Image Processing and Computer Vision (IPCV)

Prof. J. Weickert
Mathematical Image Analysis Group
Summer term 2019
Saarland University

Example Solutions for Homework Assignment 6 (H6)

Problem 1 (Fourier Analysis of Derivative Filters)

(a) We compute the Fourier transform of the function

$$g(x) = \frac{-f(x+2h) + 8f(x+h) - 8f(x-h) + f(x-2h)}{12h}$$

using linearity and shift theorem:

$$\begin{aligned}
&\mathcal{F}[g](u) \\
=&\frac{1}{12h}\big(- \mathcal{F}[f(x+2h)](u) + 8\mathcal{F}[f(x+h)](u) - 8\mathcal{F}[f(x-h)] + \mathcal{F}[f(x-2h)]\big) \\
=&\frac{1}{12h}\left(-e^{4\pi ihu} + 8e^{2\pi ihu} - 8e^{-2\pi ihu} + e^{-4\pi ihu})\mathcal{F}[f(x)](u)\right) \\
=&\frac{1}{12h}\left(-2i\sin(4\pi hu) + 16i\sin(2\pi hu)\right)\mathcal{F}[f](u)
\end{aligned}$$

(b) The computation of a Taylor expansion around 0 gives us

$$\begin{aligned}
\sin(4\pi hu) =& \frac{\sin(0)}{0!} + \frac{\cos(0)}{1!}4\pi hu - \frac{\sin(0)}{2!}(4\pi hu)^2 - \frac{\cos(0)}{3!}(4\pi hu)^3 \\
&+ \frac{\sin(0)}{4!}(4\pi hu)^4 + \frac{\cos(0)}{5!}(4\pi hu)^5 + \mathcal{O}(h^6u^6) \\
=& 4\pi hu - \frac{32}{3}\pi^3h^3u^3 + \frac{128}{15}\pi^5h^5u^5 + \mathcal{O}(h^6u^6) \\
\sin(2\pi hu) =& \frac{\sin(0)}{0!} + \frac{\cos(0)}{1!}2\pi hu - \frac{\sin(0)}{2!}(2\pi hu)^2 - \frac{\cos(0)}{3!}(2\pi hu)^3 \\
&+ \frac{\sin(0)}{4!}(2\pi hu)^4 + \frac{\cos(0)}{5!}(2\pi hu)^5 + \mathcal{O}(h^6u^6) \\
=& 2\pi hu - \frac{4}{3}\pi^3h^3u^3 + \frac{4}{15}\pi^5h^5u^5 + \mathcal{O}(h^6u^6)
\end{aligned}$$

Let us now use this Taylor expansion to compute the order of approximation explicitly. The derivation rule for the Fourier transform yields:

$$\mathcal{F}[f'](u) = i2\pi u\mathcal{F}[f](u)$$

and thus, using additionally the Taylor-expansion from (b), we get:

$$\begin{aligned}
&\mathcal{F}[g](u) - \mathcal{F}[f'](u) \\
&= \frac{1}{12h}\left(-2i\sin(4\pi hu) + 16i\sin(2\pi hu)\right)\mathcal{F}[f](u) - i2\pi u\mathcal{F}[f](u) \\
&= \Big(\frac{1}{12h}\Big(- 8i\pi hu + \frac{64}{3}i\pi^3h^3u^3 - \frac{256}{15}i\pi^5h^5u^5 + \mathcal{O}(h^6u^6) \\
&\qquad + 32i\pi hu - \frac{64}{3}i\pi^3h^3u^3 + \frac{64}{15}i\pi^5h^5u^5 + \mathcal{O}(h^6u^6)\Big) \\
&\qquad - 2i\pi u\Big)\mathcal{F}[f](u) \\
&= -\frac{16}{15}i\pi^5h^4u^5\mathcal{F}[f](u) + \mathcal{O}(h^5u^6)
\end{aligned}$$

Thus, we get a consistency order of 4. Note that it is equal to the approximation order obtains in a spatial approximation analysis. However, the dependency of the error term on the frequency u yields additional information.

(c) The approximation error will increase for higher frequencies, since the error terms in $\mathcal{O}(h^4u^5)$ depend clearly on u and will thus become larger for higher frequencies.

Problem 2 (2-D Derivative Filters)

(a) It is clear that we have a linear combination of two approximations. What remains to be shown is that they are approximations of the Laplacian.

Using the definition of the stencil notation, we can rewrite the stencils as

$$\Delta u_{ij} \approx \frac{1}{2h^2}(u_{i+1,j+1} + u_{i+1,j-1} + u_{i-1,j+1} + u_{i-1,j-1} - 4u_{ij}) \tag{1}$$

$$\Delta u_{ij} \approx \frac{1}{h^2}(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{ij}), \tag{2}$$

giving us two formulas for the Laplacian, that depend on 9 pixels. For those points, we compute the 2-D Taylor expansion, given by the formula

$$u(\boldsymbol{x} + \boldsymbol{h}) = \sum_{k=0}^{n} \frac{1}{k!}\langle \boldsymbol{h}, \boldsymbol{\nabla}\rangle^k u(\boldsymbol{x}) + \mathcal{O}(|\boldsymbol{h}|^{n+1}),$$

with $\boldsymbol{h} = (h_1, h_2)^\top$ and $\boldsymbol{\nabla} = (\partial_x, \partial_y)^\top$.
Let us compute the Taylor expansions for each pixel, starting with $u_{i+1,j+1}$, for which $\boldsymbol{h} = (h_1, h_2)^\top := (+1 \cdot h, +1 \cdot h)^\top = (h, h)^\top$:

$$\begin{aligned}
u_{i+1,j+1} &= u_{ij} + \frac{1}{1!}\langle \boldsymbol{h}, \boldsymbol{\nabla}\rangle^1 u_{ij} + \frac{1}{2!}\langle \boldsymbol{h}, \boldsymbol{\nabla}\rangle^2 u_{ij} \\
&+ \frac{1}{3!}\langle \boldsymbol{h}, \boldsymbol{\nabla}\rangle^3 u_{ij} + \frac{1}{4!}\langle \boldsymbol{h}, \boldsymbol{\nabla}\rangle^4 u_{ij} + \mathcal{O}(h^5) \\
&= u_{ij} + h\left(\partial_x + \partial_y\right)^1 u_{ij} + \frac{h^2}{2}\left(\partial_x + \partial_y\right)^2 u_{ij} \\
&+ \frac{h^3}{6}\left(\partial_x + \partial_y\right)^3 u_{ij} + \frac{h^4}{24}\left(\partial_x + \partial_y\right)^4 u_{ij} + \mathcal{O}(h^5) \\
&= u_{ij} + h\partial_x u_{ij} + h\partial_y u_{ij} \\
&+ \frac{h^2}{2}\left(\partial_{xx} + 2\partial_{xy} + \partial_{yy}\right) u_{ij} \\
&+ \frac{h^3}{6}\left(\partial_{xxx} + 3\partial_{xxy} + 3\partial_{xyy} + \partial_{yyy}\right) u_{ij} \\
&+ \frac{h^4}{24}\left(\partial_{xxxx} + 4\partial_{xxxy} + 6\partial_{xxyy} + 4\partial_{xyyy} + \partial_{yyyy}\right) u_{ij}
\end{aligned}$$

For the Taylor expansion for $u_{i+1,j-1}$ with $\boldsymbol{h} := (+1 \cdot h, -1 \cdot h)^\top =$

$(h, -h)^\top$ we get

$$\begin{aligned} u_{i+1,j-1} = u_{ij} &+ h\partial_x u_{ij} - h\partial_y u_{ij} \\ &+ \frac{h^2}{2}\left(\partial_{xx} - 2\partial_{xy} + \partial_{yy}\right) u_{ij} \\ &+ \frac{h^3}{6}\left(\partial_{xxx} - 3\partial_{xxy} + 3\partial_{xyy} - \partial_{yyy}\right) u_{ij} \\ &+ \frac{h^4}{24}\left(\partial_{xxxx} - 4\partial_{xxxy} + 6\partial_{xxyy} - 4\partial_{xyyy} + \partial_{yyyy}\right) u_{ij} + \mathcal{O}(h^5) \end{aligned}$$

and analogously for $u_{i-1,j+1}$ and $u_{i-1,j-1}$ with $\boldsymbol{h} := (-1 \cdot h, +1 \cdot h)^\top = (-h, h)^\top$ and $\boldsymbol{h} := (-1 \cdot h, -1 \cdot h)^\top = (-h, -h)^\top$:

$$\begin{aligned} u_{i-1,j+1} = u_{ij} &- h\partial_x u_{ij} + h\partial_y u_{ij} \\ &+ \frac{h^2}{2}\left(\partial_{xx} - 2\partial_{xy} + \partial_{yy}\right) u_{ij} \\ &+ \frac{h^3}{6}\left(-\partial_{xxx} + 3\partial_{xxy} - 3\partial_{xyy} + \partial_{yyy}\right) u_{ij} \\ &+ \frac{h^4}{24}\left(\partial_{xxxx} - 4\partial_{xxxy} + 6\partial_{xxyy} - 4\partial_{xyyy} + \partial_{yyyy}\right) u_{ij} + \mathcal{O}(h^5) \end{aligned}$$

$$\begin{aligned} u_{i-1,j-1} = u_{ij} &- h\partial_x u_{ij} - h\partial_y u_{ij} \\ &+ \frac{h^2}{2}\left(\partial_{xx} + 2\partial_{xy} + \partial_{yy}\right) u_{ij} \\ &- \frac{h^3}{6}\left(\partial_{xxx} + 3\partial_{xxy} + 3\partial_{xyy} + \partial_{yyy}\right) u_{ij} \\ &+ \frac{h^4}{24}\left(\partial_{xxxx} + 4\partial_{xxxy} + 6\partial_{xxyy} + 4\partial_{xyyy} + \partial_{yyyy}\right) u_{ij} + \mathcal{O}(h^5) \end{aligned}$$

The derivatives for $u_{i\pm1,j}$ and $u_{i,j\pm1}$ are a little bit simpler and reduce to the 1-D cases in x and y direction, respectively: $\boldsymbol{h} := (+1 \cdot h, 0 \cdot h)^\top = (h, 0)^\top$ gives

$$u_{i+1,j} = u_{ij} + h\partial_x u_{ij} + \frac{h^2}{2}\partial_{xx} u_{ij} + \frac{h^3}{6}\partial_{xxx} u_{ij} + \frac{h^4}{24}\partial_{xxxx} u_{ij} + \mathcal{O}(h^5)$$

$\boldsymbol{h} := (-1 \cdot h, 0 \cdot h)^\top = (-h, 0)^\top$ gives

$$u_{i-1,j} = u_{ij} - h\partial_x u_{ij} + \frac{h^2}{2}\partial_{xx} u_{ij} - \frac{h^3}{6}\partial_{xxx} u_{ij} + \frac{h^4}{24}\partial_{xxxx} u_{ij} + \mathcal{O}(h^5)$$

$\boldsymbol{h} := (0 \cdot h, 1 \cdot h)^\top = (0, h)^\top$ gives

$$u_{i,j+1} = u_{ij} + h\partial_y u_{ij} + \frac{h^2}{2}\partial_{yy} u_{ij} + \frac{h^3}{6}\partial_{yyy} u_{ij} + \frac{h^4}{24}\partial_{yyyy} u_{ij} + \mathcal{O}(h^5)$$

$\boldsymbol{h} := (0 \cdot h, -1 \cdot h)^\top = (0, -h)^\top$ gives

$$u_{i,j-1} = u_{ij} - h\partial_y u_{ij} + \frac{h^2}{2}\partial_{yy} u_{ij} - \frac{h^3}{6}\partial_{yyy} u_{ij} + \frac{h^4}{24}\partial_{yyyy} u_{ij} + \mathcal{O}(h^5)$$

When plugging these expansions into (1), several terms cancel out and we arrive at

$$\begin{aligned}
\Delta u_{ij} \approx \frac{1}{2h^2} \Bigg(& 4 \cdot \frac{h^2}{2}\partial_{xx} u_{ij} + 4 \cdot \frac{h^2}{2}\partial_{yy} u_{ij} \\
& + 4 \cdot \frac{h^4}{24}\partial_{xxxx} u_{ij} + 4 \cdot \frac{6h^4}{24}\partial_{xxyy} u_{ij} + 4 \cdot \frac{h^4}{24}\partial_{yyyy} u_{ij} \Bigg) + \mathcal{O}(h^3) \\
& = \partial_{xx} u_{ij} + \partial_{yy} u_{ij} \\
& + \frac{h^2}{12}\left(\partial_{xxxx} u_{ij} + 6\partial_{xxyy} u_{ij} + \partial_{yyyy} u_{ij}\right) + \mathcal{O}(h^3) \\
& \approx \Delta u_{ij} + \frac{h^2}{12}\left(\partial_{xxxx} u_{ij} + 6\partial_{xxyy} u_{ij} + \partial_{yyyy} u_{ij}\right) + \mathcal{O}(h^3)\,. \qquad (3)
\end{aligned}$$

When we plug these expansions into (2) we get

$$\begin{aligned}
\Delta u_{ij} \approx \frac{1}{h^2} \Bigg(& 2 \cdot \frac{h^2}{2}\partial_{xx} u_{ij} + 2 \cdot \frac{h^2}{2}\partial_{yy} u_{ij} \\
& + 2 \cdot \frac{h^4}{24}\partial_{xxxx} u_{ij} + 2 \cdot \frac{h^4}{24}\partial_{yyyy} u_{ij} \Bigg) + \mathcal{O}(h^3) \\
& = \partial_{xx} u_{ij} + \partial_{yy} u_{ij} \\
& + \frac{h^2}{12}\left(\partial_{xxxx} u_{ij} + \partial_{yyyy} u_{ij}\right) + \mathcal{O}(h^3) \\
& \approx \Delta u_{ij} + \frac{h^2}{12}\left(\partial_{xxxx} u_{ij} + \partial_{yyyy} u_{ij}\right) + \mathcal{O}(h^3)\,. \qquad (4)
\end{aligned}$$

We now see that both stencils give a consistent approximation of the Laplacian.

(b) We can use the Taylor expansion from (3) and (4) to rewrite our stencil as

$$\begin{aligned}\Delta u_{i,j} \approx & \alpha \left(\Delta u_{ij} + \frac{h^2}{12} \left(\partial_{xxxx} u_{ij} + 6 \partial_{xxyy} u_{ij} + \partial_{yyyy} u_{ij} \right) + \mathcal{O}(h^3) \right) \\ & + (1-\alpha) \left(\Delta u_{ij} + \frac{h^2}{12} \left(\partial_{xxxx} u_{ij} + \partial_{yyyy} u_{ij} \right) + \mathcal{O}(h^3) \right) \\ = & \Delta u_{i,j} + \frac{h^2}{12} \left(\partial_{xxxx} u_{ij} + 6\alpha \partial_{xxyy} u_{ij} + \partial_{yyyy} u_{ij} \right) + \mathcal{O}(h^3). \end{aligned} \tag{5}$$

(c) Using the approximation for Δu_{ij} given by (5), we can achieve the desired result and its error term (also given in Lecture 12, Slide 24) by setting $\alpha = 1/3$:

$$\begin{aligned}\Delta u_{ij} &\approx \frac{1}{6}\frac{1}{h^2} \begin{array}{|c|c|c|} \hline 1 & 0 & 1 \\ \hline 0 & -4 & 0 \\ \hline 1 & 0 & 1 \\ \hline \end{array} u_{i,j} + \frac{4}{6}\frac{1}{h^2} \begin{array}{|c|c|c|} \hline 0 & 1 & 0 \\ \hline 1 & -4 & 1 \\ \hline 0 & 1 & 0 \\ \hline \end{array} u_{i,j} \\ &= \frac{1}{6h^2} \begin{array}{|c|c|c|} \hline 1 & 4 & 1 \\ \hline 4 & -20 & 4 \\ \hline 1 & 4 & 1 \\ \hline \end{array} u_{i,j} \end{aligned}$$

leading to the second order error term

$$\frac{h^2}{12} \left(\partial_{xxxx} u_{ij} + 2 \partial_{xxyy} u_{ij} + \partial_{yyyy} u_{ij} \right) + \mathcal{O}(h^4) = \frac{h^2}{12} \Delta(\Delta u)_{ij} + \mathcal{O}(h^4) \, .$$

Although the standard discretisation has the same approximation order as the one with the rotationally invariant leading error term, the latter will often yield the better result as it does not have a preferred direction, which is often the cause for artifacts in images.

Problem 3 (Linear Filters)

(a) In the lecture, the highpass and bandpass filter are defined as substractions of lowpass filters and the identity. If two LSI filters A and B are given, one can, due to the superposition principle, also express the result of the filter $A - B$ as the difference of filtered images:

$$(A - B)u = Au - Bu.$$

This is what we exploit for a simple implementation of highpass and bandpass filters. The code for the routine `highpass` is given by

```
/*--------------------------------------------------------------------*/

void highpass
     (float   sigma,      /* standard deviation of Gaussian */
      long    nx,         /* image dimension in x direction */
      long    ny,         /* image dimension in y direction */
      float   **f)        /* input: original; output: highpass filtered */

/* highpass filter */

{
long   i,j;       /* loop variables */
float  **gauss;   /* Gaussian smoothed image */

/* allocate memory */
alloc_matrix (&gauss, nx+2, ny+2);

/* copy original image to temporary array */
for (i=1; i<=nx; i++)
 for (j=1; j<=ny; j++)
     gauss[i][j] = f[i][j];

/* apply Gaussian to temporary array */
gauss_conv(sigma, nx, ny, GRID_SIZE, GRID_SIZE, PRECISION, BOUNDARY, gauss);

/* compute highpass filter */
for (i=1; i<=nx; i++)
 for (j=1; j<=ny; j++)
     {
       f[i][j] -= gauss[i][j];
     }

/* free memory */
disalloc_matrix (gauss, nx+2, ny+2);

return;
}

/*--------------------------------------------------------------------*/
```

For the routine `bandpass` the following code is obtained:

```
/*-------------------------------------------------------------------*/

void bandpass
     (float   sigma1,     /* standard deviation of first Gaussian */
      float   sigma2,     /* standard deviation of second Gaussian */
      long    nx,         /* image dimension in x direction */
      long    ny,         /* image dimension in y direction */
      float   **f)        /* input: original; output: bandpass filtered */

/* bandpass filter: sigma1 > sigma2 */

{
long   i,j;        /* loop variables */
float  **gauss1;   /* Gaussian smoothed image (stddev sigma1) */
float  **gauss2;   /* Gaussian smoothed image (stddev sigma2) */

/* allocate memory */
alloc_matrix (&gauss1, nx+2, ny+2);
alloc_matrix (&gauss2, nx+2, ny+2);

/* copy original image to temporary arrays */
for (i=1; i<=nx; i++)
 for (j=1; j<=ny; j++)
     {
     gauss1[i][j] = f[i][j];
     gauss2[i][j] = f[i][j];
     }

/* apply Gaussian to temporary arrays */
gauss_conv (sigma1, nx, ny, GRID_SIZE, GRID_SIZE, PRECISION, BOUNDARY, gauss1);
gauss_conv (sigma2, nx, ny, GRID_SIZE, GRID_SIZE, PRECISION, BOUNDARY, gauss2);

/* compute bandpass filter */
for (i=1; i<=nx; i++)
 for (j=1; j<=ny; j++)
     {
       f[i][j] = gauss2[i][j] - gauss1[i][j];
     }

/* free memory */
disalloc_matrix (gauss1, nx+2, ny+2);
disalloc_matrix (gauss2, nx+2, ny+2);

return;
}

/*-------------------------------------------------------------------*/
```

Note that the order of Gaussians here makes a difference and that the basic idea of substracting the narrow kernel from the broad kernel references the sizes in the Fourier domain.

(b)

- In order to remove the high-frequent noise in `leopard.pgm`, we just have to apply a lowpass filter, i.e. Gaussian convolution, to taste. A possible parameter choice is $\sigma = 3$ (see Figure 1).

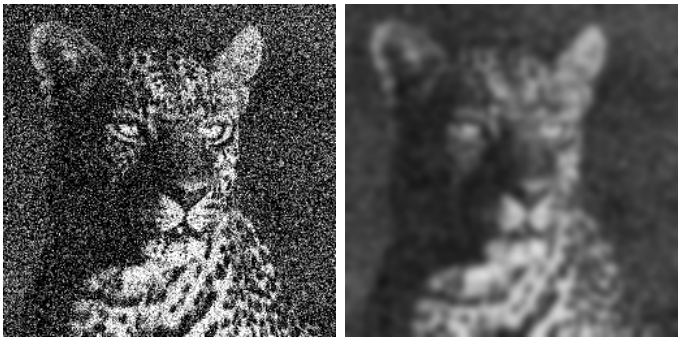

Figure 1: **Left:** `leopard.pgm`: **Right:** Lowpass filter result with $\sigma = 3$.

- Background structures are lowfrequent and thus a highpass filter should be applied to `angiogram.pgm`. We choose here $\sigma = 2$ and obtain the result in Figure 2.

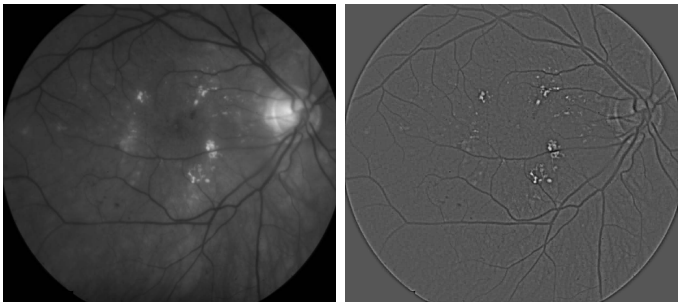

Figure 2: **Left:** `angiogram.pgm`: **Right:** Highpass filter result with $\sigma = 2$.

- Finally, we can isolate the dark gaps between the tiles in `tile.pgm` by designing a suitable bandpass filter (e.g. $\sigma_1 = 8, \sigma_2 = 2$). Since the gaps are mapped close to zero, it's best to compare the inverse of the original and the filtered image. In particular, due to the linearity of the filter, the mapping of gaps to bright structures and tile pixels to values close to zero can be obtained by swapping the roles of σ_1 and σ_2.

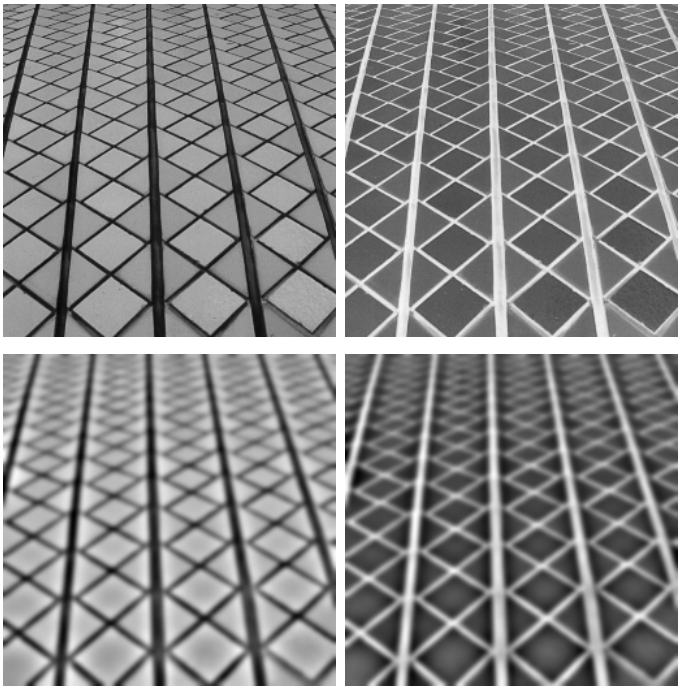

Figure 3: **Top:** `tile.pgm` and its inverse. **Bottom:** Bandpass filter result with $\sigma_1 = 8$, $\sigma_2 = 2$ and its inverse.
