DigPro1 Zusammenfassung

Dozent: G.Schuster, Buch: Digital Image Processing, Rafael C. Gonzalez and Richard E. Woods

J.Rast, S.Körner, C.Gwerder, H.Badertscher

February 10, 2014

Contents

1. Digital image fundamentals p.35

intensity values $L = 2^k$ with k-bit
bits to save $b = M \cdot N \cdot k$

1.1. Image Interpolation p.65

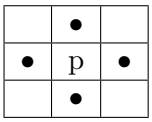
- Basic tool needed for zooming, shrinking, rotation, ...
- Nearest Neighbor Interpolation:
  - Each pixel in the new resolution gets the value of the nearest pixel in the old resolution.
  - Simple but bad
- Bilinear interpolation:
  - *Not* linear
  - uses the four nearst neighbors of a point (x, y)
  - $v(x, y) = ax + by + cxy + d$
- Bicubic interpolation
  - Uses the 16 nearest neighbors of a point (x, y)
  - Note that if the sums go from 0 to 1, this reduces to the bilinear interpolation
  - $v(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j$

1.2. Some Basic Relationships between pixels p.68

1.2.1. Neighbors of a Pixel p.68

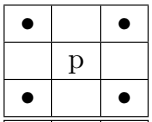
A pixel p at coordinates (x,y) has four horizontal and vertical neighbors, called the 4-neighbors of p, denoted by $\mathbf{N_4(p)}$ have the coordinates:

$$(x+1, y), (x-1, y), (x, y+1), (x, y-1)$$

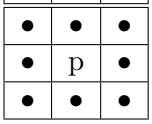


The four diagonal neighbors of p, denoted by $\mathbf{N_D(p)}$, have the coordinates

$$(x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1)$$



and are $N_D(p)$ and $N_4(p)$ together are called the 8-neighbors of p, denoted by $N_8(p)$.



1.2.2. Adjacency, connectivity, regions and boundaries

4-adjacency: Two pixels p and q with values of V (binary V = {1}) are 4-adjacency if q is in the set $N_4(p)$

8-adjacency: Two pixels p and q with values of V are 8-adjacency if q is in the set $N_8(p)$

m-adjacency: (mixed-adjacency) Two pixels p and q with values of V are m-adjacency if

1. q is in $N_4(p)$ or
2. q is in $N_D(p)$ and the set $N_4(p) \cap N_4(q)$ has no pixel whose values are from V
   $\Rightarrow$ No closed paths

1.2.3. Distance measures / Neighbors p.71

For pixels p, q with coordinates $(x, y), (s, t)$

Euclidean

$$D_e(p,q) = [(x-s)^2 + (y-t)^2]^{\frac{1}{2}} \tag{1.1}$$

City-block

$$D_4(p,q) = |x-s| + |y-t| \tag{1.2}$$

Pixels with a $D_4 = 1$ are the 4-neighbors of (x, y)

Chessboard

$$D_8(p,q) = max(|x-s|, |y-t|) \tag{1.3}$$

Pixels with a $D_8 = 1$ are the 8-neighbors of (x, y)

1.3. Math tools p.72

1.3.1. Noise reduction p.75

Image with noise

$$g(x,y) = f(x,y) + \eta(x,y) \tag{1.4}$$

Noise η is uncorrelated and has zero average. Averaging over K different images reduces noise.

$$\begin{aligned} \bar{g}(x,y) &= \frac{1}{K}\sum_{i=1}^{K} g_i(x,y) \\ E\{\bar{g}(x,y)\} &= f(x,y) \\ \sigma^2_{\bar{g}(x,y)} &= \frac{1}{K}\sigma^2_{\eta(x,y)} \end{aligned} \tag{1.5}$$

1.3.2. Set and Logical Operations p.80

Subset	$A \subseteq B$	Every element of A is also in B
Union	$C = A \cup B$	C contains all Elements in A, B or both
Intersection	$D = A \cap B$	D contains all Elements wich are in A and B
Complement	$A^c = \{\omega \mid \omega \in A\}$	Set of elements that are not in A
Difference	$A - B = \{\omega \mid \omega \in A, \omega \notin B\} = A \cap B^c$	

1.3.3. Neighborhood operations p.85

Averaging of neighborhood S_{xy}

$$g(x,y) = \frac{1}{mn}\sum_{(r,c)\in S_{xy}} f(r,c) \tag{1.6}$$

1.3.4. Geometric spatial Transformations p.87

$$(x,y) = T\{(v,w)\}$$

affine transform: $[x\ y\ 1] = [v\ w\ q]\mathbf{T} = [v\ w\ 1]\begin{bmatrix} t_{11} & t_{12} & 0 \\ t_{21} & t_{22} & 0 \\ t_{31} & t_{32} & 1 \end{bmatrix}$

- Spatial transformation of coordinates
- Intensity interpolation for transformed pixels
- See p.88 for examples of **T**!

1.3.5. Image registration p.89

- Tries to align several images of the same scheme
- Using **tie points**, whose location are known precisely

2. Intensity Transformations and Spatial Filtering p.104

2.1. Basic intensity transformation functions p.105

General intensity transformation function:

$$s = T(r) \tag{2.1}$$

- Image negatives

$$s = L - 1 - r \tag{2.2}$$

- Log transformations

$$s = c \cdot \log(1 + r) \tag{2.3}$$

- Inverse log transformations
- Power-law (Gamma) transformations

$$s = cr^{\gamma} \tag{2.4}$$

- Piecewise-linear transformation functions p.115:
  - contrast stretching
  - intensity level slicing
- Bit-plane slicing:
  Split a image in slices for each intensity bit. All resulting plane are binary images. May be used for a simple compression.

2.2. Histogram processing p.120

The histogram shows the number of occurrence of a particular intensity level relative to the total number of pixels (probability of the intensity value).
A normalized histogram is given by $p(r_k) = n_k/MN$ for $k = 0, 1, 2, \ldots, L-1$.

2.2.1. Histogram equalization p.122

The intensity level in an image may be viewed as random variables in the interval $[0, L-1]$.

For continuous Values

$$s = T(r) = (L-1)\int_0^r p_r(\omega)d\omega \tag{2.5}$$

For discrete Values

$$s_k = T(r_k) = (L-1)\sum_{j=0}^{k} p_r(r_j) = \frac{L-1}{MN}\sum_{j=0}^{k} n_j \qquad k = 0, 1, 2, \ldots, L-1 \tag{2.6}$$

where MN is the total number of pixels, n_k is the number of pixels that have the intensity r_k and L is the number of possible intensity levels in the image. A plot of $p_r(r_k)$ versus r_k is commonly referred to as a histogram.

2.2.2. Histogram Matching (Specification) p.128

If we want to highlight only some intensity levels we need other methods than histogram equalization.
$p_z(z)$ is the specified PDF, we wish the output image to have.

For continuous Values

$$s = T(r) = (L-1)\int_0^r p_r(\omega)d\omega \tag{2.7}$$

$$G(z) = (L-1)\int_0^z p_z(t)dt = s \tag{2.8}$$

$$z = G^{-1}[T(r)] = G^{-1}(s) \tag{2.9}$$

1. Obtain $p_r(r)$ from the original image and use Eq.2.7 to obtain the value of s.
2. Use specified PDF $p_z(z)$ in Eq.2.8 to obtain the transformation function $G(z)$
3. Obtain the inverse transformation $z = G^{-1}(s)$
4. Obtain the output image by first equalizing the original image using Eq.2.7; the pixels values in this image are the s values. For each pixel witch value s in the equalized image, perform the inverse mapping $z = G^{-1}(s)$ to obtain the corresponding pixel in the output image. $\Rightarrow$ PDF of output image is equal to specified PDF.

For discrete Values

$$s_k = T(r_k) = \frac{L-1}{MN}\sum_{j=0}^{k} n_j \qquad k = 0, 1, 2, \ldots, L-1 \tag{2.10}$$

$$G(z_q) = (L-1)\sum_{i=0}^{q} p_z(z_i) = s_k \qquad q = 0, 1, 2, \ldots, L-1 \tag{2.11}$$

$$z_q = G^{-1}(s_k) \tag{2.12}$$

1. Compute the histogram $p_r(r_k)$ of the given image.
2. Find the histogram equalization transformation (Eq.2.10), round the resulting s_k values to the integer range $[0, L-1]$
3. Compute all values of the transformation function G using Eq.2.11, where $p_z(z_i)$ are the values of the specified histogram. Round the values of G to integers and store them in a table
4. For every value of s_k use the stored values of G to find the corresponding value of z_q so that $G(z_q)$ is closest to s_k and store these mappings from s to z.
5. Form the output image by first histogram-equalizing the input image and then mapping every equalized pixel value, s_k, of this image to the corresponding value z_q in the output image using the mappings found in step 4. Note: T and G^{-1} can also be combined $\Rightarrow r_k \rightarrow z_q$

2.2.3. Local Histogram Processing p.139

The techniques described above are easily adapted to local enhancement. The procedure is to define a neighborhood and move its center from pixel to pixel. At each location, the histogram of the neighborhood is computed and a transformation function is obtained. This function is then used to map the intensity of the pixel centered in the neighborhood. See Fig. 3.36(c) p.140 for local histogram equalization with a 3x3 neighborhood.

2.2.4. Using Histogram Statistics for Image Enhancement p.139

The stated formulas can be used to calculate the global mean and variance. Equations for local mean and variance in a specified neighborhood can easily be found in a similar way.

mean (average intensity):

$$m = \sum_{i=0}^{L-1} r_i p(r_i) \tag{2.13}$$

n-th movement (intensity variance $\mu_2 = \sigma^2$):

$$\mu_n(r) = \sum_{i=0}^{L-1} (r_i - m)^n p(r_i) \tag{2.14}$$

sample mean:

$$m = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \tag{2.15}$$

sample variance:

$$\sigma^2 = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [f(x,y) - m]^2 \tag{2.16}$$

This information of the picture can be used for enhancing the image. Eq. 3.3-24 p.144 is an example of image enhancement with usage of mean and variance.

2.3. Spatial filtering p.144

A spatial filter consists of

1. neighborhood (typically a small rectangle) of $f(x,y)$
2. a predefined operation, filter mask $w(x,y)$

If the operation is linear its a *linear spatial filter* otherwise its *nonlinear*. Linear filters may be described by a correlation of a filter $w(x,y)$ and the picture $f(x,y)$. (Not a convolution!)

$$\text{correlation:} \quad w(x,y) \overset{\star}{\square} f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t) \tag{2.17}$$

$$\text{convolution:} \quad w(x,y) \star f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x-s, y-t) \tag{2.18}$$

Correlation and convolution can be interchanged by rotating the filter by 180°.

Zero padding

filter w of size $m \times n \Rightarrow$ Pad the image with a minimum of $m-1$ rows of 0s at top & bottom and with $n-1$ columns of 0s on left & right

2.3.1. 2D Convolution, a fast way to calculate

Shown on an example:

1	2	3
4	5	6
7	8	9

Input

$\star$

-1	-2	-1
0	0	0
1	2	1

Kernel

$\Rightarrow$

1 · 0	2 · 0	1· 0	0	0	0	0
0· 0	0· 0	0· 0	0	0	0	0
-1· 0	-2· 0	-1 · 1	2	3	0	0
0	0	4	5	6	0	0
0	0	7	8	9	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

Step 1

$=$

-1	-4	-8	-8	-3
-4	-13	-20	-17	-6
-6	-18	-24	-18	-6
4	13	20	17	6
7	22	32	26	9

Output

2.3.2. Vector Representation of Linear Filtering

When interest lies in the characteristic response R, it the sum of products can be written as

$$R = w_1 z_1 + w_2 z_2 + \cdots + w_{mn} z_{mn} = \sum_{k=1}^{mn} w_k z_k \tag{2.19}$$

$$= \mathbf{w}^T \mathbf{z} \tag{2.20}$$

where $\mathbf{w}$ and $\mathbf{z}$ are vectors formed from the coefficients of the mask and the image.

2.3.3. Smoothing spatial filters p.152

The output of a smoothing, linear spatial filter is the average of the pixels contained in the neighborhood of the filter mask. Its main application is blurring and thus reducing noise in images.

$$g(x,y) = \frac{\sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t)}{\sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t)} \tag{2.21}$$

box filter A filter which all coefficients are equal: $\frac{1}{9} \cdot$

1	1	1
1	1	1
1	1	1

weighted average Giving more importance to some pixels: $\frac{1}{16} \cdot$

1	2	1
2	4	2
1	2	1

2.3.4. Sharpening spatial filters p.157

This filter highlights transitions in intensity, thus sharpening the image. It is the opposite of the smoothing filter and since they have a integrative effect it follows that sharpening filters have a derivative effect.

$$\frac{\partial f}{\partial x} = f(x+1) - f(x) \qquad \text{first order derivative} \tag{2.22}$$

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x) \qquad \text{second order derivative} \tag{2.23}$$

Second Derivative - Laplacian p.160 The Laplacian is a isotropic (rotation invariant) derivative operator

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{2.24}$$

$$\nabla^2 f = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y) \tag{2.25}$$

0	1	0
1	-4	1
0	1	0

Implementation with filter masks (rotation invariant for 90° rotations)

1	1	1
1	-8	1
1	1	1

Incorporating the diagonal directions (rotation invariant for 45° rotations)

To sharpen an image we add the Laplacian to the original

$$g(x,y) = f(x,y) + c\left[\nabla^2 f(x,y)\right] \qquad \text{with } c = -1 \tag{2.26}$$

Unsharp Masking / Highboost Filtering p.162

1. Blur the original $\bar{f}$
2. Subtract the blurred image from the original (gives the mask g_{mask})
3. Add the mask to the original

$$g_{mask}(x,y) = f(x,y) - \bar{f}(x,y) \tag{2.27}$$
$$g(x,y) = f(x,y) + k \cdot g_{mask}(x,y) \tag{2.28}$$

with $k = 1$ this filter is called unsharp masking, with $k > 1$ highboost.

First-Order Derivative - Gradient p.165 The first derivative in an image is the magnitude of the gradient. The gradient of image f at (x,y) is a two-dimensional *vector* pointing in the direction of the greatest rate of change.

$$\nabla f \equiv grad(f) \equiv \begin{pmatrix} g_x \\ g_y \end{pmatrix} = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{pmatrix} \tag{2.29}$$

Gradient Image:
We are interested in the magnitude of the gradient, is the value of the rate of change into the direction of the gradient.

$$M(x,y) = \sqrt{g_x^2 + g_y^2} \tag{2.30}$$

- isotropic but not linear
- $M(x,y)$ is called the gradient image
- a popular approximation is
$$M(x,y) \approx |g_x| + |g_y|$$

The gradient can be approximated using filter masks.

- Roberts cross gradient operators
  - $g_x = \begin{array}{|c|c|}\hline -1 & 0 \\ \hline 0 & 1 \\ \hline \end{array} \star f(x,y)$ $\quad g_y = \begin{array}{|c|c|}\hline 0 & -1 \\ \hline 1 & 0 \\ \hline \end{array} \star f(x,y)$
  - no center of symmetry → difficult to implement
- **Sobel operators**, often used
  - $g_x = \begin{array}{|c|c|c|}\hline -1 & -2 & -1 \\ \hline 0 & 0 & 0 \\ \hline 1 & 2 & 1 \\ \hline \end{array} \star f(x,y)$ $\quad g_y = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array} \star f(x,y)$
  - 3x3 neighborhood → smoothing

2.4. Fuzzy Techniques for Intensity Transformations and Spatial Filtering p.173

3. Filtering in the frequency domain p.199

3.1. Countinous Fourier Transform p.205

3.2. Sampling p.211

Multiplication of the function with an impulse train

$$\tilde{f}(t) = f(t)s_{\Delta T}(t) = \sum_{n=-\infty}^{\infty} f(t)\sigma(t - n\Delta T) \tag{3.1}$$

The fourier transform of it

$$\tilde{F}(\mu) = F(\mu) \star S(\mu) = \frac{1}{\Delta T} \sum_{n=-\infty}^{\infty} F\left(\mu - \frac{n}{\Delta T}\right) \tag{3.2}$$

3.3. The discrete Fourier transform of one variable p.220

The Fourier transform of sampled data is continuous and infinitely periodic with period $1/\Delta T$

$$\tilde{F}(\mu) = \sum_{n=-\infty}^{\infty} f_n \cdot e^{-j2\pi\mu n\Delta T} \tag{3.3}$$

We are only interested in one period. So we obtain M equally spaced samples of $\tilde{F}(\mu)$ over the period $\mu = 0$ to $\mu = 1/\Delta T$

$$\mu = \frac{m}{M\Delta T} \tag{3.4}$$

This results in

$$F_m = \sum_{n=0}^{M-1} f_n \cdot e^{-j2\pi mn/M} \quad m = 0, 1, 2, \ldots, M-1 \tag{3.5}$$

The *inverse discrete Fourier transform* is used to recover the sample set f_n

$$f_n = \frac{1}{M} \sum_{m=0}^{M-1} F_m e^{j2\pi mn/M} \quad n = 0, 1, 2, \ldots, M-1 \tag{3.6}$$

- The discrete Fourier transform is an invertible, linear transformation
- The DFT pair is applicable to *any* finite set of uniformly discrete samples
- DFT and IDFT are periodic with period M

3.3.1. Sampling and Frequency Intervals p.223

If $f(x)$ consists of M samples, taken ΔT units apart, the duration T of the record is

$$T = M\Delta T \tag{3.7}$$

The corresponding spacing in the discrete frequency domain Δu is

$$\Delta u = \frac{1}{M\Delta T} = \frac{1}{T} \tag{3.8}$$

The frequency range spanned by the M components of the DFT is

$$\Omega = M\Delta u = \frac{1}{\Delta T} \tag{3.9}$$

3.4. 2D Countinous FT p.205

3.5. The 2D Impulse p.225

The discrete 2D impulse is defined as

$$\delta(x,y) = \begin{cases} 1 & \text{if } x = y = 0, \\ 0 & \text{otherwise} \end{cases} \tag{3.10}$$

with the sifting property

$$\sum_{x=-\infty}^{\infty} \sum_{y=-\infty}^{\infty} f(x,y)\delta(x,y) = f(0,0) \tag{3.11}$$

3.6. 2D Sampling Theorem

A continuous, band-limited function $f(t,z)$ can be recovered with no error if the sampling intervals are

$$\Delta T < \frac{1}{2u_{max}} \qquad \text{and} \qquad \Delta Z < \frac{1}{2v_{max}}$$

3.7. 2D DFT and IDFT p.235

3.7.1. 2D DFT

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \cdot e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)} \tag{3.12}$$

with u and v in the ranges $u = 0,1,2,\ldots,M-1$ and $v = 0,1,2,\ldots,N-1$

3.7.2. 2D IDFT

$$f(x,y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) \cdot e^{j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)} \tag{3.13}$$

with x and y in the ranges $x = 0,1,2,\ldots,M-1$ and $y = 0,1,2,\ldots,N-1$

3.7.3. Properties

- Separation between samples in the frequency domain are inversely proportional to the spacing between spatial samples and the number of samples $\Delta u = \frac{1}{M\Delta T} \quad \Delta v = \frac{1}{N\Delta Z}$
- Infinitely periodic
- Symmetry properties p.242, see Table 1 (Appendix)
- Summary of properties p.253, see Table 2 and 3 (Appendix)

3.7.4. Periodicity 237

Visualization is simplified if we shift the data so that F(0,0) at (M/2,N/2) with

$$f(x,y)(-1)^{x+y} \Leftrightarrow F(u - M/2, v - N/2) \tag{3.14}$$

3.7.5. "DC"

$$|F(0,0)| = MN|\bar{f}(x,y)| \tag{3.15}$$

3.7.6. 2-D Convolution p.249

The 2-D Convolution Theorem is given by:

$$f(x,y) \star h(x,y) \Leftrightarrow F(u,v)H(u,v) \tag{3.16}$$

It is necessairy to zero-pad the images $f(x,y)$ of size $A \times B$ and $h(x,y)$ of size $C \times D$ as follows:

$$f_p(x,y) = \begin{cases} f(x,y) & 0 \leqslant x \leqslant A-1 \text{ and } 0 \leqslant y \leqslant B-1 \\ 0 & A \leqslant x \leqslant P \text{ or } B \leqslant y \leqslant Q \end{cases} \tag{3.17}$$

$$h_p(x,y) = \begin{cases} h(x,y) & 0 \leqslant x \leqslant C-1 \text{ and } 0 \leqslant y \leqslant D-1 \\ 0 & C \leqslant x \leqslant P \text{ or } D \leqslant y \leqslant Q \end{cases} \tag{3.18}$$

with

$$P \geqslant A + C - 1 \tag{3.19}$$

$$Q \geqslant B + D - 1 \tag{3.20}$$

resulting in padded images of size $P \times Q$

3.8. Basics of Filtering in the Frequency Domains

Generally, filtering in the frequency domain is obtained by

$$g(x,y) = \Im^{-1}\left[H(u,v)F(u,v)\right] \tag{3.21}$$

if filter $H(u,v)$ is real & symmetric

$$g(x,y) = \Im^{-1}\left[H(u,v) \cdot R(u,v) + jH(u,v) \cdot I(u,v)\right] \tag{3.22}$$

$\Rightarrow$ Filter that affect real and imaginary parts equally have no effect on the phase $\rightarrow$ **zero-phase-shift** filters

3.8.1. Summary of Steps for Filtering in the Frequency Domain p.263

1. Given an input image $f(x,y)$ of size $M \times N$, obtain the padding parameters P and Q from Eq. 3.20. Typically, we select $P = 2M$ and $Q = 2N$.
2. Form a padded image, $f_p(x,y)$, of size $P \times Q$ by appending the necessary number of zeros to $f(x,y)$.
3. Multiply $f_p(x,y)$ by $(-1)^{x+y}$ to center its transform.
4. Compute the DFT, $F(u,v)$, of the image from step 3.
5. Generate a real, symmetric filter function, $H(u,v)$, of size $P \times Q$ with center at coordinates $(P/2, Q/2)$. Form the product $G(u,v) = H(u,v)F(u,v)$ using array multiplication.
6. Obtain the processed image, $g_p(x,y) = \left\{\text{real}\left[\Im^{-1}\left[G(u,v)\right]\right]\right\}(-1)^{x+y}$, where the real part is selected in order to ignore parasitic complex components resulting from computational inaccuracies, and the subscript p indicates that we are dealing with padded arrays.
7. Obtain the final processed result, $g(x,y)$, by extracting the $M \times N$ region from the top, left quadrant of $g_p(x,y)$.

3.9. Image smoothing using frequency domain filters p.269

3.9.1. Ideal Lowpass Filters

$$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \leqslant D_0 \\ 0 & \text{if } D(u,v) > D_0 \end{cases} \tag{3.23}$$

where D_0 is a positive constant and $D(u,v)$ is the distance between a point (u,v) in the frequency domain and the center of the frequency rectangle; that is,

$$D(u,v) = \left[(u - P/2)^2 + (v - Q/2)^2\right]^{1/2} \tag{3.24}$$

ILPFs have a **huge ringing problem** and are not frequently used.

3.9.2. Butterworth Lowpass Filters

The transfer function of a Butterworth lowpass filter (BLPF) of order n, and with cutoff frequency at a distance D_0 from the origin, is defined as

$$H(u,v) = \frac{1}{1 + \left[D(u,v)/D_0\right]^{2n}} \tag{3.25}$$

The BLPF of order 1 has no ringing, but ringing can become significant in filters of higher order. Usually only BLPFs with $n \leqslant 4$ are used.

3.9.3. Gaussian Lowpass Filters

A Gaussian lowpass filter (GLPF) of two dimensions is given by

$$H(u,v) = e^{-D^2(u,v)/2D_0^2} \tag{3.26}$$

where D_0 is the cutoff frequency. When $D(u,v) = D_0$, the GLPF is down to 0.607 of its maximum value. The GLPF will have **no ringing**.

3.10. Image sharpening using frequency domain filters p.280

A highpass filter is obtained from a given lowpass filter by

$$H_{HP}(u,v) = 1 - H_{LP}(u,v) \tag{3.27}$$

3.10.1. Ideal Highpass Filters

A ideal highpass filter (IHPF) is defined as

$$H(u,v) = \begin{cases} 0 & \text{if } D(u,v) \leqslant D_0 \\ 1 & \text{if } D(u,v) > D_0 \end{cases} \tag{3.28}$$

where D_0 is the cutoff frequency and $D(u,v)$ is given by Eq. 3.24. Like the ILPF, the IHPF creates a large ringing.

3.10.2. Butterworth Highpass Filters

A 2-D Butterworth highpass filter (HBPF) of order n and cutoff frequency D_0 is defined as

$$H(u,v) = \frac{1}{1 + \left[D_0/D(u,v)\right]^{2n}} \tag{3.29}$$

3.10.3. Gaussian Highpass Filters

The transfer function of the Gaussian highpass filter (GHPF) with cutoff frequency at D_0 is given by

$$H(u,v) = 1 - e^{-D^2(u,v)/2D_0^2} \tag{3.30}$$

3.10.4. The Laplacian in the Frequency Domain p.286

The Laplacian can be implemented in the frequency domain using the filter

$$H(u,v) = -4\pi^2 D^2(u,v) \tag{3.31}$$

The Laplacian image is obtained as

$$\nabla^2 f(x,y) = \Im^{-1}\{H(u,v)F(u,v)\} \tag{3.32}$$

The enhanced image $g(x,y) = f(x,y) + c\nabla^2 f(x,y)$ with $c = -1$ can be achieved by

$$g(x,y) = \Im^{-1}\left\{\left[1 + 4\pi^2 D^2(u,v)\right] F(u,v)\right\} \tag{3.33}$$

Usually, $f(x,y)$ is normalized to the range $[0,1]$ and $\nabla^2 f(x,y)$ is divided by its maximum value, which brings it to the approximate range $[-1,1]$. The normalizing factor in the frequency domain is not easily computed, so the preferred frequency domain implementation is using the equation

$$g(x,y) = f(x,y) + c\nabla^2 f(x,y) \tag{3.34}$$

with $\nabla^2 f(x,y)$ computed using Eq. 3.32.

3.10.5. Unsharp Masking, Highboost Filtering, and High-Frequency-Emphasis Filtering p.288

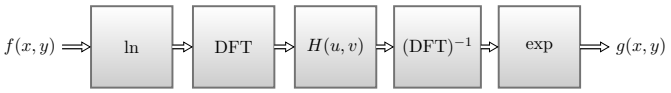
A high-frequency-emphasis filter is achieved by

$$g(x,y) = \Im^{-1}\{[k_1 + k_2 \cdot H_{HP}(u,v)]\, F(u,v)\} \tag{3.35}$$

where $k_1 \geqslant 0$ gives control of the offset from the origin and $k_2 \geqslant 0$ controlls the contribution of high frequencies.

3.10.6. Homomorphic Filtering p.290

Homomorphic filtering is used to separate illumination and reflectance components. This is achieved by using the natural logarithm ln before applying the DFT.



A modified version of a Gaussian highpass,

$$H(u,v) = (\gamma_H - \gamma_L)\left[1 - e^{-c\left[D^2(u,v)/D_0^2\right]}\right] + \gamma_L \tag{3.36}$$

allows to attenuate the low frequencies (illumination) with $\gamma_L < 1$ and amplify the high frequencies (reflectance) with $\gamma_H > 1$.

3.11. Selective Filtering p.294

3.11.1. Bandreject and Bandpass Filters

Bandreject filters are given by the following equations,

Ideal	Butterworth	Gaussian
$H(u,v) = \begin{cases} 0 & \text{if } D_0 - \frac{W}{2} \leqslant D \leqslant D_0 + \frac{W}{2} \\ 1 & \text{otherwise} \end{cases}$	$H(u,v) = \frac{1}{1+\left[\frac{DW}{D^2-D_0^2}\right]^{2n}}$	$H(u,v) = 1 - e^{-\left[\frac{D^2-D_0^2}{DW}\right]^2}$

where D_0 is the radial center of the band, W is the width of the band, and $D(u,v)$ is given by Eq. 3.24.

A bandpass filter is obtained by

$$H_{BP}(u,v) = 1 - H_{BR}(u,v) \tag{3.37}$$

3.11.2. Notch Filters

A notch filter rejects (or passes) frequencies in a predefined neighborhood about the center of the frequency rectangle. Notch reject filters are constructed as products of highpass filters whose centers have been translated to the centers of the notches. The general form is

$$H_{NR}(u,v) = \prod_{k=1}^{Q} H_k(u,v) H_{-k}(u,v) \tag{3.38}$$

where $H_k(u,v$ and $H_{-k}(u,v)$ are highpass filters whose centers are at (u_k, v_k) and $(-u_k, -v_k)$.

A notch pass filter is obtained from a notch reject filter by

$$H_{NP}(u,v) = 1 - H_{NR}(u,v) \tag{3.39}$$

3.12. Implementation p.298

4. Image restoration and reconstruction p.311

A degraded image is given in the spatial domain by

$$g(x,y) = h(x,y) \star f(x,y) + \eta(x,y) \tag{4.1}$$

where $h(x,y)$ is the degradation function, $f(x,y)$ is the original image and $\eta(x,y)$ is the noise.

4.1. Noise models p.313

4.1.1. Gaussian noise

Frequently used because of its mathematical tractability in spatial and frequency domain.

$$p(z) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(z-\overline{z})^2/2\sigma^2} \tag{4.2}$$

4.1.2. Rayleigh noise

$$p(z) = \begin{cases} \frac{2}{b}(z-a)\, e^{-(z-a)^2/b} & \text{for } z \geqslant a \\ 0 & \text{for } z < a \end{cases} \tag{4.3}$$

The mean and variance of this density are given by

$$\overline{z} = a + \sqrt{\pi b/4} \tag{4.4}$$

$$\sigma^2 = \frac{b(4-\pi)}{4} \tag{4.5}$$

4.1.3. Erlang (gamma) noise

$$p(z) = \begin{cases} \frac{a^b z^{b-1}}{(b-1)!} e^{-az} & \text{for } z \geqslant 0 \\ 0 & \text{for } z < 0 \end{cases} \tag{4.6}$$

where the parameters are such that $a > 0$ and b is a positive integer. The mean and variance of this density are given by

$$\overline{z} = \frac{b}{a} \tag{4.7}$$

$$\sigma^2 = \frac{b}{a^2} \tag{4.8}$$

4.1.4. Exponential noise

$$p(z) = \begin{cases} ae^{-az} & \text{for } z \geqslant 0 \\ 0 & \text{for } z < 0 \end{cases} \tag{4.9}$$

where $a > 0$. The mean and variance of this density are given by

$$\overline{z} = \frac{1}{a} \tag{4.10}$$

$$\sigma^2 = \frac{1}{a^2} \tag{4.11}$$

4.1.5. Uniform noise

$$p(z) = \begin{cases} \frac{1}{b-a} & \text{if } a \leqslant z \leqslant b \\ 0 & \text{otherwise} \end{cases} \tag{4.12}$$

The mean and variance of this density are given by

$$\overline{z} = \frac{a+b}{2} \tag{4.13}$$

$$\sigma^2 = \frac{(b-a)^2}{12} \tag{4.14}$$

4.1.6. Impulse (salt-and-pepper) noise

$$p(z) = \begin{cases} P_a & \text{for } z = a \\ P_b & \text{for } z = b \\ 0 & \text{otherwise} \end{cases} \tag{4.15}$$

4.1.7. Estimation of noise parameters

- Select a strip S of the image, with constant background.
  $\Rightarrow$ The histogram in these areas gives a first hint about the noise corrupting the image
- The mean and variance of the noise can be estimated by

$$\overline{z} = \sum_{i=0}^{L-1} z_i p_S(z_i) \tag{4.16}$$

$$\sigma^2 = \sum_{i=0}^{L-1} (z_i - \overline{z})^2 p_S(z_i) \tag{4.17}$$

  where $p_S(z_i)$ with $i = 0, 1, \ldots, L-1$ denotes the probability estimates of the intensity values.
- The shape identifies the noise model which matches the best $\Rightarrow$ χ^2-Test

4.2. Restoration in the presence of noise only - spatial filtering p.322

$$g(x,y) = f(x,y) + \eta(x,y) \quad \Leftrightarrow \quad G(u,v) = F(u,v) + N(u,v)$$

Spatial filtering (low pass) is the method of choise in situations when only addative random noise is present.

4.2.1. Mean Filters

Arithmetic mean filter
This filter is a standard filter.

$$\hat{f}(x,y) = \frac{1}{mn} \sum_{(s,t) \in S_{xy}} g(s,t) \tag{4.18}$$

Geometric mean filter
Smooth similar to arithmetic mean, but less loss of detail.

$$\hat{f}(x,y) = \left[\prod_{(s,t) \in S_{xy}} g(s,t) \right]^{\frac{1}{mn}} \tag{4.19}$$

Harmonic mean filter
Works well on salt noise, but fails for pepper noise. It does well also with other types of noise like Gaussian noise

$$\hat{f}(x,y) = \frac{mn}{\sum\limits_{(s,t)\in S_{xy}} \frac{1}{g(s,t)}} \tag{4.20}$$

Contraharmonic mean filter

- For positive Q, good for pepper noise
- For negative Q, good for salt noise
- For Q = -1, it is a harmonic mean filter
- For Q = 0, it is a arithmetic mean filter

This filter can not simultaneously reduce salt and pepper noise

$$\hat{f}(x,y) = \frac{\sum\limits_{(s,t)\in S_{xy}} g(s,t)^{Q+1}}{\sum\limits_{(s,t)\in S_{xy}} g(s,t)^{Q}} \tag{4.21}$$

4.2.2. Order-Statistic Filters p.325

The output of these filters depends on the **order of the pixel** values.

Median filter
Quite popular, since they result in good noise suppression, without much smoothing

$$\hat{f}(x,y) = \underset{(s,t)\in S_{xy}}{\text{median}}\{g(s,t)\} \tag{4.22}$$

Max filter
This filter picks the maximum of the neighborhood, also it reduce the pepper noise.

$$\hat{f}(x,y) = \max_{(s,t)\in S_{xy}} \{g(s,t)\} \tag{4.23}$$

Min filter
This filter picks the minimum of the neighborhood, also it reduce the salt noise.

$$\hat{f}(x,y) = \min_{(s,t)\in S_{xy}} \{g(s,t)\} \tag{4.24}$$

Midpoint filter
The midpoint filter simply computes the midpoint between the maximum and the minimum values in the neighborhood.
It works best for randomly distributed noise, like Gaussian or uniform noise.

$$\hat{f}(x,y) = \frac{1}{2}\left[\max_{(s,t)\in S_{xy}} \{g(s,t)\} + \min_{(s,t)\in S_{xy}} \{g(s,t)\}\right] \tag{4.25}$$

Alpha-trimmed filter
It is robust against outliers and it is good against "normal" noise.

Suppose that we delete the $d/2$ lowest and $d/2$ highest intensity values of $g(s,t)$ in the neighborhood S_{xy}. Let $g_r(s,t)$ represent the remaining $mn - d$ pixels.

- d can range from 0 to $mn - 1$
- When $d = 0$, then it is a arithmetic mean filter
- When $d = mn - 1$, then the filter becomes a median filter

$$\hat{f}(x,y) = \frac{1}{mn-d} \sum_{(s,t)\in S_{xy}} g_r(s,t) \tag{4.26}$$

4.2.3. Adaptive Filter p.330

Adaptive local noise reduction filter
Change the filter, depending on the contents of S_{xy}

- The local mean m_L and the local variance σ_L^2 can be used as compact descriptors of S_{xy}
- The ratio of the variances can in theory never be lager then 1 $\Rightarrow \quad \sigma_\eta^2 \leqslant \sigma_L^2$
- In practice, since the noise variance σ_η^2 needs to be estimated, this must be enforced.
  $\Rightarrow$ If $\sigma_\eta^2 > \sigma_L^2$, set ratio $\frac{\sigma_\eta^2}{\sigma_L^2}$ to 1

$$\hat{f}(x,y) = g(x,y) - \frac{\sigma_\eta^2}{\sigma_L^2}\left[g(x,y) - m_L\right] \tag{4.27}$$

Adaptive median filter p.332

4.3. Restoration in the presence of noise only - frequency domain filtering for periodic noise p.335

Noise appears as concentrated bursts of energy in the Fourier transform, at locations corresponding to the frequencies of the periodic interference.

4.3.1. Bandreject, Bandpass Filters p.337

see 3.11.1 Bandreject and Bandpass Filters, on page 15.

4.3.2. Notch Filter p.337

see 3.11.2 Notch Filters, on page 15.

4.3.3. Optimum Notch Filtering p.338

Danger: losing too much image information.
Concept: subtract a *weighted* portion of $\eta(x,y)$ from $g(x,y)$ to obtain an estimate of $f(x,y)$:

$$\hat{f}(x,y) = g(x,y) - w(x,y)\eta(x,y) \tag{4.28}$$

The weighting or modulation function $w(x,y)$ can be selected so that the resulting variance of $\hat{f}(x,y)$ is minimized over a specified neighborhood of every point (x,y). This results in

$$w(x,y) = \frac{\overline{g(x,y)\eta(x,y)} - \overline{g}(x,y)\overline{\eta}(x,y)}{\overline{\eta^2}(x,y) - \overline{\eta}^2(x,y)} \tag{4.29}$$

As $w(x,y)$ can be assumed to be constant in a neighborhood, it can be computed for *one* point in each non-overlapping neighborhood and used to process all points contained in that neighborhood.

4.4. Linear, Position-Invariant Degradations

$$g(x,y) = H[f(x,y)] + \eta(x,y) \quad \Leftrightarrow \quad G(u,v) = H(u,v)F(u,v) + N(u,v)$$

4.4.1. Estimating the degradation function p.346

By image observation

- Gather information from the image itself
- High contrast part of image is restored by hand and then degradtion function is estimated ⇒ labor intensive

By experminentaion

- Image acquisition system can be used to estimate the degradation function
  ⇒ Obtain the impulse response of the degradiation by imaging an impulse (small dot of light)

By modeling

- Preferred way, since general insights into degradation process can be found and it can be fully automated
- The Mathematical model comes for example from the physics of atmospheric turbulence.

$$H(u,v) = e^{-k(u^2+v^2)^{\frac{5}{6}}} \tag{4.30}$$

To create a **motion blurring**:

$$H(u,v) = \frac{T}{\pi(ua+vb)} \sin[\pi(ua+vb)] e^{-j\pi(ua+vb)} \tag{4.31}$$

$$\text{motion in x-dim. : } x_0(t) = \tfrac{a \cdot t}{T} \qquad \text{motion in y-dim. : } y_0(t) = \tfrac{b \cdot t}{T}$$

4.5. Inverse filtering p.351

- Clearly, even though we might know $H(u,v)$ perfectly, the noise makes it impossible to recover $f(x,y)$
- Furthermonre, if H(u,v) is close to zero, $\frac{N(u,v)}{H(u,v)}$ will dominate the result, rendering it useless.
- One trick is to focus on $H(u,v)$ near the origin, where it tends to be lager. ⇒ Use of lowpass

$$\hat{F}(x,y) = F(u,v) + \frac{N(u,v)}{H(u,v)} \tag{4.32}$$

4.6. Minimum Mean Square Error Filtering (Wiener) p.352

The goal is a minimum mean square error estimate. Image and noise are considered random variables. The goal is to deal with the degradation and

$$e^2 = E\{(f-\hat{f})^2\} \qquad \hat{f} \text{ is the estimate} \tag{4.33}$$

- noise and image are uncorrelated
- one of both is zero mean

$$\hat{F}(u,v) = \left[\frac{1}{H(u,v)}\frac{|H(u,v)|^2}{|H(u,v)|^2 + K}\right] G(u,v) \qquad \text{with } K = \frac{S_\eta(u,v)}{S_f(u,v)} \approx const \tag{4.34}$$

The parameter K is evalueted iteratevely, often using humans to judge the result. The function $H(u,v)$ is known. Wiener filter are optimal in the mean square error in the statistic sense, which means on average over an ensemble of image.

4.7. Constrained least squares filtering p.357

While the Wiener filter is powerful, the power spectral densities of the noise and the image need to be know, or a fixed ratio must be a good approximation. With the contrained least squares filter, only the noise mean and variance need to be known.
The goal is to find the smoothest image, witch satisfies the original equation, in some meaningful form.

$$\hat{F}(u,v) = \left[\frac{H^*(u,v)}{|H(u,v)|^2 + \gamma|P(u,v)|^2}\right] G(u,v) \tag{4.35}$$

- $P(u,v)$ is the Fourier transform of the Laplacian operator, which is $P(u,v) = -4\pi^2 D^2(u,v)$ according to Eq. 3.31 on page 14.
- γ can be iteratively adjusted by a human operator or automatically.

To automatically adjust the γ use the Eq. 4.38

$$||\mathbf{r}||^2 = \mathbf{r}^T\mathbf{r} \quad \text{with} \quad \mathbf{r} = \mathbf{g} - \mathbf{H}\hat{\mathbf{f}} \tag{4.36}$$

$$||\boldsymbol{\eta}||^2 = MN[\sigma_\eta^2 + m_\eta^2] \tag{4.37}$$

$$||\mathbf{r}||^2 = ||\boldsymbol{\eta}||^2 \pm a \tag{4.38}$$

1. Specify an inital value of γ
2. Compute $||\mathbf{r}||^2$ with Eq. 4.36, or Eq. 4.40
3. Stop if Eq. 4.38 is statisfied, otherwise return to step 2 after increasing γ if $||\mathbf{r}||^2 < ||\boldsymbol{\eta}||^2 - a$ or decreasing γ if $||\mathbf{r}||^2 > ||\boldsymbol{\eta}||^2 + a$. Use the new value of γ in Eq. 4.35 to recompute the optimum estimate $\hat{F}(u,v)$

Calculation of $||\mathbf{r}||^2$ in frequency domain

$$R(u,v) = G(u,v) - H(u,v)\hat{F}(u,v) \tag{4.39}$$

$$||\mathbf{r}||^2 = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} r^2(x,y) \quad \text{where } r(x,y) \text{ is inverse Transform of } R(u,v) \tag{4.40}$$

4.8. Geometric mean filter p.361

The form of the Wiener filter can be generalized to the so called geometric mean filter.

$$\hat{F}(u,v) = \left[\frac{H^*(u,v)}{|H(u,v)|^2}\right]^{\alpha} \left[\frac{H^*(u,v)}{|H(u,v)|^2 + \beta\left[\frac{S_\eta(u,v)}{S_f(u,v)}\right]}\right]^{1-\alpha} G(u,v) \tag{4.41}$$

- α and β are positive constants
- $\alpha = 1$, this is simply the inverse filter
- $\alpha = 0$ and $\beta = 1$, this is the Wiener filter
- $\alpha = 0$ and $0 < \beta < 1$, this is the parametric Wiener filter
- $\alpha = 0.5$ and $\beta = 1$, this is the geometic mean of the inverse filter and the Wiener filter, also called spectrum equalization filter

4.9. Image Reconstruction from Projections p.362

5. Color image processing p.394

5.1. Color fundamentals p.395

- Humans can perceive thousands of colors but only about 20-30 shades of gray
- Color is often a great feature of object detection and extraction
- White light can be split into a spectrum of colors
- The human visual system (HVS) is quite complex, but basically, the cones are responsible for the color vision
  - There are about 7 million cones which can be put into three categories
    - 65% are sensitive to red
    - 33% are sensitive to green
    - 2% are sensitive to blue
  - while their numbers vary widely, their sensitivity is quite similar
    - Therefore, the blue cones are the most sensitive
    - But the spatial resolution of red is much higher than that of blue
- Red, Green and Blue are called additive primary colors
- Magenta, Yellow and Cyan are called subtractive primary colors
- Humans often describe color using

  Brightness: Chromatic notation of intensity

  Hue: The dominant color

  Saturation: How much white light is mixed in with the pure color
- The very useful HSI (Hue, Saturation, Intensity) color space is based on the color perception of humans

The amount of red(X), green (Y) and blue (Z) needed to form a given color are called the tristimulus values. A color can then be specified using its trichromatic coefficients x,y and z, which are normalized so that they sum to 1. For any wavelength of light in the visible spectrum, these values can be read from tables that are the results from extensive experimentation.

$$x = \frac{X}{X+Y+Z} \qquad y = \frac{Y}{X+Y+Z} \qquad z = \frac{Z}{X+Y+Z}$$

5.2. Color models p.401

A color model is also called a color space or a color system. It allows colors to be represented in a well defined way, and depending on the task at hand, one model might be better suited than another.

5.2.1. RGB model p.402

Based on the three primary colors of the HVS and the most used model as most color camera systems and monitors have these three channels.

5.2.2. CMY and CMYK model p.406

These are models well suited for the primary colors of pigments, such as used in printing. The K stands for black, which, in theory can be mixed by summing up all subtractive primary colors, but for printing, the result tends to be a bit muddy, hence black is its own color.

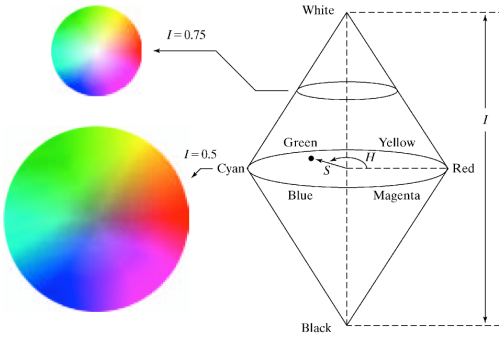
If all color values have been normalized, the conversion from an RGB to a CMY image is achieved by

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{5.1}$$

5.2.3. The HSI Color Model p.407

The HSI Color Model is well suited for image processing algorithms.

- **H** - hue: information about color, determined by an angle from some reference point → usually an angle of 0° at the red axis.
- **S** - saturation: length from the origin to the arbitrary color point
- **I** - intensity: given by the position of the plane on the vertical intensity axis.



5.2.4. Converting RGB ⇒ HSI p.410

The H component of each RGB pixel is obtained by

$$H = \begin{cases} \theta & \text{if } B \leqslant G \\ 360 - \theta & \text{if } B > G \end{cases} \tag{5.2}$$

with

$$\theta = \cos^{-1}\left[\frac{\frac{1}{2}\left[(R-G)+(R-B)\right]}{\left[(R-G)^2+(R-B)(G-B)\right]^{1/2}}\right] \tag{5.3}$$

The saturation component is given by

$$S = 1 - \frac{3}{(R+G+B)}\left[\min(R,G,B)\right] \tag{5.4}$$

The intensity component is given by

$$I = \frac{1}{3}(R+G+B) \tag{5.5}$$

5.2.5. Converting HSI ⇒ RGB p.411

Given values of HSI in the interval $[0,1]$, the applicable equations depend on the values of H, according to the 120° intervals separating the primaries.

First, H is multiplied by 360° so its range is $[0°, 360°]$.

	RG sector $(0^\circ \leqslant H < 120^\circ)$	**GB sector** $(120^\circ \leqslant H < 240^\circ)$	**BR sector** $(240^\circ \leqslant H < 360^\circ)$
H	$H = H$	$H = H - 120^\circ$	$H = H - 240^\circ$
R	$R = I\left[1 + \frac{S\cos(H)}{\cos(60^\circ - H)}\right]$	$R = I(1 - S)$	$R = 3I - (R + B)$
G	$G = 3I - (R + B)$	$G = I\left[1 + \frac{S\cos(H)}{\cos(60^\circ - H)}\right]$	$G = I(1 - S)$
B	$B = I(1 - S)$	$B = 3I - (R + B)$	$B = I\left[1 + \frac{S\cos(H)}{\cos(60^\circ - H)}\right]$

5.3. Pseudo color image processing p.414

- Assigning colors to gray values, based on a specified criterion.
- Goal: better human visualization and interpretation.
  Reason: humans can perceive thousands of color shades, but only 20-30 shades of gray.

5.3.1. Intensity Slicing p.415

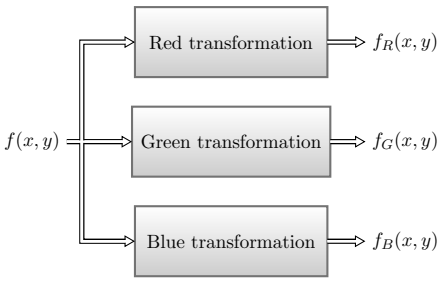
A gray-scale image with $[0, L-1]$ values is sliced into P planes at the intensity levels $l_1, l_2, \ldots, l_P$, creating the $P+1$ intervals $V_1, V_2, \ldots, V_{P+1}$. Intensity to color assignments are made according to

$$f(x,y) = c_k \qquad \text{if } f(x,y) \in V_k \tag{5.6}$$

where c_k is the color associated with the kth intensity interval V_k.

5.3.2. Intensity to color transformation p.418

A more general way than image slicing is to perform three independent transformations on the intensity. This method produces an image, whose color content is modulated by the transformation functions, which are only dependent on the intensity values.



5.4. Full-Color Image Processing p.424

In the RGB system, each point is represented by a vector

$$c = \begin{bmatrix} c_R \\ c_G \\ c_B \end{bmatrix} = \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad \text{thus an image is represented by} \quad c(x,y) = \begin{bmatrix} c_R(x,y) \\ c_G(x,y) \\ c_B(x,y) \end{bmatrix} = \begin{bmatrix} R(x,y) \\ G(x,y) \\ B(x,y) \end{bmatrix} \tag{5.7}$$

5.5. Color Transformations p.426

Color transformations describe processing of a color image within a *single* color model. Any transformation can be performed in any color model, howere some operations are better suited to specific models.

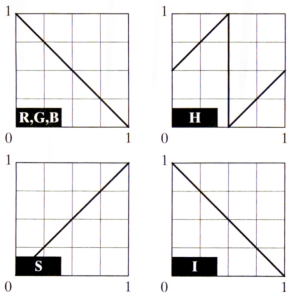
Generally a transformation is described by

$$g(x,y) = T\left[f(x,y)\right] \tag{5.8}$$

where T is an operator on f over a spatial neighborhood. Each pixel value is a triplet or quartet.

5.5.1. Color complements p.430

The hues directly opposite one another are called *complements*. They are analogous to the gray-scale negatives and are useful for enhancing details. The transformations can be achieved in the RGB or the HSI color spaces using the transfer functions shown on the right.



5.5.2. Color slicing p.431

Color slicing is used to highlight a specific range of colors, to separate objects from their surroundings. A simple way to achieve this is to map the n color-components (usually 3 or 4) outside a cube of width W, centered at $(a_1, a_2, \ldots, a_n)$, to a neutral color (e.g. gray $= (0.5, 0.5, 0.5)$ in RGB), by

$$s_i = \begin{cases} 0.5 & \text{if } \left[|r_j - a_j| > \frac{W}{2}\right]_{\text{any } 1 \leqslant j \leqslant n} \\ r_i & \text{otherwise} \end{cases} \qquad i = 1, 2, \ldots, n \tag{5.9}$$

Similarly, any region can be used. A sphere with radius R_0 and center $(a_0, a_1, \ldots, a_n)$ is given by

$$s_i = \begin{cases} 0.5 & \text{if } \sum_{j=1}^{n} (r_j - a_j)^2 > R_0^2 \\ r_i & \text{otherwise} \end{cases} \qquad i = 1, 2, \ldots, n \tag{5.10}$$

5.6. Tone and Color Corrections p.433

Used for photo enhancement and color reproduction. Most tone and color corrections are performed using a *device-independent color model*, such as the CIE $L^*a^*b^*$ model given by

$$L^* = 116 \cdot h(\frac{Y}{Y_W}) - 16 \tag{5.11}$$

$$a^* = 500 \left[h(\frac{X}{X_W}) - h(\frac{Y}{Y_W}) \right] \tag{5.12}$$

$$b^* = 200 \left[h(\frac{Y}{Y_W}) - h(\frac{Z}{Z_W}) \right] \tag{5.13}$$

where

$$h(q) = \begin{cases} \sqrt[3]{q} & q > 0.008856 \\ 7.787q + 16/116 & q \leqslant 0.008856 \end{cases} \tag{5.14}$$

and X represents red, Y represents green and Z represents blue in the XYZ color space. X_W,Y_W,Z_W are reference white values, typically the white of a perfectly reflecting diffuser under CIE standard D65 illumination.

L^* represents lightness, a^* red minus green and b^* green minus blue, making it useful in image manipulation and image compression.

5.6.1. Histogram Processing p.438

It is generally unwise to apply histogram equalization to the components of a color image independently. Usually, the color intensities are spread uniformly, leaving the colors unchanged. The HSI color space is ideally suited to this type of approach.

5.7. Smoothing and sharpening p.439

5.7.1. RGB Color Space

In the RGB color space, image smoothing can be carried out per-color-plane by

$$\overline{c}(x,y) = \begin{bmatrix} \frac{1}{K}\sum_{(s,t)\in S_{xy}} R(s,t) \\ \frac{1}{K}\sum_{(s,t)\in S_{xy}} G(s,t) \\ \frac{1}{K}\sum_{(s,t)\in S_{xy}} B(s,t) \end{bmatrix} \tag{5.15}$$

Similarly, image sharpening can be achieved using the Laplacian on each component of the color vector

$$\nabla^2\left[c(x,y)\right] = \begin{bmatrix} \nabla^2 R(x,y) \\ \nabla^2 G(x,y) \\ \nabla^2 B(x,y) \end{bmatrix} \tag{5.16}$$

5.8. HSI Color Space

The HSI color space decouples intensity and color information, so image smoothing can be achieved by simply smoothing the intensity component. The result will not be exactly the same as in the RGB color space, as only the intensity is changed but not hue and saturation.

Similarly, image sharpening can be achieved by applying the Laplacian operator on the intensity component and leaving hue and saturation unchanged.

5.9. Image segmentation based on color p.443

In the HSI space, image segmentation can be achieved by multiplying the hue with a binary mask, generated by thresholding the saturation image, resulting in a hue image. By selecting only the desired hues with a threshold, the image can be segmented.

Segmentation in the RGB space normally yields better results:
Let $\mathbf{a}$ be the average color, one tries to segment. $\mathbf{z}$ denotes an arbitrary point. If the distance between $\mathbf{a}$ and $\mathbf{z}$ is below a specific threshold D_0, the colors are *similar*.

$$D(\mathbf{z},\mathbf{a}) = ||\mathbf{z}-\mathbf{a}|| = \left[(\mathbf{z}-\mathbf{a})^T(\mathbf{z}-\mathbf{a})\right]^{1/2} = \left[(z_R-a_R)^2+(z_G-a_G)^2+(z_B-a_B)^2\right]^{1/2} \tag{5.17}$$

If $D(\mathbf{z},\mathbf{a}) \leqslant D_0$, the color is inside a sphere of radius D_0 around $\mathbf{a}$.

5.9.1. Bounding boxes p.446

As the computation of the above is rather expensive, even if the square roots aren't computed, a compromise is to use a bounding box centered on $\mathbf{a}$, with dimensions along each of the color axes chosen proportional to the standard deviation of the samples along each of the axis.

5.10. Color edge detection p.447

Two possibilities to detect edges in RGB-color space:

1. Use Sobel operators on each color to form the gradient of each RGB component and add them together at each coordinate (x,y)
2. Vector method described on p.449

5.11. Noise in color images p.451

Some of the gray scale methods are allowed for full color images.
Other schemes cannot be that easily extended to color images, such as the order statistic filters.

A. Appendix

A.1. Properties of the 2-D DFT

	Spatial Domain		Frequency Domain
1)	$f(x,y)$ real	$\Leftrightarrow$	$F^*(u,v) = F(-u,-v)$
2)	$f(x,y)$ imaginary	$\Leftrightarrow$	$F^*(-u,-v) = -F(u,v)$
3)	$f(x,y)$ real	$\Leftrightarrow$	$R(u,v)$ even; $I(u,v)$ odd
4)	$f(x,y)$ imaginary	$\Leftrightarrow$	($R(u,v)$ odd; $I(u,v)$ even
5)	$f(-x,-y)$ real	$\Leftrightarrow$	$F^*(u,v)$ complex
6)	$f(-x,-y)$ complex	$\Leftrightarrow$	$F(-u,-v)$ complex
7)	$f^*(x,y)$ complex	$\Leftrightarrow$	$F^*(-u,-v)$ complex
8)	$f(x,y)$ real and even	$\Leftrightarrow$	$F(u,v)$ real and even
9)	$f(x,y)$ real and odd	$\Leftrightarrow$	$F(u,v)$ imaginary and odd
10)	$f(x,y)$ imaginary and even	$\Leftrightarrow$	$F(u,v)$ imaginary and even
11)	$f(x,y)$ imaginary and odd	$\Leftrightarrow$	$F(u,v)$ real and odd
12)	$f(x,y)$ complex and even	$\Leftrightarrow$	$F(u,v)$ complex and even
13)	$f(x,y)$ complex and odd	$\Leftrightarrow$	$F(u,v)$ complex and odd

Table 1: Some symmetry properties of the 2-D DFT

	Name	Expression(s)
1)	Discrete Fourier transform (DFT) of $f(x,y)$	$F(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(ux/M+vy/N)}$
2)	Inverse discrete Fourier transform (IDFT) of $F(u,v)$	$f(x,y) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)e^{j2\pi(ux/M+vy/N)}$
3)	Polar representation	$F(u,v) = \lvert F(u,v)\rvert\, e^{j\phi(u,v)}$
4)	Spectrum	$\lvert F(u,v)\rvert = \left[R^2(u,v)+I^2(u,v)\right]^{1/2}$, $R=\text{Real}(F)$ $I=\text{Imag}(F)$
5)	Phase angle	$\phi(u,v) = \tan^{-1}\left[\frac{I(u,v)}{R(u,v)}\right]$
6)	Power spectrum	$P(u,v) = \lvert F(u,v)\rvert^2$
7)	Average value	$\bar{f}(x,y) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y) = \frac{1}{MN}F(0,0)$
8)	Periodicity (k_1 and k_2 are integers)	$F(u,v) = F(u+k_1M, v+k_2N)$ $f(x,y) = f(x+k_1M, y+k_2N)$
9)	Convolution	$f(x,y)\star h(x,y) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)h(x-m,y-n)$
10)	Correlation	$f(x,y)\,\☆\,h(x,y) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)h(x-m,y-n)$
11)	Separability	The 2-D DFT can be computed by computing 1-D DFT transforms along the rows (columns) of the image, followed by 1-D transforms along the columns(rows) of the result.
12)	Obtaining the inverse Fourier transform using forward transform	$MNf^*(x,y) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F^*(u,v)e^{-j2\pi(ux/M+vy/N)}$

Table 2: Summary of DFT definitions and corresponding expressions

	Name	**DFT Pairs**		
1)	Linearity	$af_1(x,y) + bf_2(x,y)$	$\Leftrightarrow$	$aF_1(u,v) + bF_2(u,v)$
2)	Translation (general)	$f(x,y)e^{j2\pi(u_0x/M+v_0y/n)}$	$\Leftrightarrow$	$F(u-u_0, v-v_0)$
		$f(x-x_0, y-y_0))$	$\Leftrightarrow$	$F(u,v)e^{-j2\pi(ux_0/M+vy_0/N)}$
3)	Translation to the center of the frequency rectangle, $(M/2, N/2)$	$f(x,y)(-1)^{x+y})$	$\Leftrightarrow$	$F(u-M/2, v-N/2)$
		$f(x-M/2, y-N/2))$	$\Leftrightarrow$	$F(u,v)(-1)^{u+v}$
4)	Rotation	$f(r, \theta+\theta_0))$	$\Leftrightarrow$	$F(\omega, \varphi+\theta_0)$
		$x = r\cos(\theta); \quad y = r\sin(\theta); \quad u = \omega\cos(\varphi); \quad v = \omega\sin(\varphi)$		
5)	Convolution theorem	$f(x,y) \star h(x,y))$	$\Leftrightarrow$	$F(u,v)H(u,v)$
		$f(x,y)h(x,y))$	$\Leftrightarrow$	$F(u,v) \star H(u,v)$
6)	Correlation theorem	$f(x,y)\stackrel{\star}{\star}h(x,y))$	$\Leftrightarrow$	$F^*(u,v)H(u,v)$
		$f^*(x,y)h(x,y))$	$\Leftrightarrow$	$F(u,v)\stackrel{\star}{\star}H(u,v)$
7)	Discrete unit impulse	$\delta(x,y))$	$\Leftrightarrow$	1
		1	$\Leftrightarrow$	$MN\delta(u,v)$
8)	Rectangle	$rect[a,b])$	$\Leftrightarrow$	$ab\frac{\sin(\pi ua)}{(\pi ua)}\frac{\sin(\pi vb)}{(\pi vb)}e^{-j\pi(ua+vb)}$
9)	Sine	$\sin(2\pi u_0x + 2\pi v_0y))$	$\Leftrightarrow$	
		$j\frac{1}{2}\left[\delta(u+Mu_0, v+Nv_0) - \delta(u-Mu_0, v_Nv_0)\right]$		
10)	Cosine	$\cos(2\pi u_0x + 2\pi v_0y))$	$\Leftrightarrow$	
		$\frac{1}{2}\left[\delta(u+Mu_0, v+Nv_0) + \delta(u-Mu_0, v_Nv_0)\right]$		
For continuous variables only:				
11)	*Differentiation*	$\left(\frac{\partial}{\partial t}\right)^m\left(\frac{\partial}{\partial z}\right)^n f(t,z))$	$\Leftrightarrow$	$(j2\pi\mu)^m(j2\pi\nu)^nF(\mu,\nu)$
11)	*Gaussian*	$A2\pi\sigma^2e^{-2\pi^2\sigma^2(t^2+z^2)})$	$\Leftrightarrow$	$Ae^{-(\mu^2\nu^2)/2\sigma^2}$

Table 3: Summary of DFT Pairs

A.2. Funktionswerte für Winkelargumente

deg	rad	sin	cos	tan
0°	0	0	1	0
30°	$\frac{\pi}{6}$	$\frac{1}{2}$	$\frac{\sqrt{3}}{2}$	$\frac{\sqrt{3}}{3}$
45°	$\frac{\pi}{4}$	$\frac{\sqrt{2}}{2}$	$\frac{\sqrt{2}}{2}$	1
60°	$\frac{\pi}{3}$	$\frac{\sqrt{3}}{2}$	$\frac{1}{2}$	$\sqrt{3}$

deg	rad	sin	cos
90°	$\frac{\pi}{2}$	1	0
120°	$\frac{2\pi}{3}$	$\frac{\sqrt{3}}{2}$	$-\frac{1}{2}$
135°	$\frac{3\pi}{4}$	$\frac{\sqrt{2}}{2}$	$-\frac{\sqrt{2}}{2}$
150°	$\frac{5\pi}{6}$	$\frac{1}{2}$	$-\frac{\sqrt{3}}{2}$

deg	rad	sin	cos
180°	π	0	-1
210°	$\frac{7\pi}{6}$	$-\frac{1}{2}$	$-\frac{\sqrt{3}}{2}$
225°	$\frac{5\pi}{4}$	$-\frac{\sqrt{2}}{2}$	$-\frac{\sqrt{2}}{2}$
240°	$\frac{4\pi}{3}$	$-\frac{\sqrt{3}}{2}$	$-\frac{1}{2}$

deg	rad	sin	cos
270°	$\frac{3\pi}{2}$	-1	0
300°	$\frac{5\pi}{3}$	$-\frac{\sqrt{3}}{2}$	$\frac{1}{2}$
315°	$\frac{7\pi}{4}$	$-\frac{\sqrt{2}}{2}$	$\frac{\sqrt{2}}{2}$
330°	$\frac{11\pi}{6}$	$-\frac{1}{2}$	$\frac{\sqrt{3}}{2}$

A.3. Additionstheoreme

$$\sin(a \pm b) = \sin(a) \cdot \cos(b) \pm \cos(a) \cdot \sin(b)$$
$$\cos(a \pm b) = \cos(a) \cdot \cos(b) \mp \sin(a) \cdot \sin(b)$$
$$\tan(a \pm b) = \frac{\tan(a) \pm \tan(b)}{1 \mp \tan(a) \cdot \tan(b)}$$

A.4. Produkte

$$\sin(a)\sin(b) = \tfrac{1}{2}(\cos(a-b) - \cos(a+b))$$
$$\cos(a)\cos(b) = \tfrac{1}{2}(\cos(a-b) + \cos(a+b))$$
$$\sin(a)\cos(b) = \tfrac{1}{2}(\sin(a-b) + \sin(a+b))$$

A.5. Doppel- und Halbwinkel

$$\sin(2a) = 2\sin(a)\cos(a)$$
$$\cos(2a) = \cos^2(a) - \sin^2(a) = 2\cos^2(a) - 1 = 1 - 2\sin^2(a)$$
$$\cos^2\left(\tfrac{a}{2}\right) = \tfrac{1+\cos(a)}{2} \qquad \sin^2\left(\frac{a}{2}\right) = \tfrac{1-\cos(a)}{2}$$

A.6. Summe und Differenz

$$\sin(a) + \sin(b) = 2 \cdot \sin\left(\tfrac{a+b}{2}\right) \cdot \cos\left(\tfrac{a-b}{2}\right)$$
$$\sin(a) - \sin(b) = 2 \cdot \sin\left(\tfrac{a-b}{2}\right) \cdot \cos\left(\tfrac{a+b}{2}\right)$$
$$\cos(a) + \cos(b) = 2 \cdot \cos\left(\tfrac{a+b}{2}\right) \cdot \cos\left(\tfrac{a-b}{2}\right)$$
$$\cos(a) - \cos(b) = -2 \cdot \sin\left(\tfrac{a+b}{2}\right) \cdot \sin\left(\tfrac{a-b}{2}\right)$$
$$\tan(a) \pm \tan(b) = \frac{\sin(a \pm b)}{\cos(a)\cos(b)}$$

A.7. Euler-Formeln

$$\cos(\alpha) = \frac{e^{j\alpha} + e^{-j\alpha}}{2} \qquad \sin(\alpha) = \frac{e^{j\alpha} - e^{-j\alpha}}{2j} \qquad e^{\pm jn\omega} = \cos n\omega \pm j \sin n\omega$$
$$e^{j\frac{\pi}{2}n} = j^n \qquad e^{-j\frac{\pi}{2}n} = (-j)^n \qquad e^{jn\pi} = e^{-jn\pi} = (-1)^n$$