Image Processing and Computer Vision (IPCV)

M I
A

Prof. Dr. J. Weickert   Summer term 2017
Mathematical Image Analysis Group   Saarland University

Example Solutions for Homework Assignment 1 (H1)

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

We first review the definition of the PSNR:

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

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

(a) We have

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

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 $\text{PSNR}(u, g) = \text{PSNR}(f, g) + 10$. We calculate that

$$\text{PSNR}(u, g) = \text{PSNR}(f, g) + 10$$
$$\log_{10}\left(\frac{255^2}{\text{MSE}(u, g)}\right) = \log_{10}\left(\frac{255^2}{\text{MSE}(f, g)}\right) + 1$$
$$\log_{10}(\text{MSE}(u, g)) = \log_{10}(\text{MSE}(f, g)) - 1$$
$$\text{MSE}(u, g) = \frac{\text{MSE}(f, g)}{10}$$

This means that filtering the signal reduced the MSE by a factor 10.

(c) When computing $\text{PSNR}(g, g)$ we have $\text{MSE}(g, g) = 0$. The fraction $\frac{255^2}{\text{MSE}(g,g)}$ is therefore not defined.
Let us have a look at the limit: When $\text{MSE}(f, g)$ becomes infinitely small, the above mentioned fraction becomes infinitely large. Therefore,

$$\lim_{\text{MSE} \to 0}\left(10 \log_{10}\left(\frac{255^2}{\text{MSE}}\right)\right) \to \infty.$$

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

Problem 2 (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

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

$$
= \begin{cases}
0, & x \leq -2, \\
\int_{-1}^{x+1} \frac{1}{4}\, \mathrm{d}z, & -2 < x \leq 0, \\
\int_{x-1}^{1} \frac{1}{4}\, \mathrm{d}z, & 0 < x \leq 2, \\
0, & x > 2
\end{cases}
$$

and thus

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

For $f * f * f$ we find

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

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

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

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

$$(f * f)(-1) = \frac{1}{4}$$
$$(f * f)(0) = \frac{1}{2}$$
$$(f * f)(1) = \frac{1}{4}$$
$$(f * f)(2) = 0$$

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

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

which corresponds to the discrete convolution $(f*f*f*f)$ from the classroom assignment. We note here that the correspondences are not exact, in particular the indices of the

3

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.

Problem 3 (Properties of the Convolution)

The continuous convolution between f and g is given by

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

It follows,

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

and thus the commutativity holds. 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. The

distributivity follows almost immediately from the definition of the convolution

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

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

$$= (f * h)(x) + (g * h)(x)$$

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

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. Since g is n times differentiable, we can repeat this step 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 $(f * g) \in C^n(\mathbb{R})$.

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 $(\alpha f) * h = \alpha(f * h)$:

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

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

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

Problem 4 (Noise, Quantisation and Dithering)

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 */
float d = powf(2.0f,8-q);
for (j=1; j<=ny; j++)
 for (i=1; i<=nx; i++)
     {
       u[i][j] = (floorf(u[i][j] / d) + 0.5) * d;
     }
```

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

```
/* quantise the input image */
float d = powf(2.0f,8-q);
double n = 0.0f;
for (j=1; j<=ny; j++)
 for (i=1; i<=nx; i++)
     {
       n = -0.5 + (double)rand() / RAND_MAX;
       u[i][j] = (floorf(u[i][j] / d + n) + 0.5) * d;
     }
```

(c) Expected supplemented code for the required Floyd-Steinberg algorithm:

```
/* quantise the input image */
float d = powf(2.0f,8-q);
float old = 0;
float error = 0;
for (j=1; j<=ny; j++)
 for (i=1; i<=nx; i++)
     {
       old = u[i][j];
       u[i][j] = (floorf(u[i][j] / d) + 0.5) * d;
       error = old - u[i][j];
       u[i+1][j  ] += 7.0f / 16.0f * error;
       u[i-1][j+1] += 3.0f / 16.0f * error;
       u[i  ][j+1] += 5.0f / 16.0f * error;
       u[i+1][j+1] += 1.0f / 16.0f * error;
     }
```

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 clouds in the upper right corner) 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.

However, the best results are obtained through the Floyd-Steinberg Algorithm. Similarly as in (b), we get smooth transitions (e.g. on the clouds and the sails of the boats). Our visual system does not consider each pixel individually, but instead it does some smoothing, such that the perception in a given point is also dependent on its neighbourhood. The Floyd-Steinberg algorithm exploits this fact to redistribute the quantised values to the neighbourhood such that our brain perceives an image that looks similar to the original one.

Using $q = 1$ reveals how the current implementation treats over- and undershoots. The noisy method as well as the Floyd-Steinberg algorithm yield 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	Floyd Steinberg

$q = 6$

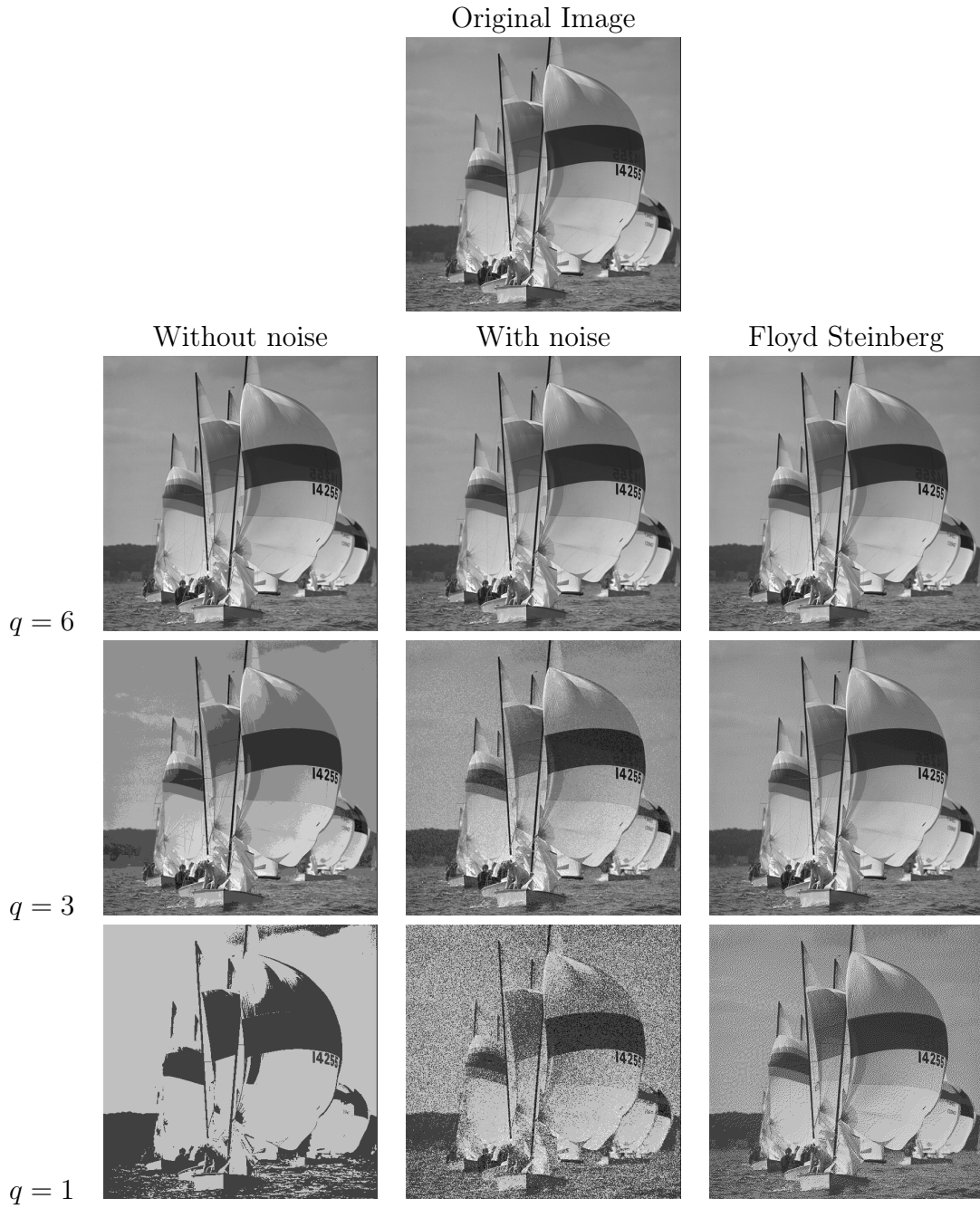$q = 3$

$q = 1$

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