Image Processing and Computer Vision (IPCV)

Prof. J. Weickert
Mathematical Image Analysis Group
Winter term 13/14
Saarland University

Example Solutions for Homework Assignment 5 (H5)

Problem 1: (Keys Interpolation)

In order to compute the interpolant for $x \in (0,1)$, the easiest way is to take in account that the Keys synthesis function $\varphi_{\text{int}}(x)$ has a limited support, i.e.it vanishes outside of the interval $[-2,2]$. If the function is shifted, the support is also shifted. Thus, the support of $\varphi_{\text{int}}(x-k)$ is $[-2+k, 2+k]$.

For us, only the shifted synthesis functions are relevant which have a support that overlaps with $(0,1)$. Thus, only for $k \in \{-1,0,1,2\}$ the synthesis function $\varphi_{\text{int}}(x-k)$ does not vanish for $x \in (0,1)$ and we get a finite sum for the interpolant:

$$u(x) = \sum_{k=-1}^{2} f_k \varphi_{\text{int}}(x-k) = 3\varphi_{\text{int}}(x+1) + \varphi_{\text{int}}(x-1) + 6\varphi_{\text{int}}(x-2).$$

Due to $f_0 = 0$, $\varphi_{\text{int}}(x)$ does not have to be computed. Fortunately, as we only evaluate the interpolant for a very specific range of x, we only have to do one case distinction for the remaining three synthesis functions, i.e. we have to decide if $|x-k| \in (-1,1)$ or if $|x-k| \in [1,2)$ holds. Following this approach and removing the absolute value by suitable sign changes under the assumption $x \in (0,1)$, we get:

$$\begin{aligned}
\varphi_{\text{int}}(x+1) &= -\frac{1}{2}|x+1|^3 + \frac{5}{2}|x+1|^2 - 4|x+1| + 2 \\
&= -\frac{1}{2}x^3 + x^2 - \frac{x}{2} \\
\varphi_{\text{int}}(x-1) &= \frac{3}{2}|x-1|^3 - \frac{5}{2}(x-1)^2 + 1 \\
&= -\frac{3}{2}x^3 + 2x^2 + \frac{x}{2} \\
\varphi_{\text{int}}(x-2) &= -\frac{1}{2}|x-2|^3 + \frac{5}{2}|x-2|^2 - 4|x-2| + 2 \\
&= \frac{1}{2}x^3 - \frac{x^2}{2}
\end{aligned}$$

Plugging these intermediate results into the formula of the interpolant yields:

$$\begin{aligned} u(x) &= \sum_{k=-1}^{2} f_k \varphi_{\text{int}}(x-k) = \varphi_{\text{int}}(x+1) + \varphi_{\text{int}}(x-1) + 4\varphi_{\text{int}}(x-2) \\ &= (-\frac{3}{2} - \frac{3}{2} + \frac{6}{2})x^3 + (3+2-3)x^2 + (-\frac{3}{2} + \frac{1}{2})x \\ &= 2x^2 - x \end{aligned}$$

Thus, the Keys interpolant reproduces $2x^2 - x$ perfectly for $x \in (0,1)$. In the same fashion, one could also prove this for arbitrary choices of x, which however requires a lot of tedious case distinctions. One should also note, that the Keys interpolant only globally interpolates quadratic functions perfectly, if an infinite number of samples is used. Using a finite number of samples means implicitly to set all other samples to zero.

Using e.g. only the three samples $f_{-1} = 1$, $f_0 = 0$ and $f_1 = 1$ in order to interpolate $2x^2 - x$ will not yield a perfect reconstruction. Due to $f_k = 0$ for $k < -1$ and $k > 1$ we actually do not provide samples of $2x^2 - x$ to the Keys interpolant, but of the piecewise defined function

$$g(x) = \begin{cases} 2x^2 - x, & x \in [-1,1] \\ 0, & \text{else} \end{cases}$$

which is clearly not quadratic on the whole domain.

Problem 2 (Quadratic B-Spline Interpolation)

(a) The synthesis function for quadratic B-splines is defined as

$$\beta_2(x) = \begin{cases} \frac{3}{4} - x^2, & |x| < \frac{1}{2} \\ \frac{1}{2}\left(\frac{3}{2} - |x|\right)^2, & \frac{1}{2} \leqslant |x| < \frac{3}{2} \\ 0, & \text{else} \end{cases}$$

Evaluating this function at the positions $0, 1$ and 2 yields

$$\beta_2(0) = \frac{3}{4}, \quad \beta_2(1) = \frac{1}{8}, \quad \beta_2(2) = 0$$

Thus, the following linear system has to be solved in order to obtain the interpolation coefficients:

$$\begin{pmatrix} \frac{3}{4} & \frac{1}{8} & 0 \\ \frac{1}{8} & \frac{3}{4} & \frac{1}{8} \\ 0 & \frac{1}{8} & \frac{3}{4} \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} 17 \\ 0 \\ 17 \end{pmatrix}$$

Computing the matrix vector product

$$\begin{pmatrix} \frac{3}{4} & \frac{1}{8} & 0 \\ \frac{1}{8} & \frac{3}{4} & \frac{1}{8} \\ 0 & \frac{1}{8} & \frac{3}{4} \end{pmatrix} \begin{pmatrix} 24 \\ -8 \\ 24 \end{pmatrix}$$

verifies that $(24, -8, 24)^\top$ indeed solves the linear system of equations.

(b) Note that the synthesis function has to be shifted by 1 to the left, since the definition of x_1, x_2 and x_3 differs from the lecture. The interpolant is given by

$$\begin{aligned} I(x) &= \sum_{k=1}^{3} c_k \beta_2(x - k - 1) \\ &= 24\beta_2(x) - 8\beta_2(x-1) + 24\beta_2(x-2) \end{aligned}$$

Let us now have a look at the domains where each summand is defined and which value it yields there. First of all, it is easy to see that for any positive a we have

$$|x - a| \in \left[0, \frac{1}{2}\right] \Leftrightarrow x \in \left[-\frac{1}{2} + a, \frac{1}{2} + a\right]$$

$$|x - a| \in \left[\frac{1}{2}, \frac{3}{2}\right] \Leftrightarrow x \in \left[-\frac{3}{2} + a, -\frac{1}{2} + a\right] \cup \left[\frac{1}{2} + a, \frac{3}{2} + a\right]$$

It follows, that $\beta_2(x-1)$ is nonzero on $\left[-\frac{1}{2}, \frac{5}{2}\right]$ and $\beta_2(x-2)$ is nonzero on $\left[\frac{1}{2}, \frac{7}{2}\right]$. Now it is enough to insert the correct expression in the corresponding interval, as it is done in the following table.

	$\left[-\frac{3}{2}, -\frac{1}{2}\right]$	$\left[-\frac{1}{2}, \frac{1}{2}\right]$	$\left[\frac{1}{2}, \frac{3}{2}\right]$	$\left[\frac{3}{2}, \frac{5}{2}\right]$	$\left[\frac{5}{2}, \frac{7}{2}\right]$
$\beta_2(x)$	$\frac{1}{2}\left(\frac{3}{2}-\lvert x\rvert\right)^2$	$\frac{3}{4}-x^2$	$\frac{1}{2}\left(\frac{3}{2}-\lvert x\rvert\right)^2$	0	0
$24\beta_2(x)$	$12\left(\frac{3}{2}-\lvert x\rvert\right)^2$	$18-24x^2$	$12\left(\frac{3}{2}-\lvert x\rvert\right)^2$	0	0
$-8\beta_2(x-1)$	0	$-4\left(\frac{3}{2}-\lvert x-1\rvert\right)^2$	$-6+8(x-1)^2$	$-4\left(\frac{3}{2}-\lvert x-1\rvert\right)^2$	0
$24\beta_2(x-2)$	0	0	$12\left(\frac{3}{2}-\lvert x-2\rvert\right)^2$	$18-24(x-2)^2$	$12\left(\frac{3}{2}-\lvert x-2\rvert\right)^2$

Thus, we obtain

$$I(x)=\begin{cases} 12\left(\frac{3}{2}-|x|\right)^2, & x\in\left[-\frac{3}{2},-\frac{1}{2}\right] \\ 18-24x^2-4(\frac{3}{2}-|x-1|)^2, & x\in\left[-\frac{1}{2},\frac{1}{2}\right] \\ 12(\frac{3}{2}-|x|)^2-6+8(x-1)^2+12(\frac{3}{2}-|x-2|)^2, & x\in\left[\frac{1}{2},\frac{3}{2}\right] \\ -4(\frac{3}{2}-|x-1|)^2+18-24(x-2)^2, & x\in\left[\frac{3}{2},\frac{5}{2}\right] \\ 12\left(\frac{3}{2}-|x-2|\right)^2, & x\in\left[\frac{5}{2},\frac{7}{2}\right] \end{cases}$$

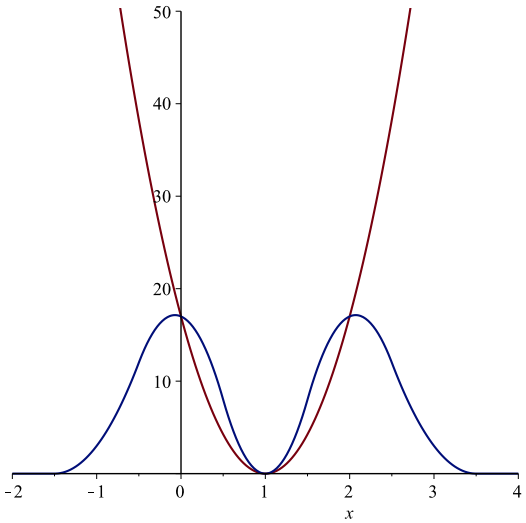
Evaluating this function at the desired points yields

$$I\left(\frac{3}{2}\right)=8, \quad I(2)=17, \quad I(3)=3, \quad I(4)=0$$

(c) The exact values at these points would be

$$f\left(\frac{3}{2}\right)=4.5, \quad f(2)=17, \quad f(3)=68, \quad f(4)=153$$

Consider the following graph that depicts the interpolant $I(x)$ in blue and the original function $f(x)$ in red, since this helps to understand the observed differences between the original and the interpolated values.



There is a fundamental difference between the approximation quality inside of the interpolation domain $[0, 2]$ and the points outside of this interval. Inside the interpolation domain, the interpolation is far from perfect (we only used a low number of samples), but the general shape of the interpolant fits fairly well. Outside of the interpolation domain, there is no similarity between the interpolant and the original function anymore. Since the synthesis functions have limited support, the interpolant also vanishes outside of this support.

These observations can also be seen from the four values that had to be evaluated in (b) and (c). The only location where the interpolation is perfect is at $I(2) = f(2) = 17$, which is clear, since the interpolation condition is met there. For $\frac{3}{2}, 3$ and 4, the interpolation error (i.e. the difference between original and interpolated value) is increasing with the distance from $[0, 2]$. For $\frac{3}{2}$, to of the shifted synthesis functions contribute to the interpolated value, for 3 only 1 and 4 is already outside of the support of the interpolant.

Note that in practice, quadratic splines are usually not used since cubic splines offer a better interpolant at comparable computational cost.

Problem 3 (Affine Rescaling)

(a) In order to apply the affine transformation $f(x) = mx + n$, we have to determine the slope m and the offset n first. Since we know that $F(x_{min}) = a$ and $F(x_{max}) = b$ we have two linear equations for the two unknowns. Solving this 2×2 linear system of equations yields $m = \frac{b-a}{x_{max}-x_{min}}$ and $n = a - x_{\min}\frac{b-a}{x_{max}-x_{min}}$. This allows us to implement the code via

```
/* ---------------------------------------------------------- */
void rescale
     (double  **u,        /* input image, range [0,255] */
      long    nx,         /* size in x direction */
      long    ny,         /* size in y direction */
      double  a,          /* smallest transformed grey level */
      double  b,          /* largest transformed grey level */
      double  *g)         /* transformed grey levels */
/*
 affine rescaling of the grey values of u such that
 min(u) -> a, and max(u) -> b.
*/
{
```

```
long    i, j;       /* loop variables */
double  min, max;   /* extrema of u */
double  m,n;        /* time saver */

/* determine extrema of u */
min = max = u[1][1];
for (i = 1; i <= nx; i++)
{
  for (j = 1; j <= ny; j++)
  {
    if (u[i][j] < min) min = u[i][j];
    if (u[i][j] > max) max = u[i][j];
  }
}


/* compute parameters */
m = (b-a) / (max-min);
n = a-min*m;

/* determine mapping for all grey values */
for (i = 0; i <= 255; i++)
    g[i] = m*i+n;
}
/* ------------------------------------------------------------ */
```

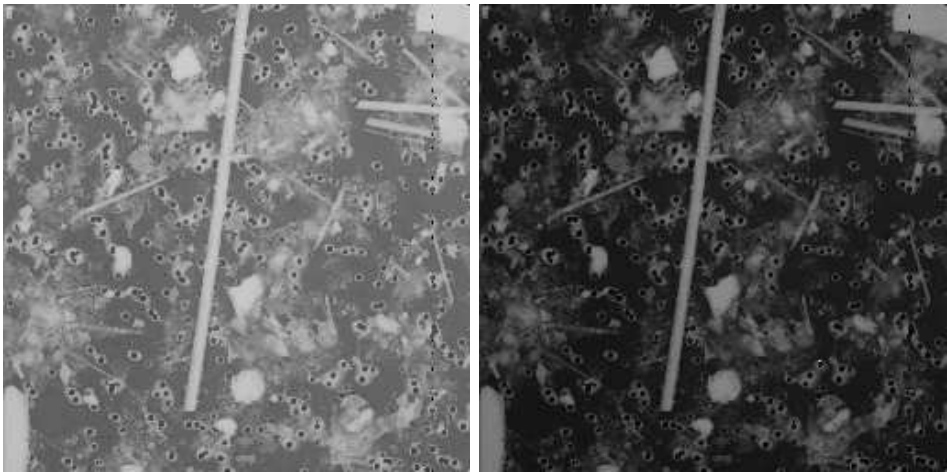
(b) Applying the affine transformation to the image `machine.pgm` yields (left: original, right: transformed image ($a = 0$, $b = 255$)).

Problem 4 (Gamma Correction)

(a) The second task is to implement the gamma corrections from the lecture. The corresponding code is given by

```
/* ------------------------------------------------------------ */
void gamma_correct

     (double  gamma,      /* gamma correction factor */
      double  *g)         /* transformed grey levels */

/*
 applies gamma correction to the 256 grey levels that may appear
 in byte wise coded images
*/

{
long  k;   /* loop variable */

/* determine mapping for all grey values */
for (k = 0; k <= 255; k++)
    g[k] = 255.0 * pow(k/255.0, 1.0/gamma);

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

(b) Applying the gamma correction to the image `asbest.pgm` yields (left: original, right: transformed image ($\gamma = 0.4$)).



Problem 5 (Histogram Equalisation)

(a) Here, we implement the histogram equalisation from the lecture. Two algorithms are presented. The first one maps the grey values directly if the inequality holds (early map). The second one is closer to the lecture and maps the grey values *en block* after the inequality is violated (late map).

```
/* ------------------------------------------------------------ */
void hist_equal_early_map

     (double  **u,        /* input image, range [0,255] */
      long    nx,         /* size in x direction */
      long    ny,         /* size in y direction */
      double  *g)         /* transformed grey levels */

/* performs histogram equalization on u. */

{
long    i, j, k;          /* loop variables */
long    r;                /* current summation index r */
long    k_r;              /* current summation index k_r */
double  hist[256];        /* histogram */
long    n;                /* pixel number */
double  psum, qsum;       /* sums in equalisation algorithm */

/* initialise histogram with zero */
```

```
for (k = 0; k <= 255; k++)
    hist[k]=0;

/* create histogram of u with bin width 1 */
for (i = 1; i <= nx; i++)
  for (j = 1; j <= ny; j++)
    hist[(long) u[i][j]]++;

/* initialise psum and k_r */
k_r = 1;
psum = hist[0];

/* total number of pixels */
n = nx*ny;

for (r = 1; r <= 256; r++)
{
  /* determine right hand side */
  qsum = r * n / 256.0;

  /* increase left hand side as long as it is smaller or equal */
  /* the right hand side; note that each k_r is only used once.*/
  for (; (k_r <= 256) && (psum <= qsum); k_r++)
  {
    /* perform mapping : we have to substract 1 from the grey */
    /* value index r and k_r to get the actual grey values */
    g[k_r-1] = r - 1;

    /* increase left hand side */
    psum += hist[k_r];

  }
}
}

/* ------------------------------------------------------------ */


/*--------------------------------------------------------------*/

void hist_equal_late_map

     (double  **u,        /* input image, range [0,255] */
      long    nx,         /* size in x direction */
```

```
      long    ny,         /* size in y direction */
      double  *g)         /* transformed grey levels */

/* performs histogram equalization on u. */

{
long    i, j, k;          /* loop variables */
long    r;                /* current summation index r */
long    k_r;              /* current summation index k_r */
long    n;                /* pixel number */
double  hist[256];        /* histogram */
double  psum, qsum;       /* sums in equalisation algorithm */

/* initialise histogram with zero */
for (k = 0; k <= 255; k++)
    hist[k]=0;

/* create histogram of u with bin width 1 */
for (i = 1; i <= nx; i++)
  for (j = 1; j <= ny; j++)
    hist[(long) u[i][j]]++;

/* initialise psum and k_r */
k_r  = 1;
psum = hist[0];

/* total number of pixels */
n = nx*ny;

/* j is the index of the last grey value that has been mapped */
j   = 0;

for(r=1; r<=256; r++)
{
  /* determine right hand side */
  qsum = r * n / 256.0;

  /* search the largest index k_r such that the inequality */
  /* is satisfied */
  while(k_r <= 256 && psum <= qsum)
  {
    psum += hist[k_r];
    k_r  = k_r+1;
```

```
  }

  /* perform mapping from j which corresponds to k_{r-1} to k_r-1.*/
  /* we have to subtract 1 from the grey value index r and k_r */
  /* to get the actual grey values */
  while(j < k_r-1)
  {
    /* perform mapping */
    g[j] = r-1;

    /* store last index that has been mapped (=k_{r-1}) */
    j    = j+1;
  }
}
}
/*-------------------------------------------------------------*/
```

(b) Applying the histogram equalisation to the image `office.pgm` yields (left: original, right: transformed)