Image Processing and Computer Vision

Dr. Pascal Peter — SS 2023

Mathematical Image Analysis Group — Saarland University

Example Solutions for Classroom Work Assignment C1

Problem C1.1 (Continuous Convolution)

We have

$$(K * H)(x) = \int_{-\infty}^{\infty} K(t)H(x-t)\,\mathrm{d}t = \int_{x}^{\infty} K(t)\,\mathrm{d}t = \int_{x}^{1} K(t)\,\mathrm{d}t\,,$$

where the boundaries of the integral follow from $H(x-t)=1$ for $x-t<0 \Leftrightarrow x<t$, and as $K(t)=0$ for $t>1$. We have to find a solution in dependence of x. As $K(t)$ is a piecewise-defined function, we have to consider four cases. For $x<-1$, by splitting the integral in the different subdomains of $K(t)$, we obtain

$$\begin{aligned}(K * H)(x) &= \int_x^1 K(t)\,\mathrm{d}t = \underbrace{\int_x^{-1} K(t)\,\mathrm{d}t}_{=0} + \int_{-1}^{0} 1+t\,\mathrm{d}t + \int_0^1 1-t\,\mathrm{d}t\\ &= \left[t+\frac{1}{2}t^2\right]_{-1}^{0} + \left[t-\frac{1}{2}t^2\right]_0^1 = 1-\frac{1}{2}+1-\frac{1}{2} = 1\end{aligned}$$

For the case $-1 \leqslant x < 0$, we obtain

$$(K * H)(x) = \int_x^0 1+t\,\mathrm{d}t + \int_0^1 1-t\,\mathrm{d}t = \left[t+\frac{1}{2}t^2\right]_x^0 + \frac{1}{2} = \frac{1}{2} - x - \frac{1}{2}x^2$$

For $0 \leqslant x < 1$ we obtain

$$(K * H)(x) = \int_x^1 1-t\,\mathrm{d}t = \left[t-\frac{1}{2}t^2\right]_x^1 = \frac{1}{2} - x + \frac{1}{2}x^2$$

For $x \geqslant 1$ we obtain

$$(K * H)(x) = \int_x^1 K(t)\,\mathrm{d}t = -\underbrace{\int_1^x K(t)\,\mathrm{d}t}_{=0} = 0$$

This means that we obtain

$$(K * H)\,(x) = \begin{cases} 1, & x < -1\\ \frac{1}{2} - x - \frac{1}{2}x^2, & -1 \leqslant x < 0\\ \frac{1}{2} - x + \frac{1}{2}x^2, & 0 \leqslant x < 1\\ 0, & x \geqslant 1 \end{cases}$$

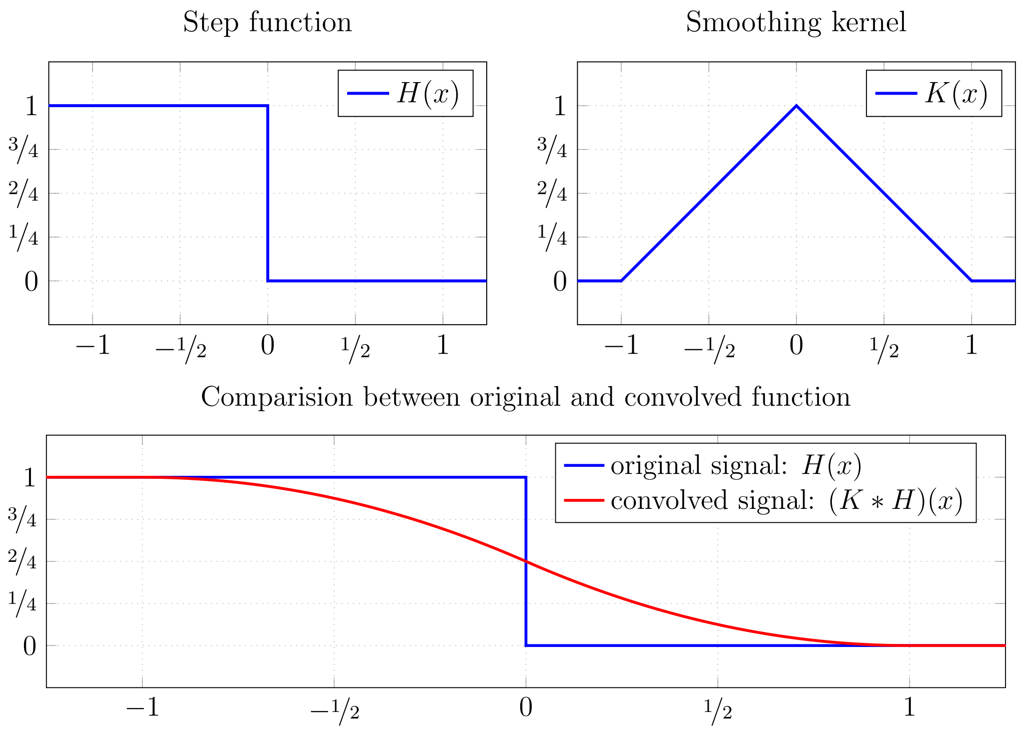


Figure 1: The function H and its convolution with kernel K.

As we can see in Fig. 1 the convolution with K resulted in a much smoother signal. The sharp boundary present in H at $x = 0$ has completely vanished.

Problem C1.2 (Proof of the Box-Muller Algorithm)

The following theorem shows how one can use uniformly distributed random numbers to generate random variables obeying any distribution. The Box-Muller Algorithm is basically just a clever application of this theorem.

Transformation/Inversion Rule: *Let F be a continuous distribution function on $\mathbb{R}$ with inverse F^{-1}. If U is a uniform $[0, 1]$ distributed random variable, then the random variable $X = F^{-1}(U)$ has distribution function F. Conversely, if X has distribution function F, then $F(X)$ is uniformly distributed on $[0, 1]$.*
The proof of this theorem is quite simple, however, it requires elementary knowledge in measure and probability theory and thus we skip it at this place and refer instead to the excellent Book of Luc Devroye, *Non-Uniform*

Random Variate Generation[1], which describes the complete theory of random number generation in every detail.

Note that the distribution function of a random variable X is defined in terms of probability and probability density function as

$$F_X(x) := P\left(X \leqslant x\right) := \int_{-\infty}^{x} p(t)\,\mathrm{d}t$$

From this it also follows that

$$\frac{\partial}{\partial x} F_X\left(x\right) = p(x)$$

although we will not need that relation in this exercise. The above theorem essentially states that it is enough to find an expression for F^{-1}. If an evaluation of $F^{-1}(U)$ yields the values stated in the Box-Muller algorithm, we have proven that it generates normally distributed random numbers. Note that certain fundamental theorems from measure and probability theory guarantee that this inverse function always exists and is unique. However, it may not always have an analytical description. The one dimensional normal distribution function for example is given by

$$F(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{t^2}{2}\right) \mathrm{d}t = \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{x}{\sqrt{2}}\right)\right)$$

where erf is the Gaussian error function, which cannot be described analytically. Further, its inverse cannot be described in a closed form either. Thus, a straightforward application of the above theorem is not possible. One could of course use numerical methods to invert the distribution function, however, it would be computationally too expensive to deploy such an approach on a large scale. In 1958, George Edward Pelham Box and Mervin Edgar Muller suggested to consider the normal distribution in $\mathbb{R}^2$ and to generate pairs of normal distributed random numbers instead of single ones.[2]

The normal distribution in $\mathbb{R}^2$ has the density function

$$p(x,y) = \frac{1}{2\pi\sigma^2} \exp\left(\frac{-(x-\mu_1)^2 - (y-\mu_2)^2}{2\sigma^2}\right)$$

For the standard normal distribution, i.e. $\mu = 0$ and $\sigma = 1$, this simplifies to:

$$p(x,y) = \frac{1}{2\pi} \exp\left(-\frac{x^2+y^2}{2}\right) \tag{1}$$

[1]Available for free at `http://luc.devroye.org/handbooksimulation1.pdf`

[2]A Note on the Generation of Random Normal Deviates. The Annals of Mathematical Statistics (1958), Vol. 29, No. 2 pp. 610–611.

Note that this function only depends on the distance to the origin and not on the direction. This little fact will be crucial in the following. We now need two further results from probability theory.

Theorem *If two random variables X and Y are independent, their common probability density function is the product of the individual probability density functions.*

$$p_{(X,Y)}(x,y) = p_X(x)p_Y(y)$$

Theorem *If the random variable pairs (X,Y) and (R,Q) have probability density functions $p_{(X,Y)}$ and $p_{(R,Q)}$, and if there exists a C^1 diffeomorphism g with Jacobian determinant different from 0, then the probability density functions are linked by the relation*

$$p_{(R,Q)} = |\det(\boldsymbol{J}_g)| p_{(X,Y)}$$

We now want to apply the two previous theorems and chose as diffeomorphism the polar coordinate transform (polar coordinates are always useful for functions that only depend on the distance from the origin):

$$\begin{aligned} x &= r\cos\varphi =: g_1(r,\varphi) \\ y &= r\sin\varphi =: g_2(r,\varphi). \end{aligned}$$

The Jacobian of the polar coordinate transformation can be computed as

$$\boldsymbol{J}_g(r,\varphi) = \begin{pmatrix} \dfrac{\partial g_1}{\partial r} & \dfrac{\partial g_1}{\partial \varphi} \\ \dfrac{\partial g_2}{\partial r} & \dfrac{\partial g_2}{\partial \varphi} \end{pmatrix} = \begin{pmatrix} \cos(\varphi) & -r\sin\varphi \\ \sin(\varphi) & r\cos\varphi \end{pmatrix}$$

and its determinant is given by

$$\det(\boldsymbol{J}_g(r,\varphi)) = r\cos^2\varphi + r\sin^2\varphi = r$$

From this it follows that for independent normal distributed random variables X and Y we have

$$p_{(R,Q)}(r,\varphi) = p_{(X,Y)}(x,y)\det(\boldsymbol{J}_g) = p_X(x)p_Y(y)\det(\boldsymbol{J}_g) = \frac{1}{2\pi}\exp(-\frac{r^2}{2})r \qquad (*)$$

where the random variables R and Q are the corresponding representations in polar coordinates. If the pair (X,Y) of random variables is independent, the same also holds for the pair $(R,Q) := (g_1^{-1}(X,Y), g_2^{-1}(X,Y))$. With equation

(∗) it follows we have the density function $p_R(r) = r\exp\left(-\frac{r^2}{2}\right)$ with domain $[0,\infty)$ for R, and for Q, we have the density function $p_Q(\varphi) = \frac{1}{2\pi}$ with domain $[0, 2\pi)$. This means, that at this point we have reduced our initial problem down to generating random variables R and Q of the latter two distributions. These can be transformed back into cartesian coordinates to obtain the desired random variables with normal distribution.

Since Q is uniformly distributed on $[0, 2\pi)$, it suffices to generate uniformly distributed numbers on $[0, 1]$ and to multiply them by 2π. As for R we may apply our first theorem. We have

$$F(t) = \int_0^t r\exp(-\frac{r^2}{2})\mathrm{d}r = \left[-\exp(-\frac{r^2}{2})\right]_0^t = 1 - \exp(-\frac{t^2}{2})$$

Now, solving the equation $F(R) = U$ for R is quite easy. Its solution is given by

$$R = \sqrt{-2\ln(1-U)}$$

Since U is uniformly distributed on $[0, 1]$, the same also holds for $1 - U$ and thus we can further write $R = \sqrt{-2\ln(U)}$. Finally, this means that we have the following algorithm:

1. Generate U, V uniformly distributed on $[0, 1]$.
2. Compute
$$\begin{aligned} R &= \sqrt{-2\ln(U)} \\ Q &= 2\pi V \end{aligned}$$
3. Use the polar coordinate transform to go back to cartesian coordinates
$$\begin{aligned} X &= R\cos(Q) = \sqrt{-2\ln(U)}\cos(2\pi V) \\ Y &= R\sin(Q) = \sqrt{-2\ln(U)}\sin(2\pi V) \end{aligned}$$

Remark: For random variables $\tilde{X}, \tilde{Y}$ of a general normal distribution with arbitrary σ and μ we compute

$$\begin{aligned} \tilde{X} &= \mu + \sigma X \\ \tilde{Y} &= \mu + \sigma Y \end{aligned}$$

Example Solutions for Homework Assignment H1

Problem H1.1 (Peak-Signal-to-Noise Ratio)

We first review the definition of the PSNR:

$$\mathrm{PSNR}(f,g) := 10\,\log_{10}\left(\frac{255^2}{\mathrm{MSE}(f,g)}\right)$$

where $\mathrm{MSE}(f,g)$ is the mean squared error of the original image and the noisy image.

(a) We have

$$\mathrm{PSNR}(f,g) = 0 \Leftrightarrow \log_{10}\left(\frac{255^2}{\mathrm{MSE}(f,g)}\right) = 0 \Leftrightarrow \mathrm{MSE}(f,g) = 255^2$$

This means that f and g have the maximal possible distance at all pixels.

(b) Here we use the fact that

$$\log_{10}\left(\frac{a}{b}\right) = \log_{10}(a) - \log_{10}(b)\,.$$

Let us assume that u is a restored version of the noisy image f such that $\mathrm{PSNR}(u,g) = \mathrm{PSNR}(f,g) + 30$. We calculate that

$$\begin{aligned}
&\mathrm{PSNR}(u,g) = \mathrm{PSNR}(f,g) + 30\\
&\log_{10}\left(\frac{255^2}{\mathrm{MSE}(u,g)}\right) = \log_{10}\left(\frac{255^2}{\mathrm{MSE}(f,g)}\right) + 3\\
&\log_{10}(\mathrm{MSE}(u,g)) = \log_{10}(\mathrm{MSE}(f,g)) - 3\\
&\log_{10}(\mathrm{MSE}(u,g)) = \log_{10}(\mathrm{MSE}(f,g)) - \log_{10}(1000)\\
&\mathrm{MSE}(u,g) = \frac{\mathrm{MSE}(f,g)}{1000}
\end{aligned}$$

This means that filtering the signal reduced the MSE by a factor 1000.

(c) When $n \to 0$, then $f \to g$. Let us have a look at the limit: When $\mathrm{MSE}(f,g)$ becomes infinitely small, the above mentioned fraction becomes infinitely large. Therefore,

$$\lim_{f\to g}\left(10\,\log_{10}\left(\frac{255^2}{\mathrm{MSE}(f,g)}\right)\right) \to \infty\,.$$

This expresses the fact, that the quality cannot become better.

Problem H1.2 (Discrete Convolution)

We observe that with the given convolution kernel f one obtains for any discrete signal $g = (g_i)_{i\in\mathbb{Z}}$ the result

$$(g * f)_i = \sum_{j=-\infty}^{\infty} f_{i-j}\, g_j = \frac{1}{2}\, g_i + \frac{1}{2}\, g_{i-1}\ .$$

Applying this iteratively leads to

$$(f * f)_i = \begin{cases} 0, & i < 0 \text{ or } i > 2, \\ \frac{1}{4}, & i = 0 \text{ or } i = 2, \\ \frac{2}{4}, & i = 1 \end{cases}$$

$$(f * f * f)_i = \begin{cases} 0, & i < 0 \text{ or } i > 3, \\ \frac{1}{8}, & i = 0 \text{ or } i = 3, \\ \frac{3}{8}, & i = 1 \text{ or } i = 2 \end{cases}$$

$$(f * f * f * f)_i = \begin{cases} 0, & i < 0 \text{ or } i > 4, \\ \frac{1}{16}, & i = 0 \text{ or } i = 4, \\ \frac{4}{16}, & i = 1 \text{ or } i = 3, \\ \frac{6}{16}, & i = 2\ . \end{cases}$$

Problem H1.3 (Continuous Convolution)

For the given convolution kernel f and any continuous signal $g(x)$ we have

$$(g * f)(x) = \int_{-\infty}^{\infty} f(x-z)\, g(z)\, \mathrm{d}z \ = \int_{x-1}^{x+1} \frac{1}{2}\, g(z)\, \mathrm{d}z\ ,$$

as $f(x-z) = \frac{1}{2}$ for $-1 \leqslant x - z \leqslant 1 \Leftrightarrow x - 1 \leqslant z \leqslant x + 1$.
Note that due to commutativity, it doesn't matter in which order f and g appear in the convolution term.

If we calculate $(f * f)$, we notice that $f(z) = 0$ for $z \leqslant -1$ and $z \geqslant 1$. Therefore we can observe that the upper bound of the integral is smaller or equal than -1 if $x \leqslant -2$, and the lower bound is bigger than 1 if $x > 2$. Furthermore, we distinguish between two more cases. For $-2 < x \leqslant 0$, we obtain

$$(f * f)(x) = \underbrace{\int_{x-1}^{-1} \frac{1}{2} f(z)\, \mathrm{d}z}_{=0} + \int_{-1}^{x+1} \frac{1}{2} f(z)\, \mathrm{d}z$$

and for $0 < x \leqslant 2$, we obtain

$$(f * f)(x) = \int_{x-1}^{1} \frac{1}{2} f(z)\,\mathrm{d}z + \underbrace{\int_{1}^{x+1} \frac{1}{2} f(z)\,\mathrm{d}z}_{=0}\,,$$

where the integral is split at the boundaries of the subdomains of the piecewise-defined function f. For $f * f$ we therefore obtain

$$\begin{aligned}(f * f)(x) &= \int\limits_{x-1}^{x+1} \frac{1}{2}\, f(z)\,\mathrm{d}z \\ &= \begin{cases} 0, & x \le -2, \\ \int_{-1}^{x+1} \frac{1}{4}\,\mathrm{d}z, & -2 < x \le 0, \\ \int_{x-1}^{1} \frac{1}{4}\,\mathrm{d}z, & 0 < x \le 2, \\ 0, & x > 2 \end{cases}\end{aligned}$$

and thus

$$(f * f)(x) = \begin{cases} 0, & x \le -2, \\ \frac{2+x}{4}, & -2 < x \le 0, \\ \frac{2-x}{4}, & 0 < x \le 2, \\ 0, & x > 2\,. \end{cases}$$

For $f * f * f$ we find

$$\begin{aligned}(f * f * f)(x) &= \int\limits_{x-1}^{x+1} \frac{1}{2}\, (f * f)(z)\,\mathrm{d}z \\ &= \begin{cases} 0, & x \le -3, \\ \int_{-2}^{x+1} \frac{2+z}{8}\,\mathrm{d}z, & -3 < x \le -1, \\ \int_{x-1}^{0} \frac{2+z}{8}\,\mathrm{d}z + \int\limits_{0}^{x+1} \frac{2-z}{8}\,\mathrm{d}z, & -1 < x \le 1, \\ \int_{x-1}^{2} \frac{2-z}{8}\,\mathrm{d}z, & 1 < x \le 3, \\ 0, & x > 3\,; \end{cases}\end{aligned}$$

$$(f * f * f)(x) = \begin{cases} 0, & x \le -3, \\ \frac{x^2+6x+9}{16}, & -3 < x \le -1, \\ \frac{6-2x^2}{16}, & -1 < x \le 1, \\ \frac{x^2-6x+9}{16}, & 1 < x \le 3, \\ 0, & x > 3\,. \end{cases}$$

If we evaluate $f * f$ at the indicated positions, we obtain:

$$\begin{aligned}(f*f)(-2) &= 0\\ (f*f)(-1) &= \frac{1}{4}\\ (f*f)(0) &= \frac{1}{2}\\ (f*f)(1) &= \frac{1}{4}\\ (f*f)(2) &= 0\end{aligned}$$

which corresponds to the discrete convolution $(f * f)$ from the Problem 2. As for $(f * f * f)$ we get

$$\begin{aligned}(f*f*f)(-3) &= 0\\ (f*f*f)(-2) &= \frac{1}{16}\\ (f*f*f)(-1) &= \frac{1}{4}\\ (f*f*f)(0) &= \frac{6}{16}\\ (f*f*f)(1) &= \frac{1}{4}\\ (f*f*f)(2) &= \frac{1}{16}\\ (f*f*f)(3) &= 0\end{aligned}$$

which corresponds to the discrete convolution $(f * f * f * f)$ from Problem 2. We note here that the correspondences are not exact, in particular the indices of the spatial grid points are not aligned and corresponding results do not necessarily take the same number of iterations. Nevertheless, both the continuous and the discrete convolutions of box filters show the same tendencies. The common patterns $\frac{1}{4}, \frac{1}{2}, \frac{1}{4}$ and $\frac{1}{16}, \frac{4}{16}, \frac{6}{16}, \frac{4}{16}, \frac{1}{16}$ are actually so called binomial kernels which are used to approximate Gaussian filters and will be discussed in upcoming lectures. In particular, both the continuous and discrete convolutions of box filters approximate Gaussians better and better the higher the number of convolutions.

(This problem will be useful for approximating continuous Gaussian convolution.)

Problem H1.4 (Properties of the Convolution)

The continuous convolution between f and g is given by

$$(f * g)(x) = \int_{\mathbb{R}} f(t)g(x-t)\mathrm{d}t$$

(a) Commutativity holds as

$$\begin{aligned}(f * g)(x) &= \int_{-\infty}^{\infty} f(t)g(x-t)\mathrm{d}t \\ &= \int_{\infty}^{-\infty} f(x-z)g(x-(x-z))(-1)\mathrm{d}z \\ &= \int_{-\infty}^{\infty} g(z)f(x-z)\mathrm{d}z \\ &= (g * f)(x)\end{aligned}$$

(b) As for the associativity, it is easy to see that

$$\begin{aligned}((f * g) * h)(x) &= \int_{\mathbb{R}} (f * g)(z)h(x-z)\mathrm{d}z \\ &= \int_{\mathbb{R}} \int_{\mathbb{R}} f(t)g(z-t)h(x-z)\mathrm{d}t\mathrm{d}z \quad y \leftrightarrow x-z \\ &= \int_{\mathbb{R}} \int_{\mathbb{R}} f(t)g(x-t-y)h(y)\mathrm{d}t\mathrm{d}y \\ (f * (g * h))(x) &= \int_{\mathbb{R}} f(t)(g * h)(x-t)\mathrm{d}t \\ &= \int_{\mathbb{R}} f(t)(h * g)(x-t)\mathrm{d}t \\ &= \int_{\mathbb{R}} \int_{\mathbb{R}} f(t)h(z)g(x-t-z)\mathrm{d}t\mathrm{d}z\end{aligned}$$

The result follows now immediately after renaming the variable y back into z.

(c) The distributivity follows almost immediately from the definition of the convolution

$$\begin{aligned}((f+g)*h)(x) &= \int_{\mathbb{R}} (f+g)(t)h(x-t)\mathrm{d}t \\ &= \int_{\mathbb{R}} f(t)h(x-t)\mathrm{d}t + \int_{\mathbb{R}} g(t)h(x-t)\mathrm{d}t \\ &= (f*h)(x) + (g*h)(x)\end{aligned}$$

The second equations is now a direct consequence of the first one combined with the commutativity.

(d) We compute the derivative $(f*g)' = (g*f)'$ (due to commutativity).

$$\begin{aligned}(g*f)'(x) &= \frac{d}{dx}\int_{-\infty}^{\infty} g(x-t)\ f(t)\ dt \\ &= \int_{-\infty}^{\infty} \frac{d}{dx} g(x-t)\ f(t)\ dt \\ &= \int_{-\infty}^{\infty} g'(x-t)\ f(t)\ dt \\ &= (g'*f)(x)\end{aligned}$$

We have assumed that all assumptions necessary for exchanging the derivative and the integral are satisfied. By induction we can repeat this step since g is n times differentiable, and write

$$\frac{d^n}{dx^n}(g*f) = \left(\frac{d^n}{dx^n} g\right) * f$$

We have shown that the n-th derivative of $(g*f)$ exists. Therefore $(f*g) \in C^n(\mathbb{R})$.

(e) For the linearity, we already know from the distributivity that

$$(\alpha f + \beta g) * h = (\alpha f) * h + (\beta g) * h.$$

It remains to show that $(\alpha f) * h = \alpha(f*h)$:

$$((\alpha f)*h)(x) = \int_{\mathbb{R}} (\alpha f(t))h(x-t)\,dt = \alpha \int_{\mathbb{R}} f(t)h(x-t)\,dt = \alpha(f*h)(x).$$

(f) Last but not least, we consider the shift invariance:

$$((T_b f) * g)(x) = \int_{\mathbb{R}} T_b f(t) g(x-t)\, dt = \int_{\mathbb{R}} f(t-b) g(x-t)\, dt.$$

We substitute $s = t - b \Leftrightarrow t = s + b$, $dt = ds$ and obtain

$$\begin{aligned}\int_{\mathbb{R}-b} f(s) g(x-(s+b))\, ds &= \int_{\mathbb{R}} f(s) g((x-b)-s)\, ds \\ &= (f * g)(x-b) \\ &= (T_b(f * g))(x).\end{aligned}$$

Problem H1.5 (Noise and Quantisation)

First of all we consider the code that has to be supplemented for each subproblem:

(a) Expected supplemented code for the required quantisation:

```
/* quantise the input image */

d = pow (2.0, 8-q);

for (j=1; j<=ny; j++)
 for (i=1; i<=nx; i++)
     u[i][j] = (floor(u[i][j] / d) + 0.5) * d;
```

(b) Expected supplemented code for the required quantisation with noise:

```
/* quantise the input image */

d = pow (2.0, 8-q);
noise = 0.0;

for (j=1; j<=ny; j++)
 for (i=1; i<=nx; i++)
     {
     noise = -0.5 + (double)rand() / RAND_MAX;
     u[i][j] = (floor (u[i][j] / d + noise) + 0.5) * d;
     }
```

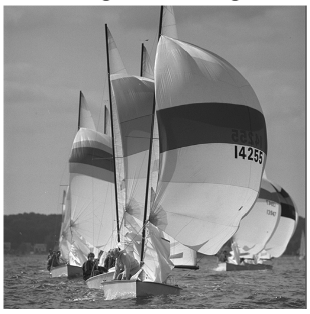
As we can see in Tab. 1, there is almost no visible difference between the original image and the one with $q = 6$, (e.g. 64 colours). This confirms the statement from the lecture that the human eye cannot distinguish many different grey values.

For $q = 3$ we clearly see a loss in quality compared to the original image. Especially in regions with a smooth gradient (such as the sky on the left side) the reduction in the number of colours really deteriorates the image. The quantisation without noise creates clear discontinuities and sharp edges. These discontinuities are further enhanced by the so called lateral inhibition in our eye so that the visual quality of the image suffers a lot.

Adding some noise, as done in the second column reduces this effect. In areas where a certain pixel value is on the edge between two different quantisation levels, the noise forces some of the pixel on the one side and some on the other such that an overall smoother transition between the two areas is achieved. This effect is visibile especially well on the sails of the boats in the centre of the picture.

Using $q = 1$ reveals how the current implementation treats over- and undershoots. The noisy method yield an images with more than two colours. The reason for this lies in the fact that the noise can lead to pixels with values larger than 255 and smaller than 0. These values are mapped to 0 (resp. 255) when the file is written to disk. Thus we actually obtain images with 4 colours, namely 0,64,192 and 255. In order to avoid this effect one could also truncate over- and undershoots, i.e. one would map the value 0 to 64 and 255 to 192. However, this would change final visual result only slightly.

Original Image



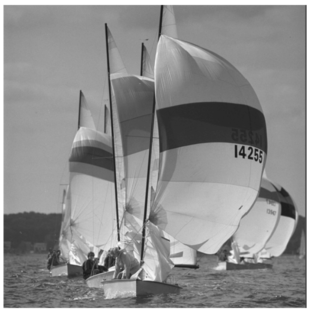
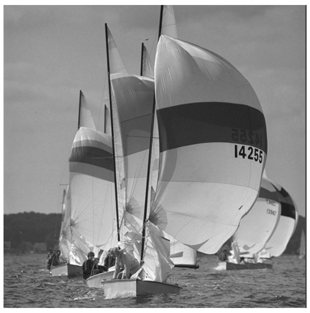
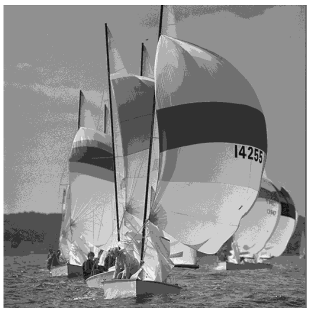
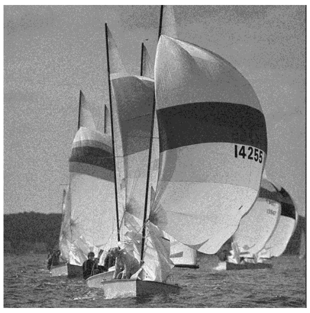
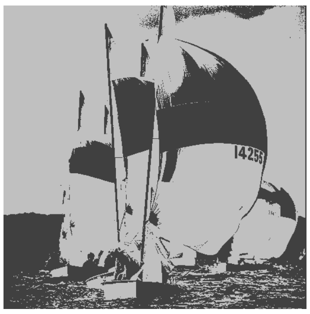
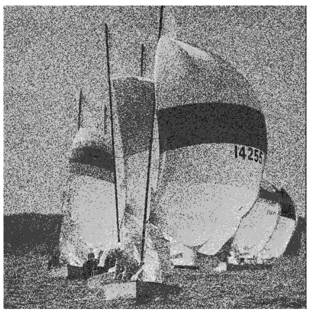
	Without noise	With noise
$q = 6$		
$q = 3$		
$q = 1$		

Table 1: Results for the different quantisation algorithms and varying numbers of colors used.