Kapitel 4

Repræsentation og kodning

I dette kapitel beskrives først en række repræsentationer af billeder. Dernæst beskrives en række metoder til såvel reversibel som ikke-reversibel billedkodning. Visse af disse kodningsmetoder udnytter en bestemt repræsentation.

Sædvanligvis betragter og behandler vi et N $\times M$ gråtonebillede som NM 8-bits enheder. Denne repræsentation har sine åbenlyse fordele og er snævert knyttet til den sædvanlige billedoptagelsesmetode. En tæt knyttet repræsentation er at betragte billedet som bestående af 8 stk. N $\times$ M bitplaner. Radikalt anderledes er repræsentationen hvor billedet angives ved dets fourierspektrum. Som vi tidligere har set, giver denne form mulighed for let at kunne foretage visse filtreringer mv. Andre lignende repræsentationer er givet ved *cosinus-transformationen*; ved *Walsh-transformationen*; ved *Hardamard-transformationen*; og ved *Karhuen-Loeve-transformationen*. Disse transformationer med tilhørende repræsentationer er beskrevet i afsnit 4.2 og 4.3. I afsnit 4.4 beskrives nogle metoder hvorved et billede repræsenteres ved en pyramide af gradvist formindskede billeder. Denne repræsentation har store fordele ved såkaldte "grov-til-fin" analyser af billedet. Repræsentationer der udnytter information om kanterne hhv. segmenterne i et billede beskrives i afsnit 4.5. Endelig diskuteres i afsnit 4.6 hvorledes farvebilleder kan repræsenteres.

I afsnit 4.7 introduceres til billedkodning. Kodning er interessant ved lagring og transmition af billeder. Den klassiske metode til billedkodning starter med en transformation af et billede til en ny repræsentation. Et formål med transformationen er at elimnere den (statistiske) afhængighed, der ofte kan observeres mellem intensitetsværdier i pixels, hvis indbyrdes afstand er lille. Et andet formål er at "pakke" så meget information som muligt i så få transformationskoefficienter som muligt. I afsnit 4.8 gennemgås sådanne transformationer . Det næste trin er at kvantificere koefficienterne, dvs. at reducere værdiområdet for disse. Metoder til kvantificering diskuteres i afsnit 4.9. Sluttelig foretages en indkodning af de kvantificerede koefficienter. Metoder til indkodning diskuteres i afsnit 4.10. I afsnit 4.11 til 4.13 beskrives tre metoder, som ofte benyttes (og i den forstand er klassiske), men som ikke kan henføres under ovennævnte skema. Tre nyere metoder til billedkodning gennemgås i afsnittene 4.14 til 4.16. Disse metoder tager udgangspunkt i repræsentation af billedet hv. ved pyramider, kanter og segmenter. Kapitlet afsluttes i afsnit 4.17 med en disskusion af nogle muligheder for kodning af billedsekvenser.

4.1 Billedrepræsentation

Digitale gråtonebilleder kan repræsenteres på et utal af måder. Hver repræsentation kan have sine fordele og ulemper mht. en given behandling af billedet. I dette afsnit introduceres til de repræsentationer, der beskrives nærmere i afsnit 4.2. Generelt gælder at en repræsentation $T(u,v)$ for billedet $f(x,y)$ opnås ved *transformationen*:

$$T(u,v) = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)g(x,y,u,v) \tag{4.1}$$

hvor vi her antager at billedet er kvadratisk med kantlængde N. Vi skal udelukkende betragte *lineære* transformationer af ovenstående type. Funktionen $g(x,y,u,v)$ kaldes *transformationskernen*. Vi antager at der findes en *invers kerne* $h(u,v,x,y)$ således at $T(u,v)$ kan transformeres tilbage til det oprindelige billede $f(x,y)$:

$$f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} T(u,v)h(u,v,x,y) \tag{4.2}$$

Kernen $g(x,y,u,v)$ siges at være seperabel hvis:

$$g(x,y,u,v) = g_1(x,u)g_2(y,v) \tag{4.3}$$

og symmetrisk hvis $g_1 = g_2$ er identiske. Vi skal primært beskrive transformationer (med tilhørende repræsentationer), der er seperable og symmetriske. I dette tilfælde kan transformationen (4.1) opskrives som en simpel matriceoperation:

$$\mathbf{T} = \mathbf{A}'\mathbf{F}\mathbf{A} \tag{4.4}$$

hvor $\mathbf{F}$ hhv. $\mathbf{T}$ er $N \times N$ billedmatricer, og $\mathbf{A}$ er en $N \times N$ transformationsmatrice med elementer $a_{ij} = g_1(i,j)$. Svarende til den inverse kerne $h(u,v,x,y) = h_1(u,x)h_1(v,y)$ kan vi på tilsvarende måde danne matricen $\mathbf{B}$ med elementer $b_{ij} = h_1(i,j)$. Skrives:

$$\mathbf{B}'\mathbf{T}\mathbf{B} = \mathbf{B}'\mathbf{A}'\mathbf{F}\mathbf{A}\mathbf{B} \tag{4.5}$$

ses, at hvis $\mathbf{B} = \mathbf{A}^{-1}$ vil:

$$\mathbf{F} = \mathbf{B}'\mathbf{T}\mathbf{B} \tag{4.6}$$

Dette er således kravet til, at vi kan rekonstruere f fuldstændigt ud fra T. Repræsentationerne beskrevet i afsnit 4.2 opfylder alle dette krav.

En vigtig fortolkning af den inverse transformation (4.2) er en beskrivelse af billedet f som en sum af *projektioner* på en mængde af *basisbilleder* B for transformationen.

$$f = \sum_u\sum_v T(u,v)B(u,v) \tag{4.7}$$

Billedet f beskrives altså ved en linearkombination af billederne $B(u,v)$. Værdien $T(u,v)$ hørende til basisbilledet $B(u,v)$ kaldes i denne sammenhæng for koefficienten til basisbilledet. Koefficienterne findes vha. (4.1) ved at projicere billedet f ned på underrummet U hørende til basisbilledet. Vi kan derfor skrive:

$$F = U_1 \oplus U_2 \oplus \cdots \oplus U_{N^2} \tag{4.8}$$

hvor F udgør mængden af billeder f, og $\oplus$ angiver direkte sum. Hvis ovennævnte dekomposition af f er entydig, dvs. hvis det indre produkt $< a, b >$ mellem to elementer $a \in U_i$ og $b \in U_j$ er lig nul (for $i \neq j$) siges dekompositionen at være ortogonal . Vi er primært interesserede i sådanne dekompositioner.

Kodningsmæssigt er det af interesse at bestemme ortogonale transformationer, der pakker mest mulig information i de $k < N^2$ første underrum. I dette tilfælde kan vi approksimere f ved:

$$\hat{f} = \sum_{i=0}^{k} f_i \tag{4.9}$$

hvor $f_i \in U_i$. Ved rekonstruktion ud fra en sådan trunkering begås en fejl. Jo mindre denne fejl er, givet tallet k, for en bestemt transformation, jo bedre er denne mht. kodning af billedet. Som vi skal se i afsnit 4.3 findes netop én transformation, der pakker informationen i et billede mest mulig ned i de første koefficienter.

En vigtig observation, der ofte glemmes, er at værdiområdet for koefficienterne sædvanligvis er meget større end gråtone værdiområdet. Dette har ikke kun kodningsmæssig interesse. Eksempelvis er værdiområdet for fouriertransformationen de komplekse tal. Da billedet er reelt, og koefficienterne derfor symmetriske, repræsenteres den fouriertransformerede af et $N \times N$ billede således ved N^2 *reelle* koefficienter. For at undgå numeriske problemer, ved implementation på en given datamat, er det vigtigt at kende nøjagtigheden af repræsentationen af flydende tal på maskinen. I modsat fald kan man let fejlfortolke et overraskende resultat. Kodningsmæssigt vil transformationer som fouriertransformationen resultere i en forøgelse af det antal bit, der skal til at repræsentere billedet. Det er altså ikke i denne fase, at der sker nogen datakompression.

4.2 Klassiske repræsentationer

I dette afsnit beskrives fire transformationer, der ofte refereres i litteraturen. Især fouriertransformationen og cosinustransformationen er meget benyttede i mange billedbehandlingsopgaver. Transformationerne beskrives kun kort, uden diskussion af algoritmer til hurtig implementation af transformationerne. Algoritmebeskrivelser kan eksempelvis findes i *Rosenfeld and Kak: Digital Picture Processing, Academic Press, 1982* og i *Gonzales and Wintz: Digital Image Processing, Addison-Wesley, 1987*. For alle transformationer gælder at de kan beskrives ved (4.1), at de er seperable og symmetriske, og derfor kan beskrives ved (4.4). Vi skal derfor kun angive de en-dimensionale transformationer. I afsnit 4.8 beskrives nogle anvendelser mht. kodning.

4.2.1 Repræsentation ved fourier-transformation

Fouriertransformationen antages velkendt, og beskrives derfor kun summarisk. I det endimensionelle tilfælde er kernen $g(x,u)$ givet ved:

$$g(x,u) = \frac{1}{\sqrt{N}} e^{-2\pi i u x/N} \tag{4.10}$$

hvor vi betrager et signal med N stikprøver, og hvor $i = \sqrt{-1}$. Den inverse kerne er bestemt ved:

$$h(u,x) = e^{2\pi i u x/N} \tag{4.11}$$

Det kan vises at fouriertransformationen er suboptimal i den forstand, at koefficienterne ikke er ukorrelerede. Dog vil graden af korrelation aftage når billedstørrelsen øges. Årsagen til transformationens popularitet er primært at repræsentationen tillader en direkte fortolkning mht. frekvensopdeling. En sekundær årsag er at transformationen kan foretages relativ let og hurtigt vha. FFT-algoritmen. Kodningsmæssigt er transformationen underlegen i forhold til både cosinus-transformationen og Carhuen-Loeve-transformationen. En ulempe ved fouriertransformationen er antagelsen om at billedet er periodisk udvidet. Hvis intensiteterne i modsatte billedrande er meget forskellige vil dette introducere høje frekvenser. Disse er altså forårsaget af kravet om periodisk udvidelse og er ikke relateret til diskontinuiteter i billedet selv. Dette fænomen giver ofte grund til problemer ved analyser i denne repræsentation. Dæmpes de høje frekvenser og invers fouriertransformeres det filtrerede billede vil dette være tydeligt forvrænget. Problemet ses oftest ved små billeder, eller ved blokkodning af billedet. Vi skal diskutere problemet videre i afsnit 4.8.

4.2.2 Repræsentation ved cosinus-transformation

Kernen for den diskrete cosinustransformation af et signal med N stikprøver, er givet ved:

$$\begin{aligned} g(x,0) &= \frac{1}{\sqrt{N}} \\ g(x,u) &= \sqrt{\frac{2}{N}} \cos\left(\frac{(2x+1)u\pi}{2N}\right) \end{aligned} \tag{4.12}$$

for $x \in [0, N-1]$ og $u \in [1, N-1]$. Den inverse transformation er bestemt ved dekompositionen:

$$f(x) = \frac{1}{\sqrt{N}} C(0) + \sqrt{\frac{2}{N}} \sum_{u=1}^{N-1} C(u) \cos\left(\frac{(2x+1)u\pi}{2N}\right) \tag{4.13}$$

hvor $C(u)$ er transformationskoefficienterne til f.

Cosinustransformationen er meget benyttet ved kodning af billeder. En årsag er, som vi senere skal diskutere, at koefficienterne for transformationen er næsten dekorrelerede.

Kodningsmæssigt er transformationen derfor approksimativt så god som Carhuen-Loeve-transformationen. Transformationen kan desuden implementeres hurtigt ved en algoritme, der meget ligner FFT-algoritmen. I forhold til fouriertransformationen antager cosinustransformationen ikke en periodisk udvidelse af billedet. I stedet er der tale om udvidelse ved spejling. Dette betyder at der kun sjældent opstår fejlfortolkninger af "frekvensinformationen". Dette gør metoden mere attraktiv ved blokkodning end fouriertransformationen. For nyligt er metoden blevet valgt som international standard ved kompression af billeder, og der er allerede dedikeret "hardware" på markedet til hurtig kompression/dekompression af billeder.

4.2.3 Repræsentation ved Walsh-transformation

Den endimensionale diskrete Walsh transformation er bestemt ved kernen:

$$g(x,u) \;=\; \frac{1}{N}\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)} \tag{4.14}$$

hvor billedstørrelsen N antages lig 2^n, og hvor $b_k(z)$ er defineret ved den k'te bit i den binære repræsentation af z. I modsætning til de tidligere omtalte basisfunktioner, f.eks. (4.10), består værdierne af Walsh transformationskernen (bortset fra konstanten $\frac{1}{N}$) af tallene 1 og -1.

x \ u	0	1	2	3	4	5	6	7
0	+	+	+	+	+	+	+	+
1	+	+	+	+	-	-	-	-
2	+	+	-	-	+	+	-	-
3	+	+	-	-	-	-	+	+
4	+	-	+	-	+	-	+	-
5	+	-	+	-	-	+	-	+
6	+	-	-	+	+	-	-	+
7	+	-	-	+	-	+	+	-

Tabel 4.1: Walsh transformationskernen for $N = 8$

Det ses at matricen for kerneværdierne er symmetrisk og at rækker og kollonner er indbyrdes ortogonale. Bortset fra konstanten $\frac{1}{N}$ er den inverse kerne identisk med $g(x,u)$. I det to-dimensionale tilfælde bliver de to kerner fuldstændig ens:

$$g(x,y,u,v) \;=\; h(x,y,u,v) \;=\; \frac{1}{N}\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)+b_i(y)b_{n-1-i}(v)} \tag{4.15}$$

Det kan let vises at den todimensionale Walsh transformationskerne er både seperabel og symmetrisk. Walsh transformationen kan derfor beregnes ved en algoritme, der meget ligner FFT-algoritmen. Yderligere beskrivelse af en implementation af transformationen kan findes i f.eks. *Gonzales and Wintz: Digital Image Processing, Addison-Wesley, 1987.*

4.2.4 Repræsentation ved Hardamard-transformation

Hardamard transformationskernen er i det en-dimensionale tilfælde givet ved:

$$g(x,u) \;=\; \frac{1}{N}(-1)^{\sum_{i=0}^{n-1} b_i(x)b_i(u)} \tag{4.16}$$

hvor $b_k(z)$ er den k'te bit i den binære repræsentation af z, hvor $N = 2^n$, og hvor summationen i eksponenten antages foretaget i modulo-2 aritmetik. Som for Walsh transformationen bestemmer Hardamard kernen en matrice hvor rækker og søjler er ortogonale.

x \ u	0	1	2	3	4	5	6	7
0	+	+	+	+	+	+	+	+
1	+	-	+	-	+	-	+	-
2	+	+	-	-	+	+	-	-
3	+	-	-	+	+	-	-	+
4	+	+	+	+	-	-	-	-
5	+	-	+	-	-	+	-	+
6	+	+	-	-	-	-	+	+
7	+	-	-	+	-	+	+	-

Tabel 4.2: Hardamard transformationskernen for $N = 8$

Bortset fra konstanten $\frac{1}{N}$ er den endimensionale inverse Hardamard transformation identisk med $g(x,u)$. I det to dimensionale tilfælse bliver de to transformationer helt ens:

$$g(x,y,u,v) \;=\; h(x,y,u,v) \;=\; \frac{1}{N}(-1)^{\sum_{i=0}^{n-1}[b_i(x)b_i(u)+b_i(y)b_i(v)]} \tag{4.17}$$

Som for Walsh transformationen kan det let vises at den todimensionale Hardamard kerne er seperabel og symmetrisk. De to transformationer minder i høj grad om hinaden. Der er dog forskelle, se f.eks. *Gonzales and Wintz: Digital Image Processing, Addison-Wesley, 1987.* Ingen af metoderne benyttes (i nævneværdigt omfang) i dag. Tidligere har metoderne været benyttet ved billedkodning. Metoderne er, som det fremgår, beregningsmæssigt meget hurtige, men har forholdsvis dårlige egenskaber mht. dekorrelation af transformationskoefficienterne.

4.3 Karhuen-Loeve -repræsentationen

Karhuen-Loeve transformationen, der også går under navnene Hotelling transformationen og Principal komponent transformationen, adskiller sig fra transformationerne i afsnit 4.2 ved at være baseret på statistiske egenskaber ved det givne forelagte billede $f(x,y)$. Basisbillederne for transformationen er således afhængige af f. Dette, at basisbillederne ikke er underforståede, men bestemt ud fra f selv gør metoden overlegen (mht. dekorrelation af intensitetsværdierne) i forhold til de tidligere beskrevne metoder. Generelt bliver informationen i et billede pakket bedre ned i færre koefficienter. Dette er, kodningsmæssigt, dog af tvivlsom værdi, fordi rekonstruktionen ikke kan foretages uden yderligere kendskab til basisbillederne. Der findes imidlertid andre anvendelser, hvor dette problem ikke er relevant.

Vi betragter billedelementerne $f(x,y)$ som stokastiske variable og opskriver disse (ved passende ordning) som en $N^2 \times 1$ søjlevektor:

$$\mathbf{x} = \begin{pmatrix} f(0,0) \\ f(1,0) \\ \vdots \\ f(N-1,N-1) \end{pmatrix} \tag{4.18}$$

Kovariansmatricen $\mathbf{C_x}$ for $\mathbf{x}$ har størrelse $N^2 \times N^2$ og defineres ved:

$$\mathbf{C_x} = E\{(\mathbf{x}-\mathbf{m_x})(\mathbf{x}-\mathbf{m_x})^t\} = E\{\mathbf{x}\mathbf{x}^t\} - \mathbf{m_x}\mathbf{m_x}^t \tag{4.19}$$

hvor t angiver transponering, $E\{\}$ angiver forventet værdi, og:

$$\mathbf{m_x} = E\{\mathbf{x}\} \tag{4.20}$$

Vektoren $\mathbf{m_x}$ har således længden N^2. Lad nu $\mathbf{e_i}$ hhv. λ_i være den i'te egenvektor hhv. egenværdi for $\mathbf{C_x}$ (i = 1,2,...,N^2), og lad $\mathbf{A}$ være en transformationsmatrice hvis rækker er bestemt ved egenvektorerne $\mathbf{e_i}$.

$$\mathbf{A} = \begin{pmatrix} \mathbf{e_{11}} & \mathbf{e_{12}} & \cdots & \mathbf{e_{1N^2}} \\ \mathbf{e_{21}} & \mathbf{e_{22}} & \cdots & \mathbf{e_{2N^2}} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{e_{N^21}} & \mathbf{e_{N^22}} & \cdots & \mathbf{e_{N^2N^2}} \end{pmatrix} = \begin{pmatrix} \vec{\mathbf{e_1}} \\ \vec{\mathbf{e_2}} \\ \vdots \\ \vec{\mathbf{e_{N^2}}} \end{pmatrix} \tag{4.21}$$

således at $\mathbf{e_{ij}}$ er den j'te komponent af den i'te egenvektor. Karhuen-Loeve transformationen er da givet ved:

$$\mathbf{y} = \mathbf{A}(\mathbf{x}-\mathbf{m_x}) \tag{4.22}$$

Denne transformation har to interessante egenskaber, nemlig:

$$\mathbf{m_y} = E\{\mathbf{y}\} = E\{\mathbf{A}(\mathbf{x}-\mathbf{m_x})\} = \mathbf{A}E\{\mathbf{x}\} - \mathbf{A}\mathbf{m_x} = 0 \tag{4.23}$$

og

$$\begin{aligned}
\mathbf{C_y} &= E\{(\mathbf{y}-\mathbf{m_y})(\mathbf{y}-\mathbf{m_y})'\} \qquad (4.24)\\
&= E\{\mathbf{y}\mathbf{y}'\}\\
&= E\{(\mathbf{Ax}-\mathbf{Am_x})(\mathbf{Ax}-\mathbf{Am_x})^t\}\\
&= E\{\mathbf{A}(\mathbf{x}-\mathbf{m_x})(\mathbf{x}-\mathbf{m_x})^t\mathbf{A}^t\}\\
&= \mathbf{AC_xA}^t\\
&= diag(\lambda_1, \lambda_2, \ldots, \lambda_{N^2})
\end{aligned}$$

Middelværdien $\mathbf{m_y}$ af transformationskoefficienterne er altså lig nulvektoren. Fra lineær algebra genkender vi udtrykket for $\mathbf{C_y}$ som diagonaliseringen af $\mathbf{C_x}$. Dette betyder at $\mathbf{C_y}$ har formen:

$$\mathbf{C_y} = \begin{pmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_{N^2} \end{pmatrix} \qquad (4.25)$$

At kovariansmatricen for koefficienterne $\mathbf{y}$ til transformationen er en diagonalmatrice betyder at koefficienterne er (fuldstædig) dekorreleret. Størrelsen af egenværdien λ_i angiver hvor stor en del af energien i det oprindelige billede, der er pakket i basisbilledet svarende til den i'te egenvektor. Sædvanligvis ordnes basisbillederne således at $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_{N^2}$. Det kan yderligere bemærkes at λ_i er lig variansen af det i'te elementet af $\mathbf{y}$.

Da $\mathbf{C_x}$ er reel og symmetrisk vil $\mathbf{A}$ altid være en ortonormal matrice. Heraf følger at $\mathbf{A}^{-1} = \mathbf{A}^t$. Den inverse transformation er derfor givet ved:

$$\mathbf{x} = \mathbf{A}^t\mathbf{y} + \mathbf{m_x} \qquad (4.26)$$

Bemærk iøvrigt at transformationsmatricen $\mathbf{A}$ *ikke* er seperabel.

Antag at vi ønsker en rekonstruktion $\hat{\mathbf{x}}$ på basis af de k første egenvektorer (basisbilleder) af $\mathbf{C_x}$. Vi har da.

$$\hat{\mathbf{x}} = \mathbf{A_k}^t\mathbf{y} + \mathbf{m_x} \qquad (4.27)$$

Ved udregning af kvadratfejlen er det praktisk at udtrykke denne som sporet (Tr) af det ydre produkt af differensen $(\mathbf{x}-\hat{\mathbf{x}})$ i stedet for, som normalt, det indre produkt. Vi kan opskrive kvadratfejlen som:

$$\begin{aligned}
e_k^2 &= ||\mathbf{x}-\hat{\mathbf{x}}||^2 \qquad (4.28)\\
&= ||(\mathbf{A^t}-\mathbf{A_k^t})\mathbf{y}||^2\\
&= Tr[E\{\left[(\mathbf{A^t}-\mathbf{A_k^t})\mathbf{y}\right]\left[(\mathbf{A^t}-\mathbf{A_k^t})\mathbf{y}\right]^t\}]\\
&= Tr[E\{(\mathbf{A^t}-\mathbf{A_k^t})\mathbf{y}\mathbf{y^t}(\mathbf{A}-\mathbf{A_k})\}]\\
&= Tr[(\mathbf{A^t}-\mathbf{A_k^t})\mathbf{AC_xA^t}(\mathbf{A}-\mathbf{A_k})]\\
&= Tr[(\mathbf{A}-\mathbf{A_k})^t\mathbf{C_y}(\mathbf{A}-\mathbf{A_k})]\\
&= \sum_{j=k+1}^{N^2}\lambda_j
\end{aligned}$$

Da egenværdierne λ_i antages sorteret efter aftagende værdi, ses at transformationen optimalt pakker mest mulig energi i koefficienterne svarende til de første basisbilleder.

Mht. kodning diskuteres metoden videre i afsnit 4.8. Metoden har været meget benyttet i analyser af multispektrale billeder. Disse optages typisk fra satellitter eller fra flybårne skannere. Typisk optages der 3-10 billeder svarende til hvert sit spektrale bånd. For nyligt er der udviklet skannere med over 200 spektrale kanaler. Da de spektrale bånd overlapper hinanden en del og da visse materialer reflekterer lys over et bredt spektrum af frekvenser er informationsindholdet i billederne meget redundant. Det er således muligt at ekstrahere det meste af informationen (og eliminere støjen) i sættet af billeder ved at foretage en principalkomponent analyse på sættet af de N billeder. Ved at ignorere (nulle ud) de mindste af de N egenværdier og tilbagetransformere fås et nyt (mindre) sæt af billeder. Informationsindholdet i disse vil være "ortogonalt" og vil beskrive det oprindelige sæt bedst muligt. Ofte kan videre analyser indskrænkes ved kun at betragte de første 2-3 billeder i det transformerede sæt.

Beregning af $\mathbf{A}$-matricen for (eksempelvis) et 512×512 billede involverer bestemmelse af egenvektorer og egenværdier for en $512^2 \times 512^2$ matrice. Denne opgave vil være særdeles beregningsmæssig intensiv. En hurtig algoritme til Karhuen-Loeve transformation er beskrevet i: *Jain: A Fast Carhuen-Loeve Transform for a Class of Random Processes; IEEE trans. on communication vol. 24.*

4.4 Pyramiderepræsentationer

Den grundlæggende ide i pyramiderepræsentationer er, at repræsentere billedet ved et sæt af trinvist "nedfotograferede" billeder. Repræsentationen omfatter således det oprindelige billede selv. Repræsentationen medfører derfor et øget lagerforbrug. Fordelen ved repræsentationen er, at den letter "grov-til-fin" analyser af billedet. Således muliggøres effektive globale analyser implementeret udelukkende ved brug af lokale operationer. Pyramiderepræsentationer har især været benyttet ved visse segmenteringsmetoder, ved billedkodning, samt ved visse datamatsynsmetoder som f.eks. stereo- og bevægelsesanalyser.

For nyligt har en repræsentation kaldet *wavelet repræsentationen* påkaldt sig stor interesse. En af fordelene ved denne pyramiderepræsentation er, at den beskriver en ortogonal dekomposition af billedet. Repræsentationen fylder, bortset fra udvidelsen af værdiområdet til de relle tal, ikke mere end det oprindelige billede. Dette er i kontrast til de tidligere nævnte pyramiderepræsentationer.

De fleste pyramiderepræsentationer fremkommer ved sampling af et sæt af lav-pass filtrerede versioner af det oprindelige billede. Ved lav-pass-filtreringen reduceres båndbredden af det filtrerede billede. Det er derfor muligt at repræsentere dette ved færre pixels. Antag at det oprindelige billede har kantlængder N og M, og antag et det lav-pass filtrerede samplede billede har kantlængder $\frac{N}{k} \times \frac{M}{k}$. Da siges 2-niveau pyramiden bestående af det oprindelige og det samplede billede at have reduktionsfaktor k. Pyramider med kun 2 niveauer er sjældne. Ofte benyttes både 3, 4 eller 5 niveauer. Dette er naturligvis afhængig af såvel reduktionsfaktoren, som af det generelle formål med analysen. Det er ikke usædvanligt at billeder er kvadratiske med en kantlængde der er en potens af 2. I dette tilfælde er det nærliggende at benytte en pyramidestruktur med reduktionsfaktor 2. En sådan pyramide kaldes en $log2$-pyramide. Generelt tales om $logk$-pyramider. Hvis det mindste billede i en pyramidestruktur har kantlængde 1 tales om en *fuld pyramide*. I modsat fald tales om en *trunkeret pyramide*, eller om en pyramidestub. Bemærk endvidere at for en fuld N×M $log2$ -pyramide er det totale pladsforbrug givet ved:

$$N \times M[1 + \frac{1}{2^2} + \frac{1}{2^4} + \frac{1}{2^8} + ...] < \frac{4}{3}\, N \times M \tag{4.29}$$

Hvert billede i en pyramidestruktur siges at have et niveau. Dette bestemmes ved at det oprindelige billede har niveau 0, det næst største billede niveau 1 etc. Billedværdierne på de enkelte niveauer i en pyramide kan bestemmes direkte ud fra det oprindelige billede eller alternativt ud fra billedet på nærmeste mindre niveau. I det følgende vil vi kun betragte pyramider, der opbygges på sidstnævnte måde. Her vil hvert billedelement på niveau $z > 0$ have en værdi, der er bestemt ved en frembringerfunktion f på billedet på niveau z-1. Typisk har frembringerfunktionen begrænset støtte og er positionsinvariant. Frembringerfunktionen for pyramidestrukturer for binære billeder er som regel ikke lineær. Hvis frembringerfunktionen er lineær, kan billedværdierne på niveau z beskrives som fremkommet ved sampling af foldningen af punktspredningsfunktionen svarende til f med billedet på niveau z-1. I disse tilfælde er egenskaberne ved pyramidestrukturen primært bestemt ved f og k. I afsnit 4.4.2 beskrives en meget anvendt pyramidestruktur af denne type.

Grov til fin -analyse

En simpel men fundamental anvendelse (kaldet grov-til-fin analyse) af pyramidestrukturer skal kort skitseres. Formålet med en grov-til-fin analyse er at foretage globale billedoperationer v.h.a. lokale operationer. Ideen er at foretage analysen gentagende gange, først på det mindste billede i strukturen, dernæst på det næstmindste osv. indtil analysen til sidst foretages på det oprindelige billede. Billedet på det øverste niveau beskriver det oprindelige billede globalt, og er (grundet lavpass-filtreringen) reelt støjfrit. Da billedet er lille kan en robust og (forholdsvis) global analyse foretages ved lokale operationer. Hvis analysens mål er angivelse af positioner (f.eks kanter eller segmenter) vil lokaliseringen af disse til gengæld være ringe. Et approksimativt resultat er imidlertid ofte tilstrækkeligt til at dirigere/vejlede analysen på det billede på det næstøverste niveau i pyramidestrukturen. Da lavpass-filtreringen på dette niveau er mindre udtalt, vil eventuelle positionsangivelser kunne bestemmes med større nøjagtighed. Ved gentagende anvendelse af resultatet fra et niveau til vejledning af analysen på det næste niveau kan analysen til sidst foretages på det oprindelige billede.

En fordel ved metoden er, at information kan spredes over store afstande selv om der kun anvendes lokale operationer. Et eksempel på dette er en adaptiv grænseværdiafskæring. Antage at en procedure til bestemmelse af grænseværdien på basis af intensitetsværdierne i et vindue er givet. Vi ønsker at bestemme et grænseværdi-billede, der for hver pixel angiver en værdi, der ikke kun er afhængig af intensitetsværdierne i en lille omegn, men også afhænger af mere globale målinger. Anvendes proceduren på øverste pyramideniveau, opnås en global information, der kan spredes ved projektion på niveauerne på mindre niveau. Proceduren kan modificeres således at den bestemmer grænseværdien (på et givet niveau) dels ud fra lokal information om intensiteten i det aktuelle vindue på niveauet, dels ud fra den mere globale grænseværdi bestemt på niveauet een større.

En anden fordel er at analysen kan foretages robust uden at præcisionen midskes. Dette er f.eks tilfældet med metoden til kantdetektion (kantfokusering), der er beskrevet i afsnit 2.9.

En tredie anvendelse er at analysen kun behøver at blive foretaget i de områder af billedet, som den grove analyse har markeret som interessante (eng. focus of attention) . Herved kan analysehastigheden øges. Dette kan f.eks udnyttes i metoder til detektion af ændringer i en tidssekvens af billeder (overvågning). Hvis noget i scenen ændrer sig markant, vil intensiteten på øverste pyramideniveau ændrer sig. Positionen af intensitetsændringen angiver hvor i billederne på lavere niveau, at en nærmere analyse skal foretages.

4.4.1 Simple pyramidestrukturer

De simpleste (og mest benyttede) pyramidestrukturer er $log2$-pyramider hvor støtten for frembringerfunktionen er 2×2 pixels. For binære billeder benyttes disse pyramidetyper næsten udelukkende. Her er frembringerfunktionen typisk et logisk udtryk i de 4 støtteelementer, f.eks logisk *and* eller *or*. Alternativt kan frembringerfunktionen give værdi 1 hvis mindst to af støtteelementerne har værdi 1, ellers 0. For gråtonebilleder vil en oplagt frembringerfunktion være middelværdien af støtteelementernes værdi. I sidstnævnte eksempel kan frembringerfunktionens værdi på niveau z beskrives ved foldning af billedet på niveau z-1 med et 2×2 box-filter efterfulgt af en sampling af hvert fjerde billedelement. De egenskaber vi tidligere har diskuteret ved box-filtret kunne gentages her.

Quad-tree repræsentation

Quad-tree repræsentationer af billeder benyttes fortrinsvis ved segmenteringsmetoder af "split"- og "split-and-merge" typen. Metoden kan måske også være velegnet til billedkodning. Den grundlæggende ide er kun at repræsentere dele af pyramiden. Hvert billedelement i pyramiden kan antage 2 former. Enten 4 "pointere" til billedelementerne i støtten for elementet, eller en binær/gråtoneværdi. Opbygning af pyramiden kan lettest beskrives ved et eksempel. Her betragtes en split-segmentering.

Ved en "split" segmentering benyttes et prædikat, der ud fra et billedområde angiver om dette bør opdeles i mindre områder eller ej. Opbygningen foretages fra toppen af en fuld $log2$-pyramide. Hvis prædikatet ikke er opfyldt for hele billedet (det aktuelle område), repræsenteres dette ved et enkelt billedelement hvis værdi bestemmes ved en funktion (en slags frembringerfunktion). Hvis prædikatet er opfyldt genereres fire nye billeelementer (svarende til støtteelementerne), og proceduren gentages rekursivt for hvert element. Et billede med konstant gråtoneværdi vil altså blive repræsenteret ved eet billedelement.

Quad-tree pyramider kan også opbygges fra bladene mod roden således at reduktionen af fire støtteelementer foretages hvis prædikatet (der her altid er funktion af 4 værdier) er falsk. Denne opbygning vil ikke nødvendigvis give samme pyramide som ovenfor beskrevne metode. Vi skal ikke her diskutere Quad-træer yderligere. En glimrende oversigt over denne repræsentationsform findes i: *H. Samet, The Quadtree and Related Hierarchial Data Structures, Computing Surveys vol. 16 no. 2 june 1984, pp 187-219.*

4.4.2 Gauss-pyramide og laplace-pyramide

I en Gauss-pyramide er frembringerfunktionen en gaussfunktion. Gauss-pyramider er nok de mest benyttede pyramiderepræsentationer. Disse er oftest $log2$ pyramider, men $log\sqrt{2}$ pyramider ses også. Gaussfunktionen fungerer her som et lav-pass filter. Selv om fuktionen aldrig når nul, aftager den så hurtigt at det i praksis ofte antages at denne kan trunkeres ved $\pm 2\sigma$, hvor σ er spredningen af gaussfunktionen. Den fouriertransformerede af en gaussfunktion er selv en gaussfunktion, med spredning $\frac{1}{2\pi\sigma}$. Båndbredden af gaussfunktionen antages derfor at være $\frac{1}{\pi\sigma}$. Dette betyder at omkring 96% af frekvensområdet bevares ved trunkeringen. Ved foldningen af gaussfunktionen med et billede vil båndbredden af det foldede billede være minimum af båndbredderne for det oprindelige billede hhv. gaussfunktionen. Idet båndbredden af det oprindelige billede kan være stor, ser vi at båndbredden af det foldede billede vil være $\frac{1}{\pi\sigma}$. Jvf. *samplingsætningen* betyder dette at det foldede billede kan samples uden væsenligt tab af information, hvis blot samplingafstanden k opfylder at:

$$k \leq \frac{1}{2w} = \frac{\pi\sigma}{2} \tag{4.30}$$

ønsker vi at opbygge en gausspyramide med reduktionsfaktor k, sådan at informationstabet sker under foldningen og ikke ved samplingen, skal vi derfor blot benytte en gaussfunktion hvis spredning er givet ved:

$$\sigma \geq \frac{2k}{\pi} \tag{4.31}$$

For en $log2$-pyramide betyder dette at $\sigma >$ ca. 1.27. Ofte anvendes værdien $\sigma = \sqrt{2}$. Gausspyramider er attraktive dels fordi de bevarer strukturen i billedet bedre (under nedfotograferingen) end pyramider med mere simple frembringerfunktioner, dels fordi de er relativt hurtige at generere. En hurtigere implementation er beskrevet i *P. J. Burt and E. H. Adelson: The Laplacian Pyramid as a Compact Image Code; IEEE trans. on communications vol. 31 no. 4 1983.* Her benyttes en såkaldt “Burt-pyramide”, der fremkommer ved valg af en separabel symmetrisk frembringerfunktion $g(x, y)$ med en støtte på 5×5 pixels.

$$g(x, y) = w_a(x)w_a(y) \tag{4.32}$$

Filtret w er parametriseret ved værdien a, således at $w(0) = a$, $w(1) = 1/4$ og $w(2) = 1/4 - a/2$. For $a \approx 0.4$ minder filterformen meget om en gaussfunktion. For $a = 0.5$ har filtret en trekantsform. Beregningsmæssigt er approksimationen (oplagt) meget effektiv.

En anden vigtig pyramidetype er laplacepyramiden. Denne kan frembringes ud fra en gausspyramide ved niveauvis anvendelse af laplaceoperatoren på billederne. Denne metode benyttes dog sjældent i praksis. I stedet approksimeres laplaceoperatoren (eller mere korrekt laplaceoperatoren på en gaussfunktion) ved differenser af to gaussfunktioner. Den bedste approksimation opnås når forholdet mellem spredningerne er ca. 1.6 (se *Marr*). Benyttes differensen af to pyramidenivauer til approksimationen bliver forholdet acceptabelt tæt på denne. Lad $G_{(i)}$ være en gausspyramide, og lad $Expand$ være en funktion der givet et billede på niveau z genererer et billede på niveau z-1 ved interpolation. Eksempelvis vil en simpel (lineær) interpolation være givet ved:

$$G(x,y,z-1) = \begin{cases} G(x/2, y/2, z) & \text{hvis x og y er lige} \\ \frac{G(\frac{x-1}{2}, y/2, z) + G(\frac{x+1}{2}, y/2, z)}{2} & \text{hvis x er ulige og y er lige} \\ \frac{G(x/2, \frac{y-1}{2}, z) + G(x/2, \frac{y+1}{2}, z)}{2} & \text{hvis x er lige og y er ulige} \\ \frac{G(\frac{x-1}{2}, \frac{y-1}{2}, z) + G(\frac{x+1}{2}, \frac{y+1}{2}, z)}{2} & \text{hvis x og y er ulige} \end{cases} \tag{4.33}$$

Andre interpolationsmetoder er beskrevet i afsnit 1.2. Laplacepyramiden $L_{(i)}$ defineres ved:

$$L_i = G_i - Expand(G_{i+1}) \tag{4.34}$$

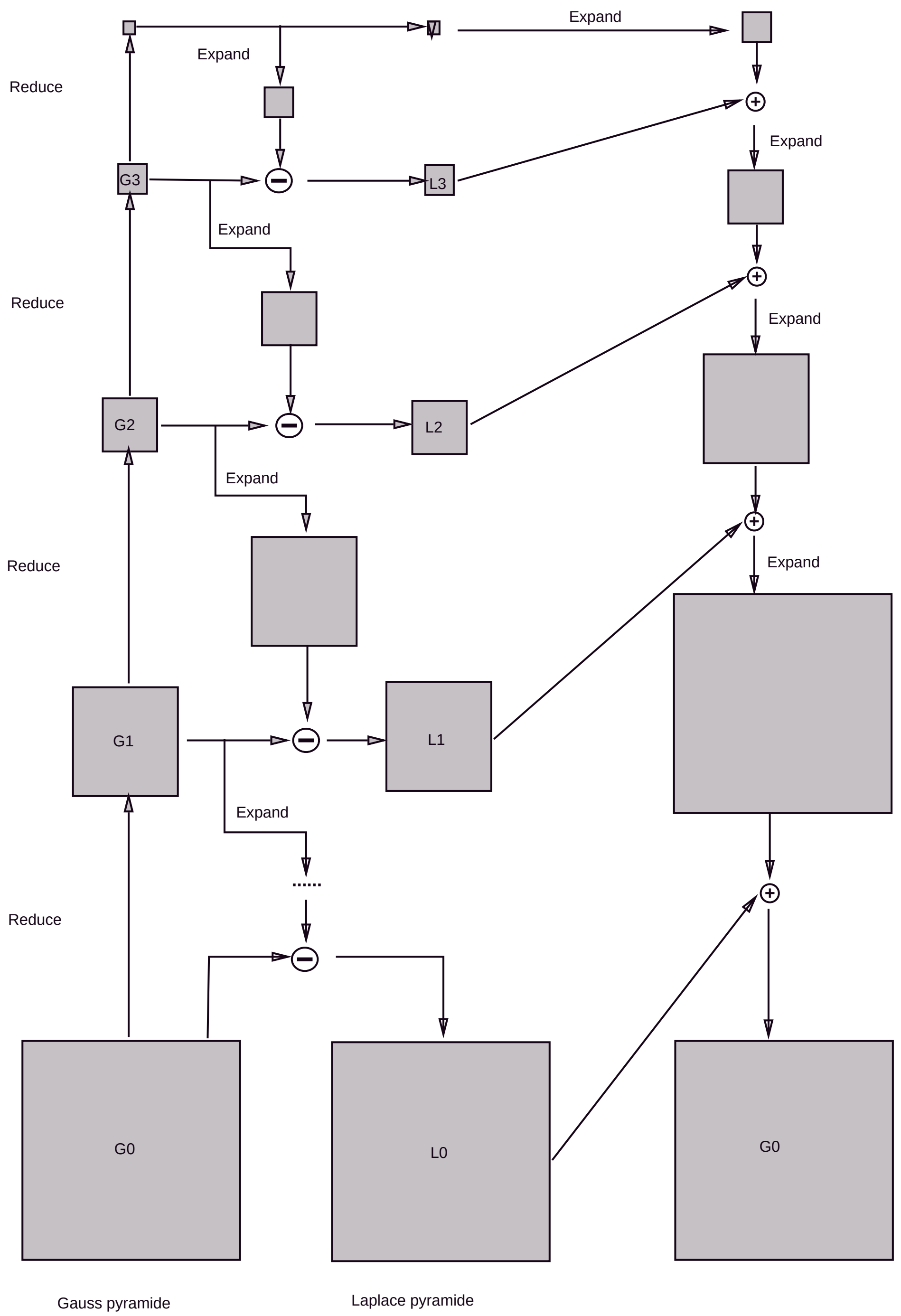
Laplacepyramider er ofte fulde, sådan at (1-pixels) billedet på største niveau er identisk med det øverste billede i gausspyramiden. Dette betyder at det oprindelige billede E kan skrives som summen af de ekspanderede laplace-billeder:

$$E = L_0 + Expand(L_1 + Expand(L_2 + ... + Expand(L_s))) \tag{4.35}$$

hvor s er niveaunummeret for det øverste billede i pyramiden. Bemærk endvidere at laplace-gaussfunktionen er et bånd-pass filter. Dette betyder at Laplacepyramiden er en repræsentation hvor hvert billede i pyramiden er en repræsentation af et vist frekvensområde i det oprindelige billede. Laplacepyramider kan, som beskrevet i afsnit 4.14, benyttes ved billedkodning. Alternativt kan de danne grundlag for en "hierakisk" (skala-bestemt) detektion af kanter.

4.5 Kant- og segmentrepræsentationer

Som diskuteret i kapitel 2 angiver kanterne i et billede de positioner hvor der (uformelt sagt) sker noget. Det er således oplagt at beskrive et billede ved en følge $\{(x,y)\}$ af kantpunkter. Afhængig af den skala hvor kanterne er bestemt vil repræsentationen mere eller mindre nøjagtigt/robust beskrive billedet. Det er teoretisk interessant at afgøre hvor godt positionerne af kantpunkterne (og disse alene) bestemmer billedet. Forskningen er imidlertid ikke nået så langt hvad dette angår. Dog har det faktisk været muligt at rekonstruere billeder godt ud fra denne information alene: *Curtis, Oppenheim and Lim: Signal Reconstruction from Fourier Transform Sign Information; IEEE trans. on Acoustics, speech, and signal*



Figur 4.1: Gauss- og Laplacepyramider

processing vol. 33 no. 3, 1985. Rekonstruktionen er dog ikke numerisk stabil, og kræver meget nøjagtig lokalisering af kanterne. I praksis er repræsentationen (i sin rå form) ikke velegnet hvis en egentlig rekonstruktion af billedet skal foretages. Som kompensation for en meget nøjagtig lokalisering af kanterne er en mulighed i stedet at knytte en række attributter til hvert kantpunkt. Ofte benyttede attributter er intensitetsgradientens retning og størrelse i punktet. Alternativt kan intensitetsværdierne på hver side af kanten registreres.

Repræsentation af billeder ved kanter er især attraktiv når formålet er en senere detektion af objekter ud fra deres omrids/silhouet i billedet. Dette emne diskuterede vi i kapitel 2. Kantrepræsentationens anvendelighed som udgangspunkt for en kodning af billedet diskuteres i afsnit 4.15. Dog kan vi her notere at hvis hver koordinat repræsenteres ved en 16-bits størrelse, og procentandelen af kantpunkter i billedet er p, da vil kompressiomsfaktoren for et gråtonebillede være $F = 25/p$. Kodning af kanter på denne (usmarte) måde kan altså ikke generelt lede til nogen høj kompressionsfaktor.

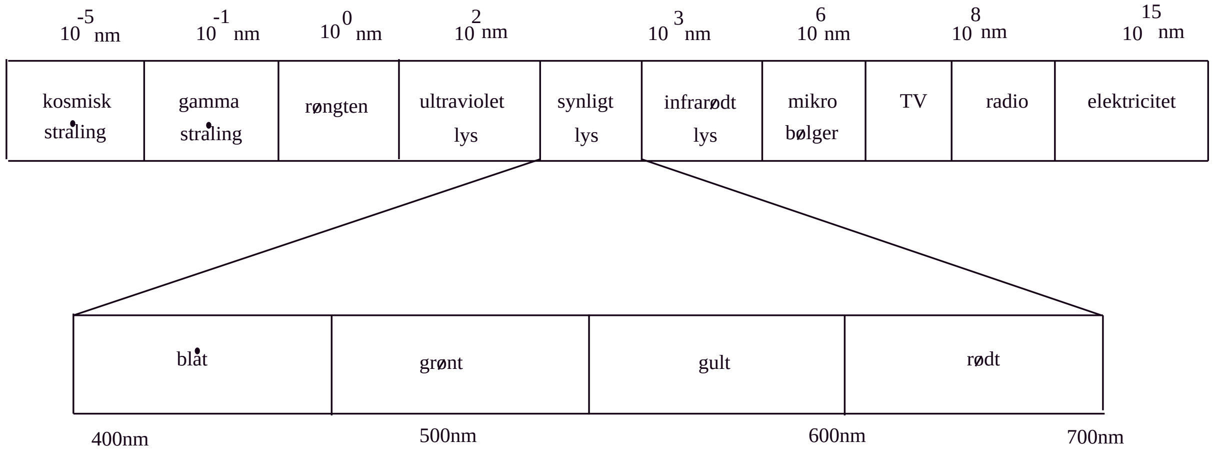
I forbindelse med repræsentation af billeder skal vi kort minde om skalarumsrepræsentationen (omtalt i kapitel 2). Denne har i de senere år tiltrukket sig en del opmærksomhed, men har endnu ikke vist sig praktisk anvendelig. En årsag er at teorien endnu er uafklaret for billeder. For signaler findes en række interessante resultater. Et af disse skal kort nævnes. Betragt "fingeraftrykket" (se afsnit 2.6) for et signal. Et *toppunkt* i skalarummet defineres som et punkt (x, σ) hvor to kanter mødes og forsvinder. Lad T = mængden af toppunkter. Antag at signalet er båndbegrænset. Det kan da vises at T entydigt bestemmer signalet pånær en konstant. Dette betyder at T udgør en hierakisk repræsentation af signalet. En sådan repræsentation er attraktiv ved en "grov-til-fin" analyse af signalet. Det kan imidlertid også vises, at en lille ændring i signalet kan ændre lokaliseringen af toppunkterne. Repræsentationen er således ustabil. Vi skal ikke her diskutere skalarumsrepræsentationen yderligere.

Som en dual til repræsentation af et billede ved kanter kan vi repræsentere billedet ved segmenter. Dette kræver oplagt adgang til en segmenteringsalgoritme af meget høj kvalitet. Som beskrevet i Kapitel 3 er segmentering af billeder i sig selv et svært problem. Hertil kommer problemerne med at repræsentere segmentgrænserne og intensitetsfunktionens opførsel inden for hvert segment. Som vi skal diskutere videre i afsnit 4.16 er det i visse sammenhænge tilstrækkeligt at kunne approksimere intensitetsfladen inden for hvert segment f.eks. med et polynomie af lav grad . Hvis et billede kun består af relativt få segmenter, alle med glat varienende intensitet (ingen tekstur), og hvis segmenternes omrids ikke er for "snørklede", da er der mulighed for særdeles store kompressionsfaktorer ved kodning. Svarende til situationen for kantrepræsentationen er repræsentation ved segmenter særdeles attraktiv når formålet er en efterfølgende genkendelse af objekter i billedet.

4.6 Repræsentation af farvebilleder

Farvebilleder bliver stadig mere almindelige i digital billedbehandling. Farver giver ofte et godt udgangspunkt for segmentering og klassifikation af mange bileder. Desuden kan menneskets evner til at skelne mellem farver ofte udnyttes ved præsentation af data, der ellers er komplicerede at se mønsteret i. Indtil omkring 1985 var brugen af farvemonitorer relativ sjælden (og kostbar). Typisk kunne man vise op til 256 forskellige samtidige farver ud af en større palette af farver. Først i 1990'erne er farvemonitorer blevet mere almindelige, såvel som at opløsningen på monitorerne er øget. Dog er det stadig meget kostbart at anskaffe monitore med såkaldt *fuld farve.* For almindelige farvemonitorer vil (8-bits) gråtoneværdien angive en indgang i en 24-bits farvetabel. Tabellen indeholder referencer til de mulige farver i paletten , og kan som regel ændres interaktivt ved visning af et billede. Herved kan man farvelægge et gråtonebillede således at bestemte værdier fremhæves. Dette betegnes *false color* eller *pseudo color* visualisering.

Før vi diskuterer de forskellige farvesystemer er det vigtigt at huske hvad lys og farvet lys er. Synligt lys er den del af det elektromagnetiske spektrum med en bølgelængde mellem ca. 400 og ca. 700 nanometer. Som illusteret i nedenstående figur er dette interval omgivet af ultraviolet lys (kortere bølgelængde) og infrarødt lys (længere bølgelængde). Længere ude i begge ender af det elektromagnetiske spektrum findes der også områder benyttes til optagelse af billeder.



Hvidt lys kan opspaltes i bånd af blåt, grønt, gult, og rødt lys. Båndene er ikke skarpe, idet farverne glider glat over i hinanden. Man taler ofte om farverne: rød, grøn og blå som de tre *primærfarver* . Ifølge international standardisering svarer disse farver til bølgelængderne: rød = 700nm, grøn = 546.1nm, og blå = 435.8nm. Ved at superponere forskellige mængder af lys med disse tre bøngelængder er det muligt at give et visuelt indtryk af næsten enhver farve. Dette skyldes at vores retina er forsynet med tre typer farvesensorer. Disse kan (meget forsimplet), på registrere den udsendte linearkombination af de tre frarver. Det er imidlertid vigtigt at huske, at der reelt kun er tale om tre forskellige bølgelængder, samt at det ved linearkombinationen ikke er muligt at konstruere enhver farve. Overlejres tre områder med rødt, grønt og blåt lys vil resultatet blive hvidt lys. De øvrige kombinationer er:

-	rødt og grønt	giver	gult
-	rødt og blåt	giver	magenta
-	grønt og blåt	giver	cyan

Farverne gul, cyan, og magenta kaldes for *sekundærfarverne* . Overlejrer man de tre sekundærfarver fås sort. Anvendelsen af de to farvesystemer RGB (rød, grøn, blå) hhv. CMY (cyan, magenta, yellow) diskuteres i det følgende afsnit.

4.6.1 Farvesystemer

I *full color* visualisering benyttes typisk tre gråtonebilleder til at angive farven. Farven i en bestemt pixel er altså bestemt ved tre 8-bits størrelser. For farvemonitorer angiver disse størrelser (som regel) mængden af rødt, grønt, og blåt lys i pixlen, svarende til anvendelse af RGB-farvesystemet. Farverne på en monitor fremkommer ved at en elektronkanon exiterer de små områder på monitoren, der hver (ved tilbagefald) udsender lys i en af de tre primærfarver. Hver af de tre 8-bits billeder styrer en elektronkanon, der genererer pulser svarende til de røde, grønne, og blå områder på monitoren. Intensiteten af et RGB-signal er defineret ved summen af de tre komponenter:

$$I = R + G + B \tag{4.36}$$

De relative farvekomponenter, også kaldet de *trikromatiske koefficienter* er tilsvarende defineret ved:

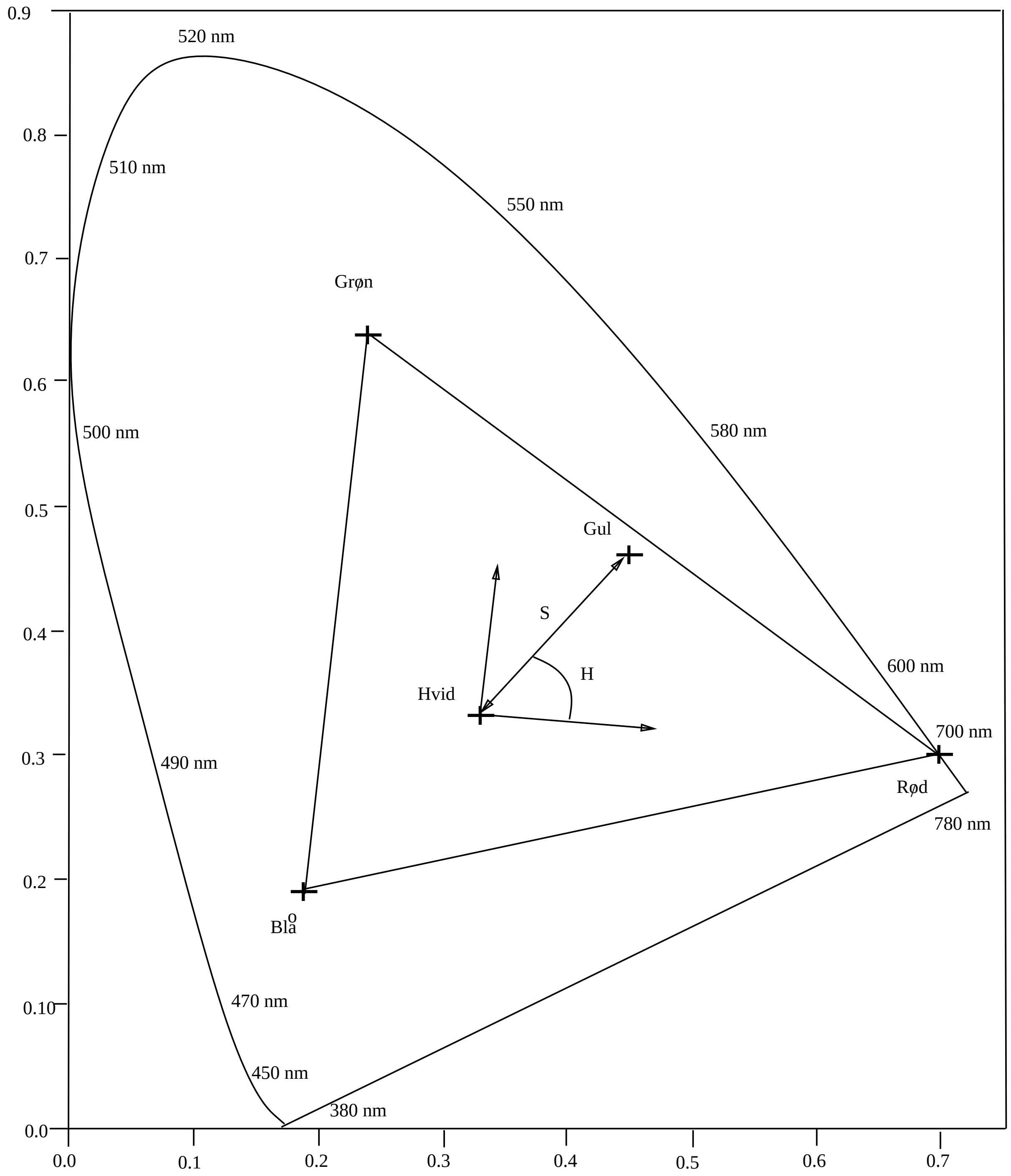
$$\begin{aligned} r &= \frac{R}{R + G + B} \\ g &= \frac{G}{R + G + B} \\ b &= \frac{B}{R + G + B} \end{aligned}$$

Farverummet er således udspændt af rgb-vektorene (1,0,0), (0,1,0) og (0,0,1). Sort svarer til Origo i systemet, hvidt til punktet (1,1,1). Sekundærfarverne (cmy) er placeret i kubens øvrige tre hjørner: (0,1,1), (1,0,1), og (1,1,0). I det normaliserede RGB-system vil der gælde at $b = 1 - r - g$. Vi kan derfor danne et *kromaticitets-diagram* ved i første kvadrant af et rg-koordinatsystem at afbilde farven (r,g,b). Hvidt lys vil i dette diagram have koordinaterne $W = (\frac{1}{3}, \frac{1}{3})$. Fareven blå vil blive afbilder nær Origo, farven grøn i toppen af diagrammet etc.

CMY-systemet er således defineret ved:

$$\begin{bmatrix} c \\ m \\ y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} r \\ g \\ b \end{bmatrix} \tag{4.37}$$

CMY-systemet benyttes fortrinsvis i farveprintere. I modsætning til superpositionsprincippet, der anvendes i farvemonitorer er der her tale om et subtraktionsprincip. Hvis et papir, der er malet med cyanfaret pigment belyses med hvidt lys vil det røde lys blive absorberet


520 nm
510 nm
550 nm
Grøn
580 nm
500 nm
Gul
S
H
Hvid
600 nm
700 nm
490 nm
Rød
780 nm
Blå
470 nm
450 nm
380 nm
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.10
0.0
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7

(omdannet til varme). Cyan subtraherer på denne måde det røde lys fra det hvide lys. Tilsvarende reflekteres kun grønt lys fra et magenta-farvet materiale. Overlejres pigmenter for magenta og cyan vil der således kun blive reflekteret lys med en bølgelængde svarende til blåt.

Ved transmision af fjernsynsudsendelser benyttes YIQ-systemet . Dette er defineret ud fra RGB-systemet ved:

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{4.38}$$

Bemærk her at størrelsen I intet har at gøre med intensiteten. Tværtom angiver Y den monokrome komponent af signalet, mens komponenterne I (inphase) og Q (quadrature) specificerer farvesammensætningen. Fordelen ved anvendelsen af YIQ-systemet er at Y-komponenten svarer til den monokrome komponent i standard sort/hvid-fjernsyn. Endvidere kan farvekomponenterne I og Q i praksis samples sjældnere end Y-komponenten hvilket giver mulighed for en datakompression.

I mange billedbehandlingssituationer ønsker vi ikke at ændre farveinformationen, men kun intensitetsforholdene. Dette er eksempelvis tilfældet ved histogramlinearisering. Hvis lineariseringen foretages på RGB-komponenterne hver for sig vil det resulterende billede få farver, der næppe har stor lighed med originalen. I andre tilfælde er man intereseret i, interaktivt, at specificere bestemte farver. Dette er tilfældet ved de såkaldte "paint-box"-systemer, der benyttes a visse kunstnere. I RGB-systemet er det meget vanskeligt at angive den RGB-værdi der svarer til en bestemt farve. I sådanne situationer er HSI-systemet ofte at foretrække. De tre komponenter: *hue* , *saturation* og *intensitet* angiver farvetonen, mætningsgraden af farven og lysheden.

Indlægges i kromaticitetsdiagrammet et lokalt koordinatsystem med Origo i W, og med førsteakse parallel med r-aksen kan H- og V-værdierne i HSI-systemet beskrives ved farvens polære koordinat. Således vil værdien af S angvive hvor langt farven ligger fra punktet W. Igen skal man huske at de farver, der kan frembringes er linearkombinationer af primærfarverne, og derfor indeholdt i den optegnede trekant i kromaticitetsdiagrammet.

4.6.2 Transformation mellem RGB of HSI systemerne

Transformationerne mellem RGB-systemet og HSI-systemet er ganske kompliceret. I det følgende opsummeres transformationerne. En grundig beskrivelse af hvorledes disse er fremkommet kan f.eks. findes i: *Gonzales and Woods: Digital Image Processing, Addison Wesley 1992.* I transformationen fra RGB til HSI gælder der at:

$$I = \frac{1}{3}(R + G + B) \tag{4.39}$$

For B $\leq$ G gælder der at:

$$H = \cos^{-1}\left\{\frac{1}{2}\frac{(R-G) + (R-B)}{\sqrt{(R-G)^2 + (R-B)(G-B)}}\right\} \tag{4.40}$$

hvis B > G benyttes (360° - H) i stedet for H. Endelig defineres:

$$S = 1 - 3\frac{MIN[R,G,B]}{R+G+B} \tag{4.41}$$

Ved transformationen fra HSI-systemet til RGB-systemet skal der opdeles efter hvilken farvesektor som H angiver, dvs. RG-sektoren (fra 0 til 120 grader), GB-sektoren (fra 120 til 240 grader), og BR-sektoren (fra 240 til 360 grader). I RG-sektoren fås:

$$\begin{aligned} b &= \frac{1}{3}(1-S) \\ r &= \frac{1}{3}\left[1 + \frac{S cos(H)}{cos(60^\circ - H)}\right] \\ g &= 1-(r+b) \end{aligned} \tag{4.42}$$

For GB-sektoren fås:

$$\begin{aligned} \hat{H} &= H - 120^\circ \\ r &= \frac{1}{3}(1-S) \\ g &= \frac{1}{3}\left[1 + \frac{S cos(\hat{H})}{cos(60^\circ - \hat{H})}\right] \\ b &= 1-(r+g) \end{aligned} \tag{4.43}$$

Endelig fås for BR-sektoren: For GB-sektoren fås:

$$\begin{aligned} \hat{H} &= H - 240^\circ \\ g &= \frac{1}{3}(1-S) \\ b &= \frac{1}{3}\left[1 + \frac{S cos(\hat{H})}{cos(60^\circ - \hat{H})}\right] \\ r &= 1-(g+b) \end{aligned} \tag{4.44}$$

Sluttelig kan RGB-værdierne beregnes ved: $(R,G,B) = 3I(r,g,b)$.

4.6.3 Reduktion af farverummet

Visse flybårne og sattelitbårne skannere er udstyret med et stort antal sensorer, der hver er er i stand til at registrere lys af et vist bølgeinterval. Typisk er der mellem 4 og 20 bånd i et sådant multispektralt billede , men visse billedoptagesystemer er udstyret med over 200 spektrale kanaler. Ofte overlapper de frekvensområder hvor de enkelte sensorer er følsomme. Dette betyder at informationsindholdet i billedsættet er yderst redundant. Som omtalt i afsnit 4.3 er én måde at reducere antallet af bånd at foretage en principalkomponentanalyse på det multispektrale billede. Ved nøje inspektion af de transformerede billeder hørende til hver egenværdi er det ofte muligt at udvælge et lille antal billeder med relevant information. Hverved kan farverummet, som det blev repræsenteret ved det multispektrale billede, reduceres.

En anden situation hvor man ønsker at reducere farverummet er hvis et *full color* billede skal vises på en monitor med et lille antal mulige forskellige samtidige farver. Typisk ønskes et 24-bits RGB-billede vist på en 8-bits farvemonitor. En oplagt mulighed er at gennemløbe RGB-billedet og registrere de 256 mest brugte farver. Pixels med andre farver tildeles en ny farve, der bestemmes som den af de 256 registrerede som ligger tættest på den oprindelige farve. Denne metode virker ofte glimrende hvis billedet består af store ensfarvede flader. I andre situationer, f.eks. hvor de 256 mest brugte farver er "nabofarver" er denne metode næppe ønskelig. Ofte vil små områder med markant afvigende farve ikke være synlige efter reduktionen. I stedet kan man analysere histogrammet over de benyttede farver. En mulighed er at finde de 256 største toppe i en (evt. udglattet) version af histogrammet. Denne metode kan oplagt give problemer hvis histogrammet er meget jævnt. Det kan her bemærkes at situationer hvor der findes mere end 256 omtrentlig lige store toppe med omtrentlig samme indbyrdes afstand i farvehistogrammet altid vil give problemer uanset reduktionsmåden. Ved reduktion af farverummet ønsker vi overordnet at finde N farver således at:

1. Antallet af pixels med hver af de N farver er stor.
2. Afstanden mellem hver af de N farver er stor.
3. Middelfejlen ved reduktionen er lille.

Hertil kan komme kriterier der udtrykker hvilke farve som øjet har sværere/lettere ved at skelne. Endelig er prioriteringen af ønskerne oplagt afhængig af formålet med visualiseringen.

Som nævt overnfor er algoritmer, der ensidigt forsøger at minimere antallet af pixels hvor der sker nogen farveændring, ofte utilstrækkelige. Alternativt findes algoritmer hvor der (efter at første farve er valgt som den mest hyppige) iterativt vælges en ny farve, der (blandt de aktuelle i billedet) er karakteriseret ved at have størst mulig afstand til de allerede udvalgte farver. På denne måde sikres at afvigende detaljer ikke tabes under reduktionen. Problemet er at store områder med en glat varierende farvetoning ofte gengives med én eller meget få farver.

I programmet *imxv*, der benyttes på DIKU benyttes en blanding af ovennævnte to metoder. Først bestemmest den hyppigst benyttede farve. De næste 10 farver bestemmes iterativt som de farver der ligger længst væk fra de allerede udvalgte. Derefter alterneres mellem valg af farve på basis af hyppighed hhv. afvigelse fra allerede udvalgte farver.

Kasseopdeling

Der findes andre klasser af metoder til opdeling af farverummet. I en klasse af metoder opbygges først (som i ovennævnte metoder) et (tre-dimensionalt) histogram over den aktuelle farvespredning. Herefter lægges en kasse i RGB-kuben tæt rundt om de benyttede farver. Denne kasse opdeles nu i to delkasser. En af disse udvælges og opdeles etc. Processen fortsætter iterativt indtil det ønskede antal farver (kasser) er fundet. Derefter vælges en repræsentant for hver kasse som den farve hvormed alle oprindelige farver i kassen skal gengives ved. En overordnet opdelingsalgoritme er:

1. Vælg den kasse der skal opdeles.

2. Vælg den dimension langs hvilken den udvalgte kasse skal opdeles.

3. Vælg den værdi hvorved kassen skal opsplittes.

Det er let at forestille sig mange kriterier for de tre valg. En oplagt mulighed er at vælge den kasse der rummer flest pixel, at vælge den dimension hvor farvespredningen er størst, og at vælge medianen af fordelingen som grænseværdi. Denne metode vil tilnærme et farvevalg hvor antallet af pixel med hver farve er omtrentlig lige stor. Dette er oplagt ikke altid en god ide. Alternativt kan de tre valgt foretages ud fra ønsket om minimering af en straffunktion. Denne vil typisk indeholde et udtryk for middelkradratfejlen ved approximation af farverne med den bedst mulige repræsentant (middelfarven). Hvad angår det tredie af de ovennævnte valg kan tidligere overvejelser om segmentering på basis af et intensitetshistogram gentages her.

Klyngedetektion

En tredie klasse af metoder til farvereduktion stammer fra den gruppe af metoder til mønstergenkendelse der omhandler *uovervåget klassifikation*. I disse metoder søger man at gruppere de aktuelle observationer i klynger, således at data fra samme klynge kan tænkes at stamme fra samme "proces". I visse af metoderne kræves at antallet af klynger er kendt på forhånd (svarende til det aktuelle problem i farvereduktionen).

I *K-means algoritmen* vælges K initielle klyngecentre z_0^0, z_1^0, op til z_K^0. Herefter bestemmes iterativt et nyt sæt af klyngecentre. Dette sker ved først at mærke hver observation (farve) med den klynge, der ligger tættest på observationen. Dernæst beregnes (for hver klynge) det ny center som middelværdien af observationerne tilhørende klyngen. Genberegningen af klyngecentre fortsætter så længe at der blot er ét center der flytter sig. Der findes intet bevis for konvergens af algoritmen. I praksis konvergerer metoden forholdsvist hurtigt. Det skal bemærkes at det endelige sæt af klyngecentre er meget følsomt over for antaller af klynger K, samt valget at initielle klyngecentre. En oplagt mulighed er at benytte en af de tildigere beskrevne metoder til valg af initielle klyngecentre. Endelig skal det bemærkes at valget af afstandsmålet naturligvis også er betydende for resultatet.

En simplere metode er ikke at kræve kendskab til antal klynger, men kun til den største tilladelige afstand T mellem observationerne i en klynge. Første vælges en farve (f.eks. den hyppigste). Denne farve udgør det første klyngecenter z_0. Dernæst gennemløbes samtlige farver. Alle farver, hvis afstand til z_0 er mindre end T tilknyttes den udvalgte farve. Blandt de resterende farver udvalges den hyppigst forekomne. Denne farve udgør det andet klyngecenter z_1. Dernæst gennemløbes de resterende farver, og afstandene til alle tidligere fundne klyngecentre beregnes (for hver farve). Hvis den mindste af disse afstande er større end T gemmes farven til næste runde, etc. Det er klart at metoden er særdeles følsom mht. angivelse af værdien T.

En tredie meget benyttet algoritme er *Maximin-algoritmen*. Først vælges en farve (f.eks. den hyppigste) som det første klyngecenter z_0. Dernæst vælges den farve, der ligger længst fra z_0. Denne farve vil bestemme klyngecenteret z_1. Herefter tilføjes nye klygecentre iterativt indtil en vis betingelse er opfyldt. Lad T være afstanden mellem farverne z_0 og z_1. Metoden er nu :

1. Beregn for hver af de resterende farver x afstanden r_{x,z_i} til alle klyngecentre z_i, og registrer (for hvert x) den mindste af disse afstande r_x^{min}.

2. Find den farve y der blandt de resterende farver x der maksimerer afstanden r_x^{min}, og lad r_0 være denne afstand.

3. Hvis $r_0 \geq T/2$ er et nyt klyngecenter fundet. Dette har farven y. Lad T være middelværdien af aftandene mellem de nu registrerede klyncegentre. Fortsæt søgningen efter nye klynger som beskrevet under punkt 1).

4. Hvis $r_0 < T/2$ er alle klyngecentre fundet. Alle farver løbes igennem og knyttes til det klyngecenter (farve), der ligger tættest på farven.

En fordel ved *Maximin-algoritmen* er at den ikke kræver angivelse af nogen parameter. Det skal dog bemærkes at algoritmen er følsom mht. valg af det første klyngecenter. Envidere skal det bemærkes (som for de øvrige metoder) at afstandsbegrebet ikke er specificeret. I mange sammenhænge vil det sædvanlige euclidiske afstandsbegreb være hensigtsmæssigt. Valget af afstandsfunktionen bør dog altid ske ud fra en analyse af de aktuelle data og deres fortolkning (samt ud fra beregningsmæssige hensyn).

Endelig skal *ISODATA-algoritmen* (Iterative Self-Organizing Data Analysis Techniques A) nævnes. Denne er en ganske kompliceret algoritme, der minder om *K-means-algoritmen*, men er udvidet med en lang række heuristikker, som på basis af års erfaring har vist sig gavnlige for virkemåden. Det vil føre for vidt at beskrive algoritmen i detaljer her. En grundig beskrivelse kan f.eks. findes i *Tou and Gonzales: Pattern recognition Principles. Addison-Wesley 1974.*

4.7 Billedkodning

Kodning af billeder er især af interesse ved lagring og transmission af disse. Mest banalt kan kodningen foretages ved kvantificering (f.eks. ved bortsmidning af de mindst betydende bit), ved gruppering af billedelementer med samme gråtoneværdi eller ved stikprøvetagning (sampling) sådan at billedet repræsenteres ved færre pixels. Disse metoder giver imidlertid kun en acceptabel kompressionsfaktor for binære (eller næsten binære) billeder. Mere avanceret kan de enkelte gråtoneværdier erstattes af en variabel-længde bit-kode, sådan at gråtoner der forekommer hyppigst har kort bit-kode. Sådanne avancerede metoder (som eksempelvis Huffman-kodning) er imidlertid "blinde" overfor den struktur der ofte er i billeder.

Overordnet findes to forskellige metoder til kodning af billeder, nemlig de der tillader en fuldstændig rekonstruktion af billedet (reversible kodninger) , og de der kun tillader en approksimativ rekonstruktion (ikke-reversible kodninger). Førstnævnte metode benyttes eksempelvis ved kompression af alle filtyper (incl. billeder) på instituttets UNIX-anlæg. Den metode der oftest benyttes kaldes "adaptiv Lempel-Ziv kodning" og er beskrevet i *A technique for High Performance Data Compression, Terry A. Welch, IEEE Computer vol. 17, no 6, june 1984 pp 8-19.* Metoden er at opbygges en tabel over delstrenge under kompressionspassagen. Hver delstreng tilknyttes en 9-bits kode, og efterfølgende fælles delstrenge erstattes af denne kode. Når de første 511 delstrenge er tabelleret erstattes 9-bits koden af en 10-bits kode. Sådan fortsættes op til kodelængden er 16 bit. Løbende foretages målinger af kompressionsraten. Hvis denne udviser faldende tendens gemmes tabellen over de hidtil fundne delstrenge, og en ny tabelopbygning startes. Dette betyder at metoden i en vis grad er adaptiv. Ved anvendelse af denne algoritme på billeder opnås typisk en kompressionsfaktor på under 2. Dette er for ringe. Ved transmission af billeder ønsker vi kompressionsfaktorer på mellem 10 og 100. Problemet er at algoritmen ikke udnytter den struktur der er i (de to-dimensionale) billeder.

Reversible kodningsmetoder er ikke så effektive mht. kompression som ikke-reversible metoder. Til gengæld er det ofte ikke noget krav at kunne rekonstruere billedet absolut korrekt. Vi skal derfor koncentrere opmærksomheden mod ikke-reversible kodningsmetoder. Mange af disse kan komprimere et billede med en faktor 10 samtidig med at billedet kan rekonstrueres så godt, at der skal en meget grundig inspektion til at afsløre forskellen fra originalbilledet. Forståelsesmæssigt kan de fleste metoder til ikke-reversibel billedkodning beskrives ved følgende tre faser:

$$\rightarrow Transformation \rightarrow Kvantificering \rightarrow Indkodning \rightarrow \tag{4.45}$$

Der findes dog også metoder der ikke følger denne faseopdeling. I det følgende introduceres til faseopdelingen, der tænkes anvend på hele billedet eller på et sæt af disjunkte delbilleder (blokke) af dette.

Ved transformationen vælges en ny repræsentation for billedet. Formålet er at minimere korrelationen mellem intensitetsværdierne i billedet. Desuden er det ønskeligt at transformationen pakker så meget information som muligt sammen i så få transformationskoefficienter som muligt. I afsnit 4.8 beskrives nogle hyppigt anvendte transformationer. Det bør bemærkes at der sjældent sker nogen datakompression under transformationen, snarere tværdigmod.

Formålet med kvantificeringen er at komprimere det transformerede billede. Dette sker på to måder: 1) Ved bortsmidning af koefficienter, der kun indeholder ringe (uvæsenlig) information, 2) Ved reduktion af værdiområdet (egentlig kvantificering). En retningslinie for bortsmidning af koefficienter er kun at bevare en vis procentdel af energien i billedet. Opdeling af værdiområdet i *kvanter* sker ofte på basis af et krav om at minimere kvantificeringsfejlen. Metoder til kvantificering diskuteres i afsnit 4.9.

Før lagring eller transmission af det transformerede og kvantificerede billede indkodes koefficienterne således at disse optager ringe (mindst mulig) plads. Typisk foregår dette ved at benytte variabel længde bitkoder. Værdier, der forekommer hyppigt tildeles korte koder. Et eksempel på en sådan metode er Huffman-kodningen. Metoder til indkodning diskuteres i afsnit 4.10.

Visse (simple) metoder følger ikke den ovenfor beskrevne faseopdeling. I afsnit 4.11 til 4.13 beskrives tre sådanne særdeles anvendelige og meget benyttede metoder.

De fleste af de klassiske metoder til billedkodning opnår sjældent kompressionsfaktorer over 10. En af årsagerne til dette er at de enkelte pixels betragtes som isolerede størrelser. Hvis højere kompressionsfaktorer skal opnås tyder meget på at det er nødvendigt i højere grad at betragte *strukturene* i billedet. Dette er i nogen grad tilfældet ved tre nyere metoder til billedkodning der beskrives i afsnit 4.14 til 4.16. Især de sidste to metoder har vist sig lovende (kompressionsfaktorer mellem 20 og 80 er rapporteret). Sluttelig introduceres i afsnit 4.17 til kodning af billedsekvenser. Sådanne kodninger har p.t. overordentlig stor bevågenhed, især inden for kommunikationsbranchen.

Når en given metode til kodning af billeder skal vurderes er der to modsat rettede kriterier. Dels ønsker vi en høj kompressionsfaktor, og dels ønsker vi at et kodet billede kan rekonstrueres godt. Det er meget let at forestille sig metoder, der tilfredsstiller hvert af ønskerne, men særdeles svært at få begge ønsker opfyldt samtidig. Kompressionsfaktoren kan måles som forholdet mellem antal bit i det oprindelige og i det kodede billede. Ved evalueringen af en metode er det ofte sædvane at se bort fra indkodningsfasen. I stedet måles konpressionsfaktoren ved forholdet mellem *entropien* af det oprindelige og det transformerede kvantificerede billede. Entropien er en reel størrelse og angiver det mindste antal bit, som det teoretisk set er muligt at kode en mængde af tal ved. Entropien af et billede kan altså udmærket være f.eks. 0.72. Entropibegrebet defineres og diskuteres i afsnit 1.10.

I modsætning til målet for kompressionen er det meget sværere at vurdere hvor god en given rekonstruktion af et kodet billede er. Det er naturligvis muligt at betragte middel kvadratafvigelsen mellem originalen og rekonstruktionen som mål for kvaliteten af rekonstruktionen. Dette, eller tilsvarende mål benyttes ofte i praksis. Imidlertid er der ofte uoverensstemmelse mellem dette mål og den subjektive opfattelse af lighed mellem billeder. Den rekonstruktion, der forekommer øjet optimal kan meget vel have en stor sådan middelrekonstruktionsfejl. Det er derfor “farligt” at søge metoder, der ensidigt forsøger at minimere en sådan fejl.

4.8 Transformationer

Formålet med transformationsfasen i den klassiske kodning er at opnå en repræsentation hvor afhængigheden mellem koefficienterne er elimineret. Koefficienterne siges så at være dekorreleret. Som vi tidligere har set opfylder Karhuen-Loeve transformationen dette krav. Denne transformation er imidlertid ikke velegnet ved kodning, idet sættet at basisbilleder hvorved transformationen er foretaget, er afhængig af billedet selv. Modtageren af koefficienterne kan altså ikke fortolke disse uden også at kende basisbillederne. Ved kodning vil man altid vælge en metode hvor både afsender og modtager kender basisbillederne. Eksempler på sådanne transformationer er fourier-, cosinus-, og Hardamard transformationerne. Ved disse metoder er det imidlertid ikke muligt at dekorrelere koefficienterne fuldstandigt. Lad f og g være to transformationer om hvilket der gælder at f bedre end g er i stand til at dekorrelere koefficienterne. Man kan da vise at f bedre end g er i stand til at pakke mest mulig information i få koefficienter. Informationen pakket i en given koefficient (svarende til et givet basisbillede) kan estimeres ved variansen (energien) af koefficientværdien målt over et stort antal transformationer. Ved sortering af koefficientværdierne efter variansværdi, fås et plot, der afbilder energien som funktion af koefficient index. Funktionen er aftagende. Jo hurtigere funktionen aftager jo bedre er informationen pakket i få koefficienter. Ved plot af den beskrevne funktion for Hardamard-, Fourier-, Cosinus-, og for Karhuen-Loeve transformationen vil man kunne iagttage at kurverne ligger under hinanen. Således er Hardamard transformationen dårligst til at pakke informationen (dekorrelere koefficienterne), Fourier transformationen lidt bedre, Karhuen-Loeve transformationen (ikke overraskende) bedst, og cosinus transformationen kun lidt ringere en Karhuen-Loeve transformationen og bedre end fourier transformationen. Kodningsmæssigt er cosinus transformationen derfor det bedste valg. Cosinustransformationen er for nyligt blevet valgt som ISO-standard for kodning af billeder *G. K. Wallace: Digital Images and Video Standards; Communications of the ACM vol. 34 no. 4, 1991.*

Ved transformationskodning vil man sjældent transformere hele billedet, men i stedet opdele dette i disjunkte blokke (delbilleder). I dette tilfælde tales om *blokkodning.* Som vi skal diskutere i næste afsnit er det i blokkodning muligt at kvantificere koefficienterne til hver blok uafhængigt. Vi skal her antage at alle blokke er kvadratiske med kantlængde n. Problemet er nu at vælge n. Det kan vises at for alle tranformationerne gælder, at jo større n vælges, jo bedre er koefficienterne i hver blok dekorreleret. Dette taler for at vælge n stor. Omvendt taler at korrelationen mellem intensitetsværdier i pixels hvis indbyrdes afstand er over 15-20 pixels som regel er forsvindende. Desuden vil "skakbordseffekten" ved en rekonstruktion ud fra kvantificerede koefficienter ofte øges med voksende blokstørrelse. Dette taler for at vælge n lille. I praksis bør n ikke vælges under 4 og ikke over 16. Et valg af n lig 8 er typisk.

4.9 Kvantificering

I dette afsnit beskrives nogle metoder til kvantificering af koefficienter fremkommet ved en transformation. Det antages at transformationen har dekorreleret koefficienterne fuldstændigt, således at vi kan kvantificere disse uafhængigt. Vi antager at den givne koefficient x kan betragtes som en kontinuert stokastisk variabel. Vi kan derfor tale om tæthedsfunktionen $p(x)$ for x. Den kvantificerede værdi af x betegnes q. Kvantificeringsfejlen er derfor $|x - q|$. Generelt betragtes værdimængden af x som opdelt i et antal *kvanter* (intervaller) $[z_i, z_{i+1}]$ således at der om den kvantificerede værdi q_i gælder at $z_i < q_i < z_{i+1}$. Kvantlængden $z_{i+1} - z_i$ betegnes Δ_i.

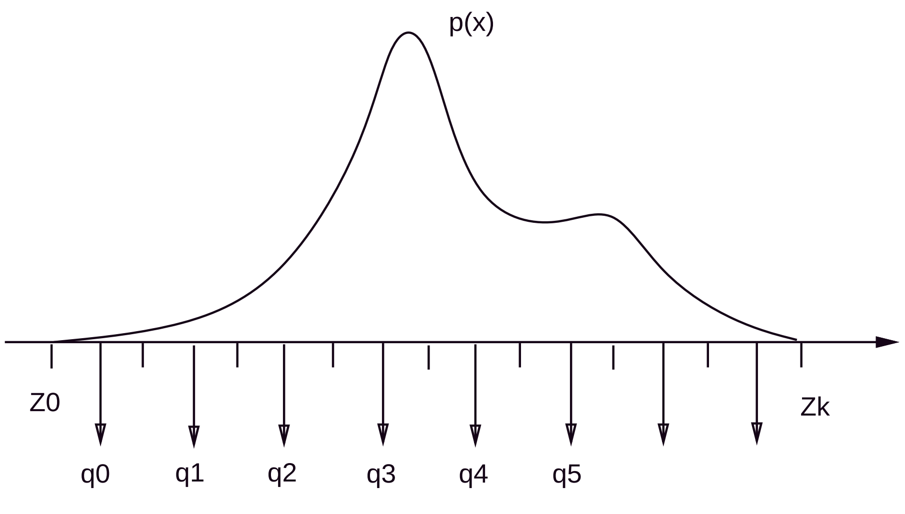
Kvadratmiddelfejlen ved kvantificering af $x \in [z_i, z_{i+1}]$ kan opskrives som:

$$e_i^2 \quad = \quad \int_{z_i}^{z_{i+1}} (x - q_i)^2 p(x) dx \tag{4.46}$$

Overalt-kvadratmiddelfejlen ved opdeling af værdiområdet i K kvanter (svarende til K værdier q_i) er da:

$$e_{ms}^2 \quad = \quad \sum_{i=0}^{K-1} \int_{z_i}^{z_{i+1}} (x - q_i)^2 p(x) dx \tag{4.47}$$

hvor z_0 hhv. z_K muligvis kan være $\pm\infty$ afhængig af værdimængden for x.



Figur 4.2: Kvantificering af en stokastisk variabel med fordeling $p(x)$, og med værdiområde $[Z_0\colon Z_k]$ i repræsentanter $q_0, q_1, \ldots q_{k-1}$

Eksempel

Antag at x er ligefordelt i intervallet [0,1], og antag at $q_i = \frac{z_{i+1}+z_i}{2}$. Da er:

$$\begin{aligned}
e^2 &= \sum_{i=0}^{K-1} \int_{z_i}^{z_{i+1}} (x-q_i)^2 p(x)dx \\
&= K \int_0^{1/K} (x - \frac{1}{2K})^2 dx \\
&= \frac{1}{12K^2}
\end{aligned}$$

Heraf ses er rms-fejlen er $\frac{1}{\sqrt{12}K}$. For $K = 10$ får vi eksempelvis at $e_{rms} \approx 0.03$.

4.9.1 Uniform kvantificering

Ved uniform kvantificering har alle kvanter samme størrelse, dvs. $\Delta_i = \Delta$. Ved denne metode vælges, som i eksemplet ovenfor, $q_i = \frac{z_{i+1}+z_i}{2}$. Kvantificeringsmetoden er således meget enkel. De eneste parametre er antallet K af kvanter, samt positionen af første kvanteværdi q_0. Vi skal senere vende tilbage til denne metode.

4.9.2 Gradueret kvantificering

Antag at fordelingsfunktionen $p(x)$ er koncentreret omkring en bestemt værdi x_0, og at $p(x)$ aftager hurtigt når $|x - x_0|$ vokser. Ideen er da at øge kvantestørrelsen Δ_i med afstanden fra x_0. Anderledes sagt ønsker vi at mindske kvantificeringsfejlen for de fleste koefficientværdier. Gradueret kvantificering kan foretages mere eller mindre ad hoc. Dog er nedenfor beskrevne optimale kvantificering ofte den bedste af disse metoder.

4.9.3 Optimal kvantificering

I udledningen af den optimale kvantificering tager vi udgangspunkt i ligning (4.47). Vi ønsker at bestemme følgerne z_i og q_i således at e^2_{ms} minimeres. Dette gøres på sædvanlig måde:

$$\frac{\partial}{\partial z_i} e^2_{ms} = (z_i - q_{i-1})^2 p(z_i) - (z_i - q_i)^2 p(z_i) = 0 \qquad i = 2,3,\ldots,K \tag{4.48}$$

$$\frac{\partial}{\partial q_i} e^2_{ms} = -2 \int_{z_i}^{z_{i+1}} (x - q_i)p(x)dx = 0 \qquad i = 1,2,\ldots,K \tag{4.49}$$

For $p(z_i) \neq 0$ får vi:

$$z_i = \frac{q_{i-1} + q_i}{2} \tag{4.50}$$

$$q_i = \frac{\int_{z_i}^{z_{i+1}} xp(x)dx}{\int_{z_i}^{z_{i+1}} p(x)dx} \tag{4.51}$$

Resultatet er altså, at intervalgrænserne skal ligge midt imellem kvantificeringsværdierne q_i, og at kendskab til $p(x)$ er nødvendig for at beregne disse. For K kvanter har vi således $2K$ ligninger. Disse er imidlertid ikke lineære. Kun for særligt simple tæthedsfunktioner $p(x)$ kan vi angive analytiske udtryk for kvantificeringsværdierne. Generelt er det nødvendigt at benytte numeriske metoder, der benytter sig at sekvenser af gradvist bedre og bedre gæt på værdierne. En sådan algoritme er beskrevet i: *Max: Quantizing for Minimum Distortion; IRE trans. Informat. Theory IT-6, 1960.* Bemærk at i det særlige tilfælde hvor $p(x)$ er ligefordelt vil $q_i = \frac{z_i+z_{i+1}}{2}$. Dette er ikke overraskende.

Valget af kvantificeringsmetode er ikke så oplagt som man umiddelbart skulle tro. Den optimale kvantificering er bedst i den forstand, at middel kvantificeringsfejlen minimeres. Dette sker på bekostning af en meget dårlig kvantificering af de sjældent forekomne koefficientværdier. Hvis disse værdier imidlertid markerer "usædvanlige begivenheder" i billedet, og er væsenlige for en god rekonstruktion, er en god kantificering af disse mindst så vigtig som en god kvantificering af de hyppigst forekomne koefficientværdier. Ved optimal kvantificering kan det således være nødvendigt at benytte forholdsvist flere kvanter end for den uniforme kvantificering (for at opnå en acceptabel rekonstruktion). Eksperimenter indikerer, at dette forhold ikke kan ignoreres i praksis. Derfor er der, men mindre særlige forhold taler imod, god grund til at vælge en uniform kvantificering frem for den optimale.

4.9.4 Kvantificering ved blokkodning

Som nævnt i afsnit 4.8 har koefficienterne for en tranformeret blok meget forskellig varians. Anderledes sagt er værdiområdet for nogle koefficienter meget større end for andre koefficienter. Det er derfor ikke rimeligt at benytte samme kvantificering for alle koefficienter i en blok. Hvis vi gjorde dette ville visse kvanter for nogle koefficienter aldrig blive benyttet. En sådan kvantificering ville ikke være effektiv. Hvis værdiområderne for koefficienterne kendes på forhånd kan vi skalere koefficienterne således at de skalerede værdier vil have samme værdiområde. Skaleringskonstanterne kan bestemmes eksperimentelt ved transformation af et stort antal typiske blokke. Man kunne nu forestille sig at benytte samme kvantificering for alle (skalerede) koefficienter. Dette vil imidlertid også være urimeligt, fordi nogle koefficienter bærer meget energi, mens andre kun bærer ringe energi. Benyttes et fast antal kvanter vil kvantificeringsfejlen for alle koefficienter være numerisk ens. Da koefficienter med stor varians (energi) bidrager forholdsvist mere til rekonstruktionen, vil den samlede fejl i rekonstruktionen altså kunne mindskes ved at benytte flere kvanter for disse koefficienter. I *Gonzales and Wintz: Digital Image Processing; Addison-Wesley, 1987* anbefales således at antallet af kvanter for en koefficient vælges proportional med logaritmen til variansen af koefficienten, dvs. $\log(\sigma^2)$. En ofte benyttet metode, er først at bortsmide de koefficienterne med mindst værdi (f.eks. halvdelen af koefficienterne), derefter at skalere hver af de resterende koefficienter med variansen af koefficienten. Til slut kvantificeres og indkodes de skalerede værdier med bitkoder af fast længde l, således at der om den i'te koefficient gælder at l_i er proportional med $\log(\sigma_i^2)$. Denne metode kaldes *blok kvantificering.* Yderligere diskussion findes i f.eks. *Rosenfeld and Kak: Digital Picture Processing, Academic Press, 1982.*

4.9.5 Adaptiv kvantificering

De fleste mere avancerede metoder til blokkodning er adaptive i den forstand, at kvantificeringsmåden og/eller graden tilpasses hver blok. Tilpasningen kan foregå ud fra analyser af koefficientværdierne eller ud fra intensitetsfordelingen i blokken. Simplest er at bortsmide et antal koefficienter. Lad N være det totale antal koefficienter y_i og n antallet af bevarede koefficienter. Valget af n kan f.eks foretages således at:

$$\frac{\sum_{i=1}^{n} |y_i|^2}{\sum_{i=1}^{N} |y_i|^2} \geq T \tag{4.52}$$

hvor T er en prædefineret grænseværdi (f.eks. 0.99). I dette tilfælde kodes tallet n samt de n bevarede koefficienter. Alternativt kunne man vælge de n koefficienter hvis aktuelle værdi oversteg en vis prædefineret værdi T. Her er det nøvendigt at kode (f.eks ved run-længde kodning) en bitsekvens angivende hvilke koefficienter der bevares. Valget af de koefficienter, der bevares kan også foretages således at en bestemt brugerangiven kompressionsfaktor opnås.

En anden metode at tilpasse kvantificeringen på er, at analysere intensitetsfordelingen i blokken, og på dette grundlag vælge såvel antallet af bevarede koefficienter, samt hvor fint disse er kvantificeret. Lad m hhv. σ være middelintensiteten hhv. spredningen af intensiteten i blokken. Et foreslag til klassifikation af en blok er:

1. $m \geq T_m$ og $\sigma \leq T_\sigma$.
2. $m \leq T_m$ og $\sigma \leq T_\sigma$.
3. $\sigma \geq T_\sigma$.

hvor T_m og T_σ er prædefinerede grænseværdier. I første gruppe falder meget lyse områder med ringe detalie. Sådanne blokke kan kvantificeres brutalt ved kun at bevare få koefficienter og ved at kvantificere disse groft. I anden gruppe falder mørke områder med ringe detalie. Da detalier som regel springer mere i øjnene i mørke områder, er en finere kvantificering af sådanne områder ofte påkrævet. Relativt mange koefficienter kan også bortsmides for blokke i denne gruppe. For blokke i den tredie gruppe er det nødvendigt at bevare relativt mange koefficienter og at kvantificere disse finere. Et alternativ til ovennævnte gruppering er at bestemme kvantificeringen ud fra kendskabet til hvorvidt blokken indeholder kanter, samt hvor betydningsfulde disse er.

Da kvantificeringsfasen er hovedansvarlig for den egentlige komprimering af et billede er det især i denne fase at der forskes. Ved at forfine kriterierne for adaptiv kvantificering er det ofte muligt at forbedre kodningsegenskaberne betydeligt.

4.10 Indkodning

Ved indkodning af en følge af N elementer w_i omsættes disse (uden nogen form for kvantificering) til en følge c_i af bit. Det antages at værdierne af w_i er fuldstændigt dekorrelerede.

Vi skal antage at hvert element kan antage M mulige (forskellige) værdier. Indkodningen er reversibel, i den forstand, at der til hver værdi svarer en og kun en bitfølge, og at forskellige elementer tildeles forskellige bitkoder . Ved *fastlængde bitkode* tildeles hvert element en bitfølge af samme længde. Dette er f.eks. tilfældet ved den sædvanlige kodning af gråtoneværdier. Vi kunne imidlertid ordne gråtonerne i en vilkårlig rækkefølge. Hvis hver pixel kan antage $M = 256$ mulige værdier findes atlså 256! mulige forskellige indkodninger med fastlængde bitkode. Generelt vil hver værdi blive kodet ved $\log_2(M)$ bit. Ved *variabel længde* kodning benyttes koder c_i hvis længde varierer. En *entydig kode* udmærker sig ved at en given sekvens af kodeord kun kan afkodes på en måde. F.eks er variabel længde indkodningen $c_1 = 0, c_2 = 1, c_3 = 01, c_4 = 10$ ikke entydig fordi bitsekvensen 0011 kan afkodes dels som følgen $c_1c_1c_2c_2$ og dels som $c_1c_3c_2$. Vi er oplagt kun interesseret i entydige indkodninger. Ved afkodning af en bitfølge der er *direkte kodet* er det muligt at omsætte koden til elemter under læsningen af bitfølgen og uden at "se frem i denne" (eng. look ahead).

Det er muligt at vælge mange mulige indkodninger af en given sekvens. Vi er interesseret i at bestemme nogle entydige indkodninger, der er direkte (eller næsten direkte) og som resulterer i kodninger ved så få bit som muligt. Det sidste ønske udelukker i de fleste situationer indkodninger af fast længde. Den grundlæggende ide er at knytte korte bitkoder til hyppigt forekomne værdier, og lange koder til mere sældent observerede værdier.

Betragt den diskrete sandsynlighedsfordeling $p(v_i)$ af de M mulige værdier $v_1, v_2, \ldots, v_M$ for elementerne w_i. *Entropien* af sekvensen w_i defineres da ved:

$$H = -\sum_{i=1}^{M} p(v_i) \log_2(p(v_i)) \tag{4.53}$$

Entropien er et mål for graden af tilfældighed blandt tallene v_i. Hvis f.eks. $M = 8$, og alle værdier v_i er lige sandsnynlige, dvs. $p(v_i) = 1/8$ ses at:

$$H = -\sum_{i=1}^{8} \frac{1}{8} \log_2(\frac{1}{8}) = -\log_2(\frac{1}{8}) = 3 \tag{4.54}$$

Hvis omvendt alle værdier v_i bortset fra en, forekommer med en hyppighed på 0, ser vi at $H = 0$. Anderledes sagt, indeholder sekvensen ingen information. I førnævnte eksempel var entropien maksimal, dvs. at vi ingen information havde om sekvensen på forhånd. Generelt gælder der at $0 \leq H \leq \log_2(M)$.

Et væsenligt resultat fra informationsteorien siger at vi i middel ikke kan indkode nogen sekvens på færre bit end entropien angiver. Vi kan derfor bruge entropien til kvalitetsmåling af en given indkodning. For en sådan kan vi beregne:

$$\xi = \frac{\sum_{i=1}^{M} p(v_i)\beta(v_i)}{H} \tag{4.55}$$

hvor $\beta(v_i)$ angiver hvor mange bit v_i aktuelt er kodet ved. Tælleren angiver derfor middellængden af indkodningen. For en god indkodning vil ξ være tæt på 1.

Entropien benyttes ofte til vurdering af en kodningsmetode. Typisk foretages der (under udvikling af metoden) ingen indkodning. Ved vurderingen af metoden vil man sige: "Hvis vi nu indkodede de fundne transformerede kvantificerede koefficienter ved entropikodning, da ville billedet kunne kodes ved så og så mange bit per pixel". Entropikodning kan naturligvis ikke altid foretages i praksis. Entropien er et kontinuert reelt mål, og vi kan kun tildele bitkoder med heltallig længde. Rimeligheden ved ovennævnte betragtning er, at det er op til transformationen og kvantificeringen at foretage komprimeringen. Selve indkodningen er jo reversibel, og uafhængig heraf.

Ved valget af indkodningsmetode er der flere hensyn at tage. Vi ønsker en metode hvor middelkodelængden vil ligge tæt på entropien. Dette krav er optimalt opfyldt for Huffman kodningen. Huffmankodningen er imidlertid beregningsmæssig noget tung. To andre indkodninger, der beregningsmæssigt er enklere, men ofte vil resultere i større middelkodelængder, er B-kodning og Skift-kodning. Disse metoder beskrives i det følgende. Der findes en lang række andre metoder til indkodning (f.eks. fibonacci-kodning, aritmetisk kodning mv.). Det vil føre for vidt her at beskrive disse metoder nærmere. Ved beskrivelserne vil vi benytte et eksempel med 6 mulige element-værdier v_i. Skemaet angiver de a priori sandsynligheder for observation af hver værdi:

element	w_1	w_2	w_3	w_4	w_5	w_6
sandsynlighed	0.4	0.3	0.1	0.1	0.06	0.04

Ved udregning ses at entropien af en sekvens med denne fordeling er $H \approx 2.14$.

4.10.1 Huffman kodning

Huffman indkodningen er bemærkesesværdig, idet den genererer en bitfølge hvis middellængde er mindst så god som enhver anden entydig indkodning. Indkodningen består af to faser. Først opbygges et træ hvor a priori hyppighederne for observartion af elementerne er noteret ved bladene. For hver knude i træet noteres summen af sandsynligheder noteret ved børnene til knuden. Træet opbygges iterativt fra bladene således at der introduceres en ny knude for hver iteration. Denne vælges således at sandsynlighederne noteret ved børnene er de mindste blandt de konstruerede knuder og tilbageblevne blade. I anden fase fordeles bitværdier ned gennem træet, således at hvis en knude har fået tildelt bitfølgen X, da vil knudens børn få tildelt følgerne $X0$ hhv. $X1$. For det tidligere (mindre illustrative) omtalte eksempel bliver træet:

blade	w_1	w_2	w_3	w_4	w_5	w_6
sandsynlighed	0.4	0.3	0.1	0.1	0.06	0.04
knudeniveau 1	0.4	0.3	0.1	0.1	0.1	
knudeniveau 2	0.4	0.3	0.1	0.2		
knudeniveau 3	0.4	0.3	0.3			
knudeniveau 4	0.4	0.6				

Som det ses er der tre knuder/blade i niveau 1, der har samme mindste (sum af) sandsynlighed(er). I sådanne tilfælde kan to af disse vælges vilkårligt, når den ny knude på næste niveau skal dannes. Forskellige valg vil resultere i forskellige koder, men middellængden af disse vil være ens. Ved anden fase er det let at indse, at kodetildelingen bliver:

element	w_1	w_2	w_3	w_4	w_5	w_6
$Huffmankode$	0	10	110	1110	11110	11111

Sekvensfølgen $w_3w_2w_5w_1$ vil således blive kodet ved: 11010111100. Vi ser at middelbitlængden ved indkodning af en sekvens med sandsynlighedsfordeling som i eksemplet bliver:

$$m = 0.4 * 1 + 0.3 * 2 + 0.1 * 3 + 0.1 * 4 + 0.06 * 5 + 0.04 * 5 = 2.20$$

Bemærk at denne værdi kun er lidt større end entropien. Værdien ξ er lig 1.03.

4.10.2 B-kodning

I B_n koder kodes en sekvens af værdier v_i ved en bitfølge hvor hver n'te bit er en såkladt *fortsættelsesbit*, der angiver om der skiftes til indkodning af et nyt element eller om indkodningen af elementet fortsætter over de næste n bit. For en B_1 kode rummer hver anden af bittene i indkodningen således egentlig information mens hver anden bit beskriver hvorledes bitfølgen skal fortolkes. Generelt indkodes hvert element ved et multipla af $n + 1$ bit. Tildelingen af bitkoder sker ved sortering af værdierne v_i efter hyppighed og (iøvrigt vikårlig) tildeling af koder således at de hyppigste elementer tildeles de korteste koder. For det benyttede eksempel kunne B_1 kodetildelingen være:

element	w_1	w_2	w_3	w_4	w_5	w_6
$B_1 kode$	$C0$	$C1$	$C0C0$	$C0C1$	$C1C0$	$C1C1$

hvor C angiver fortsættelsesbitten. Lader vi denne have bitværdien 1 for angivelse af skift til koden for et nyt element, vil bitsekvensen: 100011110110 entydigt kunne afkodes til sekvensfølgen $w_3w_2w_5w_1$. Det kan bemærkes at kodningsmetoden ikke er direkte, idet læsning af fortsættelsesbitten er nødvændig for at kunne afgøre om kodningen for elementer er afsluttet eller ej. Middelkodelængden for ovenstående indkodning vil være:

$$m \;=\; 0.4*2+0.3*2+0.1*4+0.1*4+0.06*4+0.04*4 \;=\; 2.60$$

Middelkodelængden er her noget større end entropien. Værdien af ξ er 1.21. Ved benyttelse af B_2 koder kunne kodetildelingen for ovennævnte eksempel være:

$element$	w_1	w_2	w_3	w_4	w_5	w_6
B_2kode	$C00$	$C01$	$C10$	$C11$	$C00C00$	$C00C01$

I dette tilfælde ville sekvensen $w_3w_2w_5w_1$ blive kodet ved bitfølgen: 110101100000100. Middelkodelængden vil være:

$$m \;=\; 0.4*3+0.3*3+0.1*3+0.1*3+0.06*6+0.04*6 \;=\; 3.30$$

I dette tilfælde er ξ lig 1.54. B_2 kodningen er altså ringere end B_1 kodningen i det aktuelle tilfælde.

4.10.3 Skift-kodning

Skift-koder er særdeles enkle og lette. Ved skiftkoden S_n opdeles kodningen af hvert element i blokke af n bit. Hver blok rummer altså mulighed for 2^n værdier. Alle disse bortset fra een, benyttes til kodning af information, medens den sidste benyttes til angivelse af, at kodningen af elementet strækker sig over endnu en blok. Tildelingen af bitkoder foretages som for B-kodningen. For det tidligere nævnte eksempel fås S_2-kodningen:

$element$	w_1	w_2	w_3	w_4	w_5	w_6
S_2kode	00	01	10	1100	1101	1110

Ved udregning af middelkodelængden for denne indkodning fås $m = 2.40$. Værdien af ξ er her lig 1.12. Denne indkodning er således næsten så god som såvel Huffmankodningen og som entropikodningen. Generelt gælder, som for B-indkodningen, at skift-kodning er forholdsvis effektiv for elementfølger hvis "sandsynlighedsfordeling" aftager hurtigt.

4.11 Prediktiv kodning

Ideen bag prediktiv kodning (prediktiv differental kodning) er meget simpel. Punkterne i billedet gennemløbes i en (på forhånd) fastlagt rækkefølge. For hvert nyt punkt predikteres værdien i punktet ud fra værdierne i de tidligere besøgte punkter. Metoden er således kausal. Transformationen består i en bestemmelse af den løbende prediktionsfejl:

$$e(x,y) \;=\; I(x,y) - P(x,y) \tag{4.56}$$

Hvis prediktionen er fuldstændig korrekt vil prediktionsfejlen være konstant lig nul. I dette tilfælde kan fejlen (usmart) kodes ved kun 1 bit per pixels svarende til en kompressionfaktor på 8, eller smartere ved kun 1 bit ialt. I praksis er prediktionsfejlen typisk eksponentialfordelt, med middelværdi 0, og med lille varians. Entropien af e er derfor mindre end entropien af I selv.

Den simpleste prediktive kodning er givet ved $P(x, y) = I(x, y - 1)$. Her svarer prediktionen til en antagelse om konstant intensitet langs y-aksen. Prediktionen siges at være af 1. orden. En simpel 3. ordens prediktion er givet ved:

$$P(x, y) \;=\; I(x-1, y-1) - I(x-1, y) - I(x, y-1) \tag{4.57}$$

Denne prediktion svarer til antagelsen om at intensitetsfladen er lokalt plan. Erfaringen viser at prediktorer af 1. orden er for lokale i den forstand at variansen af prediktionsfejlen ofte er stor. Dette vil medføre en ringe kompression. Prediktorer af orden højere end 3 vil være langsommere, og vil kun sjældent give væsentlig bedre resultater end en 3. ordens prediktor. Det kan bemærkes at for højere ordens prediktorer er det ikke oplagt hvilke koordinatsæt, der skal indgå i prediktionen. Det er dog sædvane, som illustreret ovenfor, at benytte de umiddelbare naboer til det aktuelle punkt. Under denne antagelse er den generelle 3. ordens prediktor er bestemt ved:

$$P(x, y) \;=\; aI(x-1, y-1) + bI(x-1, y) + cI(x, y-1) \tag{4.58}$$

hvor a, b, og c er reelle konstanter. Givet et billede er det muligt at vælge disse således at prediktionen bliver optimal. Optimalitetskriteriet er at den forventede kvadratmiddelfejl:

$$e^2 \;=\; E\{[I(x, y) - P(x, y)]^2\} \tag{4.59}$$

minimeres. Ved indsættelse af (4.59) og udregning ses at der fremkommer et udtryk, hvori der forekommer faktorer af formen:

$$R(\alpha, \beta) \;=\; E\{I(x, y)I(x-\alpha, y-\beta)\} \tag{4.60}$$

idet vi udnytter identiteter som f.eks.

$$E\{I(x-1, y)I(x-1, y)\} \;=\; E\{I(x, y)I(x, y)\} \tag{4.61}$$

Funktionen $R(\alpha, \beta)$ kaldes *autokorrelationsfunktionen* og angiver et mål for den stokastiske sammenhæng mellem en observeret intensitet og intensiteten i punktet forskudt (α, β). I praksis vil $R(\alpha, \beta)$ være stor for små værdier af α og β, og vil aftage meget hutrigt mod nul når disse værdier vokser. Udnyttes ovenstående definitioner kan vi altså skrive:

$$\begin{aligned} e^2 \;=\; & R(0,0) + a^2R(0,0) + b^2R(0,0) + c^2R(0,0) \\ & -2aR(1,1) - 2bR(1,0) - 2cR(0,1) \\ & +2abR(0,1) + 2acR(1,0) + 2bcR(1,-1) \end{aligned} \tag{4.62}$$

Vi ønsker at bestemme a, b, og c således at e^2 minimeres. Ved på sædvanlig måde at differentiere og sætte resultatet lig nul fremkommer ligningssystemet:

$$\begin{bmatrix} R(0,0) & R(0,1) & R(1,0) \\ R(0,1) & R(0,0) & R(1,-1) \\ R(1,0) & R(1,-1) & R(0,0) \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} R(1,1) \\ R(1,0) \\ R(0,1) \end{bmatrix} \tag{4.63}$$

Ved på forhånd at måle de fire autokorrelationsværdier kan vi let løse dette system. En simplifikation af systemet fremkommer hvis vi antager at $R(\alpha, \beta)$ er seperabel:

$$R(\alpha, \beta) = R_h(\alpha) R_v(\beta) \tag{4.64}$$

Det overlades til læseren at opstille udtryk for de tre koefficienter a, b, og c i dette tilfælde.

Som nævnt er transformationen givet ved prediktionsfejlen. Prediktiv differentialkodning er (nødvendigvis) reversivel. Hvis vi foretog nogen form for kvantificering ville der før eller siden opstå en rekonstruktionsfejl. Denne ville forplante sig videre gennem resten af billedrekonstruktionen. Af denne grund er metoden uegnet til kodning af billeder der skal transmiteres gennem en støjfyldt kanal. Derimod vil det være oplagt at benytte en variabel længde bitindkodning, da hyppigheden af prediktionsfejlen e må forventes at aftage meget hurtigt som funktion af fejlens numeriske størrelse.

Eksperimenter indikerer at kompressionfaktorer mellem 2 og 4 som regel er det bedste, der kan opnås ved prediktiv kodning. Dette afhænger naturligvis af billedet.

4.12 Run-længde kodning

I den endimensionelle run-længde kodning transformeres en sekvens af tal til tupler (I, l), hvor I angiver en talværdi (intensitet) og l angiver hvor mange efterfølgende tal i sekvensen, der har samme værdi. Indeholder sekvensen kun tal med samme værdi kodes denne således ved kun een tupel. Run-længde kodning er velegnet til billeder med relativt store områder med samme intensitet. Bemærk at hvis billedet indeholder en regelmæssig vertikal eller horisontal struktur, vil det være afgørende om billedet gennemløbes rækkevis eller søjlevis. Det kan også bemærkes at fejl i kodningen (f.eks. opstået under transmition) vil bringe en rekonstruktion "ud af takt", således at fejlen vil spredes til resten af rekonstruktionen.

For billeder med mange gråtoner og med moderat til stor detaljerigdom vil antallet af "runs" være stort. I sådanne situationer opnås ringe kompression. Generelt aftaget hyppigheden af antals "runs" med run-længden. Oplagt er derfor at kode længdeangivelserne ved variabel længde bitkoder. I stedet for at registrere intensitetsværdierne for hver run kan man i stedet registrere startintensiteten (f.eks. for hver skanlinie), og derefter benytte en simpel prediktiv kodning for de øvrige intensitetsværdier. I dette tilfælde bliver det også attraktivt at indkode prediktionsfejlen ved bitkoder af variabel længde.

Der kan broderes meget på run-længde kodningen for at øge metoden kompressionsegenskaber. I praksis er disse dog ringe (uanset de benyttede raffinementer) med mindre billedet kun indeholder få gråtoner og større enstonede områder.

Den endimensionale run-længde kodning udnytter kun den struktur i billedet der findes inden for hver skanlinie, ikke strukturen imellem skanlinierne. Det er muligt at udvide run-længde kodningen til 2 dimensioner ved (efter behandling af den første skanlinie) at sammenkæde "runs" over skanlinier. Praktisk kan dette foretages ved en algoritme, der ligner den simple segmenteringsalgoritme fra afsnit 3.2. Resultatet af et segment kan kodes ved følgen:

$$(I, (x_0, y_0), (\Delta_1^1, \Delta_1^2), (\Delta_2^1, \Delta_2^2), \ldots, (\Delta_n^1, \Delta_n^2)) \tag{4.65}$$

hvor (x_0, y_0) angiver koordinatværdierne for startpunktet i den første skanlinie for segmentet. Δ_i^1 angiver forskydningen af startpunktet for intervallet i den i'te skanlinie i forhold til startpunktet for intervallet i den (i-1)'te skanlinie. Tilsvarende angiver Δ_i^2 ændringen i runlængden. Der findes flere algoritmer til bestemmelse af sådanne todimensionale "runs" i et billede. Da de alle lider af samme svagheder som metoderne for det endimensionelle tilfælde skal vi ikke diskutere disse metoder yderligere.

4.13 Kodning ved blok-trunkering

Ideen bag bloktrunkering er at foretage en lokal binarisering af billedet. Som ved alle blokbaserede metoder er problemet, at hvis der benyttes en stor blokstørrelse vil rekonstruktionen ofte være domineret af en "skakbords effekt". Dette er i særdeleshed tilfældet ved bloktrunkering, hvor kun to værdier bevares for hver blok. I praksis er blokstørrelsen derfor 4×4 pixel. For hver blok i billedet måles middelværdien m og spredningen σ af intensitetsværdierne i blokken. Blokken binariseres derefter, således at intensitetsværdier større end m tildeles bitværdien 1, og resten værdien 0.

Ved rekonstruktionen af det kodede billede ønsket momenterne af første og anden orden af intensiteterne i hver blok bevaret. Lad $N = 16$ være antal pixel i blokken, og lad k være antal pixels med kodet bitværdi lig 1. Lad A hhv. B være de to rekonstruerede intensitetsværdier. Kravet om bevaring af de to momenter kan da udtrykkes ved:

$$\begin{aligned} m &= \frac{(N-k)A + kB}{N} \\ \sigma^2 + m^2 &= \frac{(N-k)A^2 + kB^2}{N} \end{aligned} \tag{4.66}$$

Ved at løse ligningerne for A og B (en andengradsligning hvoraf den ene løsning ikke kan bruges, da vi ved at $A < B$) får vi:

$$\begin{aligned} A &= m - \sigma\sqrt{\frac{k}{N-k}} \\ B &= m + \sigma\sqrt{\frac{N-k}{k}} \end{aligned} \tag{4.67}$$

I praksis bestemmes de to gråtoneværdier, der ligger tættest på de fundne (reelle) værdier af A og B. Hver blok kodes således ved $N + 2 * 8 = 32$ bit, svarende til en konstant kompressionsfaktor på 4. En mulig videreudvikling af metoden kunne være at foretage yderligere kodning af A- og B-billederne, f.eks. ved prediktiv kodning, eller ved kvantificering af disse koefficientbilleder. En anden mulighed er kun at registrere A-værdien hvis spredningen af intensiteten i en blok er meget lille. Eksperimenter har bekræftet at metoden kan forfines åledes at gode rekonstruktioner er opnåelige ved kompressionsfaktorer omkring 20 (se *T. Frandsen: Kodning af billeder ved blok-trunkering, DIKU-projekt 90-6-20*).

Metoden, der både er let og hurtig, virker i de fleste tilfælde fortrinligt. Især hvis billederne har lav kontrast kan en efterbehandling ved lavpassfiltrering genskabe det oprindelige billede godt. En ulempe ved metoden er den relativt lave kompressionsfaktor.

4.14 Pyramidekodning

To (ofte refererede) eksempler på kodninger, der tager udgangspunkt i en pyramiderepræsentation er kodning af Laplacepyramiden og kodning af Waveletpyramiden. Ideen bag disse kodninger er, at de forskellige lag i pyramiderne indeholder forskelligartet information og derfor kan kvantificeres forskelligt. Således kan det nederste niveau i laplacepyramiden som regel kvantificeres til kun 3 værdier, mens højere niveauer kræver flere kvanter. Kodning ved Laplacepyramiden er oprindeligt beskrevet i *P. J. Burt, E. H. Adelson: The Laplacian Pyramid as a Compact Image Code, IEEE trans. on Communications vol. 31 no. 4 april 1983, pp.532-540.* Denne metode tillader typisk en kompressionsfaktor mellem 4 og 16 svarende til mellem 0.5 og 2.0 bits/pixel. Kodning på basis af Waveletrepræsentationen er f.eks. beskrevet i: *Mallat: Multiresolution Representations and Wavelets; PhD-thesis, Univ. of Pennsylvania USA, 1988.* Denne metode er overlegen i forhold til kodning af Laplacepyramiden, primært fordi waveletrepræsentationen er ortogonal, men også fordi metoden tilsyneladende bedre er i stand til at pakke mere information med i de mindre (og derfor "billigere") billeder i pyramiden.

4.14.1 Kodning af Laplacepyramiden for et billede

En metode *P. J. Burt and E. H. Adelson: The Laplacian Pyramid as a Compact Image Code; IEEE trans. on communications vol. 31 no. 4 1983* beskriver hvorledes et billede kan komprimmeres effektivt på basis af en laplacepyramidereræpræsentation af billedet. Som beskrevet i afsnit 4.4.2 beregnes laplace-pyramiden ud fra en gauss-pyramide ved for hvert niveau i i gauss-pyramiden at subtrahere det ekspanderede billede svarende til niveau $i+1$. Det højeste niveau i laplace-pyramiden er identisk med det tilsvarende billede i gauss-pyramiden. Burt og Adelson benytter approksimationen til gaussfilteret som den er beskrevet i afsnit 4.4.2. Filtret har en støtte på 5×5 pixels, er seperabelt, symmetrisk, og parametriseret ved værdien a. Burt og Adelson har ikke brugt værdien $a = 0.4$, der giver den bedste approksimation til gaussfilteret, men har fundet værdien $a = 0.6$ bedre mht. kodningsanvendelsen.

Det er klart, at det laveste niveau i laplace-pyramiden beskriver de høje frekvenser i billedet, og at lavere frekvenser er repæsenteret højere oppe i pyramiden. Dog skal det bemærkes at frekvensopdelingen ikke er ideel. Der er således nogen korrelation mellem billederne på forskellige niveauer i pyramiden. Ved undersøgelse af histogrammet over værdierne på de forskellige niveauer i laplace-pyramiden kan man iagttage at disse er meget koncentreret omkring 0 på laveste niveau, mens fordelingen har gradvist større spredning for voksende niveaunummer. På et velkendt billede (lenna), der ved sædvanlig repræsentation har en spredning af intensitetsværdierne på omkring 42 og en entropi på 7.2, er de interessante tal opsummeret i nedenstående skema.

niveau	0	1	2	3
spredning	8.4	11.5	16.0	21.0
entropi før kvantificering	4.5	5.0	5.6	6.2
entropi efter kvantificering	0.8	1.0	3.4	4.7
bidrag til entropi-kodning	0.8	0.25	0.21	0.07

Tabel 4.3: Spredning og entropi af værdierne på en 4-niveau Laplacepyramide repræsentation af et billede (lenna). Tallene er opnået for en værdi a $a = 0.6$

Den grundlæggende idé ved laplace-pyramidekodningen er at kvantificere forskelligt på de forskellige niveauer i pyramiden. På det laveste niveau er det typisk tilstrækkeligt at benytte tre til fem kvanter. Med voksende niveaunummer (og spredning) må antallet af kvanter øges. Dette medfører naturligvis et voksende forbrug af bit, men da billederne her er mindre er udgiften tilsvarende lavere. Typisk benyttes en uniform kvantificering. Som det ses af ovenstående tabel er det især de laveste niveauer i pyramiden der bidrager til den samlede teoretiske antal bit per pixel (1.33 ved entropikodning). Det er ganske svært at vælge et optimalt sæt af kvanter. For 5 niveauer i pyramiden beskriver *P. Flagstad: Billedkompression ved Pyramidekodning: Laplacepyramiden og Wavelet-transformationen (DIKU-speciale)* at for mange billeder vil valg af antal kvanter lig 3, 5, 7, 11, og 17 give rimeligt gode resultater.

Det største "problem" ved laplacepyramidekodningen er, at selv om det laveste niveau i pyramiden kvantificeres til kun 3 værdier, vil antallet af bit der i praksis skal til at repræsentere dette niveau være relativt stort (større end 1 per pixel). Bevares det laveste niveau er det vanskeligt at opnå kompressionsfaktorer over ca. 5-7. Til gængæld er kvaliteten af rekonstruktionen ofte meget god. Det viser sig meget vanskeligt at kode det laveste niveau effektivt ved andre "smarte" metoder (f.eks. run-længde kodning). Det eneste alternativ er derfor at bortkaste informationen knyttet til det laveste niveau. Dette vil ofte give muligheder for kompressionsfaktorer over 20, men på bekostning af noget skarphed og detalierigdom i rekonstruktionen. En oplagt fordel ved pyramidekodningsmetoden er, at denne egner sig fortrinligt ved progressiv transmission af billeder.

4.15 Kodning ud fra kanter

Kodning af kanter er interessant, fordi en sådan vil kunne afklare i hvor høj grad kantinformationen alene er en god beskrivelse af et billede. En implementation af en sådan kodning er beskrevet i: *Carlsson: Sketch Based Coding of Grey Level Images; Signal Processing 15, 1988.* Den beskrevne metode lader meget lovende i den forstand at gode rekonstruktionen kan opnås for store kompressionsfaktorer, dvs. for faktorer mellem 20 og 60.

Metoden består i først at detektere kantpunkter i billedet, derefter at kode kanterne (en art chain-codes) samt for hver kant at kode udtryk for intensiteterne på hver side af kanten. På basis af de kodede mål rekonstrueres et approksimativt billede. Dette fremkommer ved tilpasning af en glat flade til de kodede mål (dvs. positionen af kanterne og intensiteterne på hver side af kanterne). Tilpasningen foretages ved regularisering, dvs. ved at bestemme det billede, der stemmer godt overens med målingerne og samtidig minimerer integralet af gradientstørrelsen.

$$\text{minimere:} \qquad \int\int_{\Omega} [f_x^2 + f_y^2] dxdy \; + \; \mu \sum_{i \in K} [f - c_i]^2 \tag{4.68}$$

Fejlen ved denne rekonstruktion (residualbilledet) kodes dernæst ved en Laplacepyramidekodningen beskrevet i afsnit 4.14.1. Ved denne del af kodningen benyttes kun de 1-3 øverste niveauer i laplacepyramiden.

Ved udeladelse af kodning af residualbilledet kan det udledes (Se *Carlsson: Sketch Based Coding of Grey Level Images; Signal Processing 15, 1988* at kompressionsfaktoren F er:

$$F \; = \; \frac{8N^2}{27N_c + 1.3L_c} \tag{4.69}$$

hvor N er billedkantlængden, N_c er det totale antal (akseparallelle) kontursegmenter, og L_c er det totale antal kantpunkter. I mange billeder vil man typisk finde at $L_c \approx \frac{N^2}{10}$, og at $N_c \approx \frac{L_c}{5}$. Det ses herved at for sådanne billeder vil $F \approx 12$. Dette illustrerer metodens svaghed, nemlig at en høj kompressionsfaktor kræver at der kun detekteres få kanter. Benyttes kun de mest markante kanter vil denne del af kodningen ofte give kompressionsfaktorer op mod 100. Dette er naturligvis på bekostning af kvaliteten af rekonstruktionen. Benyttes desuden kodning af residualbilledet (ved de øverste niveauer i laplacepyramiden) fås en acceptabel god rekonstruktion ofte ved kompressionsfaktorer mellen 20 og 40. Dette illustrerer styrken ved metoden. Kantkodningen fanger de globale strukturer i intensitetsbilledet og koder disse effektivt. De lokale variationer i intensiteten (teksturen) kodes derefter ved brug af laplacepyramiden.

4.16 Kodning ud fra segmenter

Et eksempel på kodning af segmenter er beskrevet i oversigtsartiklen: *M. Kunt, M. Bernard, R. Leonardi: Recent Results in High-Compression Image Coding, IEEE trans. on Circuits and Systems, vol. 34 no. 11, november 1987 pp 1306-1336.* Fundamentalt for denne metode er (naturligvis) segmenteringsalgoritmen. Metoden er en af de mest lovende af en række nyere metoder til billedkodning. Gode rekonstruktioner rapporteres for kompressionsfaktorer mellem 20 og 60.

Segmenteringsmetoden er af typen *split-and-merge*. Initielt detekteres kanterne i billedet. Kriteriet for at opdele et segment (under split fasen) er at det kvadratiske segment indeholder et eller flere kantpunkter. Største segmenter er 32 gange 32 pixels. Der opdeles ned til segmenter af størrelsen 4 gange 4 pixels. De segmenter, der derefter indeholder kantpunkter opdeles i ikke-kvadratiske segmenter efter kanten. Herefter tilpasses intensitetsfladen i alle segmenter med et polynomium. Der benyttes polynomier af op til tredie grad. Desuden konstrueres en nabograf.

I foreningsfasen sammenlignes approximationerne for intensiteten i nabosegmenter, og det afgøres om to nabosegmenter skal forenes. Kravet er at approksimationsfejlen (for foreningssegmentet) forbliver under en prædefineret grænseværdi, samt at foreningssegmentet ikke indeholder nogen "væsentlige" kanter. Segmenter forenes i rækkefølge givet ved den til enhver tid mindste appriksimationsfejl (for foreningssegmentet). Foreningsfasen består, mere detalieret, af tre delfaser. I første delfase forenes de store (pålidelige) segmenter. I anden delfase forenes de (ofte mange) små segmenter, med størrelse mindre end 4 gange 4 pixels, med større segmenter. Dette sker for at skabe overensstemmelse mellem segmentgrænserne og de fundne kanter. I tredie delfase forenes segmenter yderlige indtil et mindste antal segmenter er fundet eller indtil approximationsfejlen bliver for stor.

Kodningen består af en (art "chain-codes") kodning af segmentgrænserne samt af en kodning af polynomiumskoefficienterne for de tilpassede intensitetsflader for hvert segment. Det er klart at jo mindre polynomiegraden vælges jo færre bit kan de tilpassede polynomier kodes ved. Tilsvarende vil antallet af segmenter ofte være stort. Dette betyder at kodningen af segmentgrænserne bliver dominerende. Hvis tredie ordens polynomier benyttes vil der være relativt færre segmenter. Dette betyder at kodningen af koefficienterne vil være dominerende for kompressionsfaktoren. Det er generelt ikke let, selv med kendskab til billedet, at afgøre hvilken polynomiumsgrad, der vil give den bedste kodning. Det kan her bemærkes at formålet med tredie delfase af foreningsfasen er at opnå en forudbestemt omtrentlig kompressionsfaktor. Et væsenligt problem ved metoden er at denne kun i meget ringe omfang (og ved lav kompressionsfaktor) er i stand til at kode tekstur i billederne. På dette felt ligner metoden hvor segmenter kodes således metoden hvor kun kanter i billedet kodes (se foregående afsnit).

4.17 Kodning af billedsekvenser

Dette afsnit skrives ikke i år.

4.18 Opgaver

1. Skriv et program til opbygningen af en $log2$-pyramide med 2×2-støtte hvor frembringerfunktionen er givet ved middelværdien af de 4 støtteelementers værdi. Udnyt at billedbiblioteket stiller rutiner til ind -og udlæsning samt allokering af pyramider til rådighed.

2. Som opgave 1, blot er frembringerfunktionen her en gaussfunktion (overvej selv hvor stor støtten bør være).

3. Skriv et program til opbygning af en Laplace-pyramide, hvor frembringerfunktionen er valgt som i Burt og Adelsons metode (se afsnit 4.4.2). Undersøg (den visuelle) effekt ved variation af parameteren a.

4. Sammenlign pyramiderne genereret i opgave 1 og opgave 2 for et billede. I hvilke situationer adskilder de to repræsentationer sig væsentlig fra hinanden.

5. Skriv et program til en simpel billedkodning, hvor billedet først segmenteres vha. algoritmen beskrevet i afsnit 3.2, og middelintensiteten for hvert segment registreres. Kodningen foretages nu skanlinie for skanlinie, sådan at fællesmængden mellem linien og hvert segment kodes ved (antal pixels, middelintensitet). Mål kompressionsfaktoren for et sæt af billeder.

6. Beskriv den optimale kvantificering med K kvanter for en koefficient $x \in [0, 1]$, med tæthedsfunktion $p(x)$ givet ved $p(x) = 2(1 - x)$ for $0 < x < 1$, og ellers 0. Antag at $z_0 = 0$ og $z_K = 1$. Angiv kvantificeringsværdierne for K lig 2, 3, 5, og 11. Advarsel: Denne opgave er større end den ser ud til.

7. Lad w_i være en følge af elementer med 12 mulige værdier, hvis hyppighed er angivet i tabellen.

w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	w_{10}	w_{11}	w_{12}
0.01	0.05	0.12	0.09	0.20	0.13	0.12	0.08	0.04	0.11	0.03	0.02

Bestem entropien H. Angiv indkodningen af elementerne ved Huffmankodning, ved B_1- og B_2-kodning, samt ved S_2- og S_3-kodning. Angiv middelkodelængden for hver kodning.

8. Som i ovenstående opgave blot med følgende hyppigheder:

w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	w_{10}	w_{11}	w_{12}
0.01	0.01	0.02	0.03	0.20	0.43	0.12	0.08	0.04	0.01	0.03	0.02

9. Bestem eksperimentelt den typiske kompressionfaktor ved anvendelse af UNIX-komandoen "Compress". Benyt ca. 10 billeder med varierende motivtype.

10. Skriv et program til beregning af entropien i et billede.

11. Skriv et program til optimal 3. ordens prediktiv kodning. Mål den typiske reduktion i entropien ved transformationen.

12. Skriv et program til endimensional run-længde kodning. Mål den typiske kompressionsfaktor.

13. Skriv et program til bloktrunkering og efterfølgende rekonstruktion. Beskriv reduktionen i billedkvalitet for et antal billeder.

14. Betragt blok-kvantificeringen af koefficienterne y_i til en transformation af et 4 gange 4 delbillede. Antag at koefficienterne er sorteret efter aftagende varians, og at dennne approksimativt er givet ved $\sigma^2(y_i) \approx ke^{-x}$. Antag at vi bortsmider halvdelen af koefficienterne (de med mindst varians), og antag at den af de tilbageblevne koefficienter, der har mindst varians kodes ved 2 bit. Benyt en blokkvantificering hver antallet af kvanter for y_i vælges proportionalt med $\log(\sigma_i^2)$. Hvor mange bit kodes blokken ialt ved? Hvor stor bliver kompressionsfaktoren?

15. Skriv et program til reduktion af et 24-bits RGB-farvebillede til et 8-bits farvebillede. Sammenlign kvaliteten af din reduktion med den der foretage i programmet *imxv*.

16. Skriv et program til interaktiv farvelægning af et billede. Bemærk at opgaven er meget omfattende.

17. Skriv et program til bokkodning vha. cosinustransformationen. Efter at alle blokke er transformeret beregnes variansen for hver koefficient. Sorter varianserne og udskriv en liste (evt. et plot) over koefficientnummer og varians. Prøv programmet på et antal billeder og diskuter om kurveformen er ens, og om koefficientrækkefølgen er ens.

18. Udvid programmet fra opgave 17 med en egentlig (evt. adaptiv) kodning og mål kompressionsfaktoren for et sæt af billeder. Udskriv samtidig det dekodede billede og vurder hvor god rekonstruktionen er.

19. Benyt visualiseringsprogrammet *imxv* til at betragte en 24-bits farvebillede (se i kataloget: /usr/local/image/archive/color/). Bemærk at *imxv* konverterer til 8-bits farve. Benyt de forskellige konverteringer. Kan du se nogen forskel? Benyt vinduet "coloredit" til at modificere farverne. Brug både RGB-systemet og HSI-systemet.