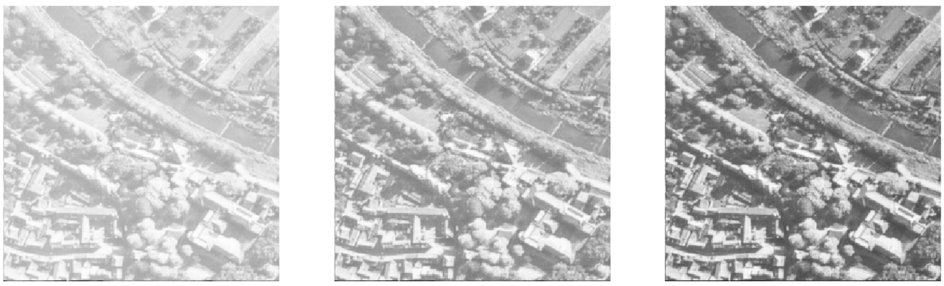

Figure 3.2: gammaValues from .2, .5, .9

Figure 3.3: gammaValues from .2, .5, .9

4 SPATIAL CONVOLUTION

4.1 Arbitrary linear convolution kernel

4.2 Convolution function

4.2.1 Convolve a greyscale image

Figure 4.1: Convolve a greyscale

4.2.2 Convolve a colour image

Figure 4.2: Convolve a colour

4.3 Use convolution implementation

4.3.1 Use the convolution kernels to approximate the horizontal and vertical gradient of a greyscale image

4.3.2 Compute the magnitude of the gradients $|\nabla|$

4.3.3 Explain what $|\nabla|$ tell us about the image