Innholdsfortegnelse

Objektplan – bildeplan

https://www.uio.no/studier/emner/matnat/ifi/IN2070/v21/undervisningsmateriale/forelesning/2021_02_sampling_6pp.pdf

Romlig oppløsning sier noe om graden av fine detaljer som er representert i et bilde. Det er dog ikke det samme som pikselering. Det oppgis ofte som hvor langt fra hverandre to punkter må være for å kunne skille dem fra hverandre i bildet.

$$y' = \frac{yf}{s-f} \qquad \sin\theta \approx \tan\theta \approx \theta$$

f = 35 mm og D = 10 mm (Tilnærmet vanlig kamera)
s = 5 m (Avstanden til det som avbildes)
$\lambda = 500 \cdot 10^{-9}$ m (Grønt lys)

$\tan\theta \approx \sin\theta = 1.22\,\lambda / D = 6.1 \cdot 10^{-5}$ (Rayleigh)

$y = \tan\theta \cdot s \approx 3.05 \cdot 10^{-4}$m ≈ **0.3 mm** (I objektplanet)

y' = 0.3mm · 35 / (5000-35) ≈ **2.1 µm** (I bildeplanet)

Rayleight-kriteriet sier hvor langt fra hverandre to punkter må være for å kunne skille dem fra hverandre. To punkter kan skilles fra hverandre i bilde hvis vinkelen mellom dem tilfredstiller $\sin(\theta) = 1.22\lambda/D$ (radianer).

Samplingsteoremet (Shannon/Nyquist) sier at det kontinuerlige bildet kan rekonstrueres fra det digitale bilde dersom samplingsraten $f_s=1/t$ er større enn $2f_{max}$ ($2*f_{max}$ = Nyquist-raten). Er samplingen mindre enn dette får du aliasing. Da vil det digitale bilde ikke inneholde de samme frekvensene og detaljene som det kontinuerlige bildet (virkeligheten). Fine detaljer kan fremstå helt annerledes. Antialiasing er å redusere den romlige oppløsningen til bilde før samplingen s.a. samplingsraten vil være bra nok til bilde.

Kvantifisering er å velge nivåene for styrken til det digitale bilde fra det kontinuerlige bilde(ekte). pikslene kan inneholde heltalsverdier fra 0 til 2^n-1 (n=antall bits)

Geometriske operasjoner

https://www.uio.no/studier/emner/matnat/ifi/IN2070/v21/undervisningsmateriale/forelesning/in2070-2021-geometriskeoperasjoner.pdf

En geometrisk operasjon er å endre posisjonen til pikselen. Ofte brukes affine transformer som tar inn en pikselposisjon og returnerer den nye posisjonen din.
De affine transformasjonene beskrives ved
$x' = a_0x + a_1y + a_2$
$y' = b_0x + b_1y + a_2$
Rette linjer bevares
Parallelle linjer forblir parallelle
lineær transformasjon + translasjon
kan utrykkes ved matrisemultiplikasjon

Mapping
Forlengs-mapping transformerer piksler fra inn til ut-bildet. Da vil det som regel bli noen verdier som ikke har blitt mappet til.
Baklengs-mapping finner piklsene i utbildet ved å kjøre transformasjon for å finne ut hvilken piksel fra inn-bildet som passer
Når vi bruker baklengs-mapping får vi ofte en verdi som ikke er et heltall. Da må vi anta hvilken piksel vi skal bruke fra inn-bilde. Da brukes interpolasjon.

Interpolasjon
Nærmeste nabo interpolasjon tar en enkel avrunding av koordinatene.
- Gir taggete kanter
- Hver piksel bruker en verdi fra inn-bilde.

Bilineær interpolasjon buker de fire pikslene rundt der transformasjonen «landet» for å approksimere hvilken pikselverdi det punktet skal ha.
- kontinuerlige resultat
- Noe mer regnekrevende

Høyere ordens interpolasjon bruker kontinuerlig derivasjon av høyere orden
- kontinuerlig derivert av ønsket orden
- mye mer regnekrevede
- kan gi kantglorie-effekt

Samregistrering
Samregistrering brukes til å transformere et bilde til gitte koordinater. Om du har noen koordinater i inn-bildet og ønskede koordinater til samme sted i ut-bildet kan du kjøre en transformasjon som gjør at punktene du har valgt ut ligger der du vil i ut-bildet.

Gråtonemapping
https://www.uio.no/studier/emner/matnat/ifi/IN2070/v21/undervisningsmateriale/forelesning/2021_04_gratonemapping.pdf

Kontrast er forskjellen i frage/lysstyrke eller andre fysiske egenskaper på et objekt.
Et spredt histogram gir høy kontrast.
Gråtonetransformer kan brukes på bilde samt på histogrammet for å øke bla. kontrast eller lyshet.
I lineær kontrastendring brukes transformasjonen T[i] = ai + b.
Her øker a kontrasten hvis a > 1 og b flytter alle gråtoner b nivåer
a = -1 gir negativer
Kommer noen verdier utenfor intervallet [0, 255] blir de enten klippet bort eller satt til 255/0

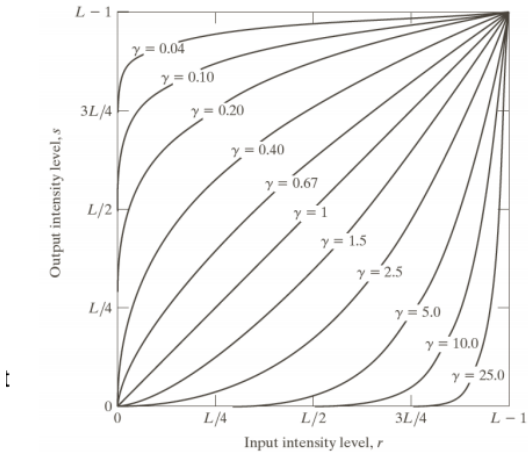
Standardisering
Standardisering av bilder brukes for å få flere bilder til å ha lik varians og middelverdi. kan brukes for å sammenligne bilder.
Man må huske på å velge et standardavvik som gjør at minst mulig piksler blir klippet

Ikke lineær transform
Endrer f.eks. lyshet i mørke deler av bildet mer enn i lyse.

- logaritmisk T[i] = log(i)
- eksponentiell T[i] = e^i
- gamma-skalering T[i] = i^γ

$\gamma < 1$: Den mørke delen av bildet blir lysere
$\gamma = 1$: identitets-transform
$\gamma > 1$: Den lyse delen av bildet blir mørkere

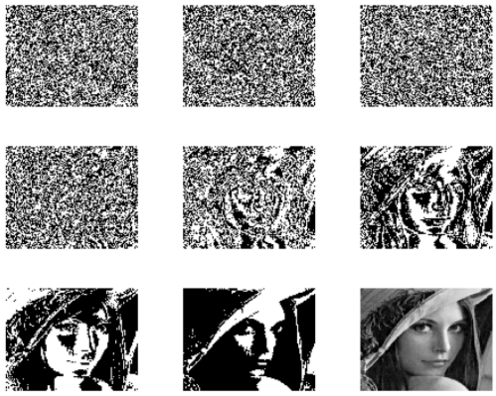
(Fig 3.6 i DIP)



- stykkevis lineær skalering

Terskling er å sette alle verdiene større enn en gitt grense til 1 og alle verdiene mindre enn grensen til 0. Det gir et binært bilde.

Bit-plan-oppdeling



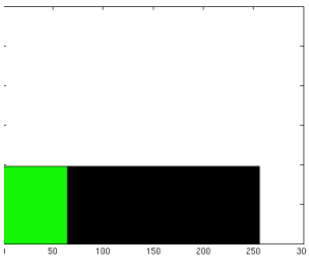
Vise frem bildene de forskjellige bitsene gir. Man ser at kun de fire første bildene har visuell signifikans.
Kan benyttes i kompresjon

LUT

(Siste bilde er inn-bildet)

Histogrambaserte operasjoner

https://www.uio.no/studier/emner/matnat/ifi/IN2070/v21/undervisningsmateriale/forelesning/2021_05_histogramtransformer.pdf



For et bra bilde vil vi generelt ha et ganske jevnt histogram med store mellomrom mellom høye søyler og små mellomrom mellom små søyler. Dette gir **kumulative histogram** oss mulighet til. Histogramutjevning kan gi oss tilnærmet flatt histogram.

Det kan være hensiktsmessig å standardisere histogrammer til en rekke bilder. Dette slik at det fjerner effekt av bla. døgnvariasjon i belysning, aldrigseffekt i lamper og detektor,akkumulering av støv på linser osv.. og brukes i bla. produkt-inspeksjon i industri, ansiktsgjenkjenning, medisinsk avbilding osv.
Dette bør ikke gjøres om formen på histogrammet har verdi ved videre analyse, og det kan være smart å bruke lineære transformasjoner slik at strukturene i histogrammet bevares.

Lokal gråtonetransform

Brukes til å endre lyshet eller kontrast i deler av bildet. Kan være nyttig om deler av bilde har annen belysning. Det gjøres ved å velge ut delen som skal transformeres og regne ut et nytt (kumulativt) histogram for den delen for å så utføre for den delen.
Det kan være regnekrevende å lage mange histogram og transformer.

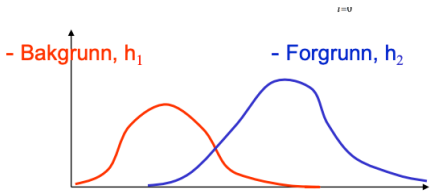
Segmentering ved terskling

https://www.uio.no/studier/emner/matnat/ifi/IN2070/v21/undervisningsmateriale/forelesning/2021_06_segmentering.pdf

Segmentering er en prosess som deler bilder i meningsfulle regioner. Vi bruker ofte segmentering for å dele et bile inn i forgrunn og bakgrunn. Ved terskling fremhever vi pikslene som ligner hverandre. Segmentering brukes også til å få frem kanter.

Terskling

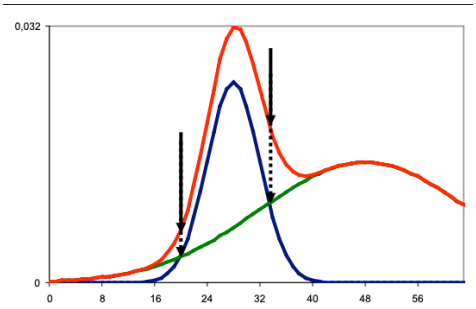
Om vi vet at forgrunnen er mye lysere eller mørker enn bakgrunnen kan vi bruke terksling for å dele bildet og laget et binært bilde. Vi kan da klassifisere piksler ved hjelp av intensitet alene.



Vi ønsker her å klassifisere flest mulige piksler som forgrunn eller bakgrunn. Pikslene under begge grafene er piksler som vil bli feilkategorisert. Denne mengden vil vi minimere.

Vi må da altså finne to modellhistogram som passer til histogrammet ved å f.eks. anta to Gauss-fordelinger og tilpasse de.

Når vi så har funnet disse to histogrammene må vi finne grensen som gir minst mulig tap. Det er når grensen i det laveste punktet, der grafene krysses (Det trenger ikke alltid å være akkurat det stedet!) Det kan man finne ved å sette verdien av de to grafene lik hverandre, evnt. derivert = 0.



Vi kan også i noen tilfeller få to verdier der grafene treffer hverandre.

Ridler og Calvards metode:

Om $\mu_1 \approx \mu_2$ og $\sigma_1 \approx \sigma_2$ vil vi kunne finne en T(terskling) ved å gjette to middelverdier og sette $T = (\mu_1 + \mu_2)/2$, deretter sette nye μ_1 og μ_2 som henholdsvis middelverdien til pikslene under og over T. Gjennta dette til T konvergerer og du har funnet en bra terksling.

Otsus metode:

Denne metoden handler om å finne det punktet som har størst varians mellom middelverdiene i forgrunn og bakgrunn.

Begge disse metodene krever at sannsynligheten for forgrunn er ganske lik som bakgrunn. Feilen øker ganske fort når sannsynligheten blir forskjellig.

For å terskle ved a priori sannsynligheter som er ulike kan man kun bruke piksler som ligger på eller nær overgangen mellom objekt og bakgrunn, altså ved kanter.

Terskling med støy

Om det er mye støy i bilde kan det føre til histogrammer uten klare middelverditopper. Da kan det være lurt å utføre støyfjerning først.

I mange bilder vil lysheten variere, og da må man ofte bruke flere tersklingsverdier for å kunne segmentere.

Filtrering i bildedomenet/Naboskapsoperasjoner

https://www.uio.no/studier/emner/matnat/ifi/IN2070/v21/undervisningsmateriale/forelesning/in2070_2021_07_filtrering_i.pdf

Filtrering er et verktøy som brukes i bildeforbedring og analyse til å bl.a. fjerne76redustere støy, forbedre skaprhet eller dektere kanter, gjerne gjennom Konvolusjon.
HUSK! Roter filteret 180 grader.
Konvolusjon er både

Kommutativ(f*g = g*f)
Assosiativ (f*g)*h = f*(g*h)
Distrubtiv f*(g´h) = (f*g) + (f*h)
Assosiativ ved skalar multiplikasjon a(f*g) = (a*f)*g = f(a*g)

Ved en konvolusjon vil normalt ut-bilde bli floor(filter_size/2) større på hver side enn inn-bildet.

Bilderandsproblem

Man kan ente bli kvitt hele problemet ved å kun beregne responser for der origo er innenfor inn-bildets dimensjoner.
Hvis man ikke gjør dette og ut-bildet blir større kan man enten nullutvide, nærmestenabo eller bildets gjennomsnittverdi på randen til inn-bildet.
En annen variant, men mindre vanlig er å sette faste verdier til ut-bilde. f.eks g(x,y) = 0, og da ignorere posisjoner der filteret ikke overlapper med inn-bildet.

Andre praktiske problemer ved filtrering kan være:
Får ut-bildet samme kvantifisering som inn-bildet?
Kan vi direkte endre inn-bildet, eller må vi mellomlagre resultatbildet.

Korrelasjon

Samme som konvolusjon bare uten å rotere filteret.
Kan brukes i mønstergjenkjenning, eller til å finne bestemte objekter i bildet.
Korrelasjonskoeffisient

Om filteret kun er 1 piksel er det en gråtonetransform. Ny pikselverdi avhenger bare av den gamle pikselverdien.

Filtertyper

Lavpassfilter: Slipper gjennom lave frekvenser og demper/fjerner høye. Glatter/blurrer ut bildet. Fjerner støy. Kan ødelegge kanter.
Middelverdifilter (lavpass): Glatter ut. Alle verdiene i filteret er like og summer seg til 1.
Større filter gir større utglatting
Gauss-filter(lavpass)

Kantbevarende lavpassfilter

Man vil ofte ønske å bevare kantene under filtrering, dette gjør ikke alle lavpass-filtrering så godt. Men man kan bruke f.eks. Rang-filtrering. En type Rang-filtrering er median-filter(lavpass). Den velger medianen av naboskapet som verdi.
Uleper er
Tynne linjer kan forsvinne, hjørner kan rundes av og objekter kan bli litt mindre.

Se sammenligning av middelverdi og medianfilter på slide 46.

Man kan separere et 2D-filter til to 1D-filtre.
Konvolusjon ved slike filter finnes på slide 38

Nærmeste-nabo-filter side 52.

MinimalMeanSquareError(MMSE) side 54

Filtrering i bildedomenet II / Naboskapsrelasjoner II

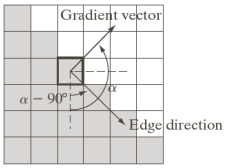
Høypassfilter

slipper gjennom høye frekvenser og demper eller fjerner lave frekvenser
Frehever skarpe kanter, linjer og detalje. Demper langsomme variasjoner f.eks. ujevn bakgrunn.
Øker skarphet, men kan øke støy.
Et høypassfilter ser typisk slik ut:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Høypassfiltre kan brukes i punkt-deteksjon.
Høypass = Original-lavpass

Digital derivasjon



En kant detekteres ved endring i intensitetsverdi.
Gradienten peker i retningen der funksjonen øker mest.
Kanten vil da gå vinkelrett på gradienten
For å regne gradienten til et digitalt bilde bruker man to konvolusjonsfiltre som tilnærmer hver sin gradient-komponent.
h_x tilnærmer partiell-derivert i x-retning ved å beregne differanse i vertikal retning.
se slides fra side 23 for forskjellige gradient-operatorer.

Lager man større filtre (gradient-operatorer) kan de bli mer støy-robuste, men det blir mye mer regnekrevnde. For å gjøre det litt lettere kan man dele opp 2D-filteret til 2 1D filtre.

Laplace-operator

For å gjøre gradient-operatorn mer robust mot støy kan vi bruke en 3x3 laplace-operator. Den vil finne to eksremverdier per kant, på starten og slutten av kanten. Kantens eksakte posisjon er null-gjennomgangen. En full 3x3-laplace operator ser slik ut.

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

LoG-funksjon

Cannys algoritme

oppskrift slide 59

Fourier-transformasjon

https://www.uio.no/studier/emner/matnat/ifi/IN2070/v21/undervisningsmateriale/forelesning/in2070-2021-fourier1.pdf

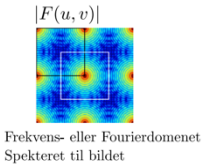
Et gråtonebilde kan representeres både som en matrise av gråntoneintensiteter eller ved en vektet sum av sinuser og cosinuser med ulik frekvens og orientering. Et skifte mellom disse kalles et basis-skifte. Det kan brukes ved f.eks. analyse, fjerne støy, kompresjon osv.
Enhver sinus-funksjon kan dannes ved å ved å legge sammen en vektet sinus-funksjon og en vekte cosinus-funksjon med samme frekvens.

Basisbilder er matriser med én 1-er og resten 0. n^2 antall slike blir en NxN-matrise.

Digitale gråtonebilder av størrelse MxN kan representeres ved en vektet sum av MxN sin- og cos-bilder. Ved hjel av cos- og sin-bidragene fra et slikt bilde kan vi finne fasen og amplituden. Dette kan fortelle oss viktig informasjon om bildet. Resultatet fra dette blir ofte representert som komplette tall. Amplituden er lengden i det komplekse planet og fasen er vinkelen

Fourier transformasjon

Vanligvis forskyves spektere slik at origo (u = 0, v = 0), ligger mitt i bilde. Det går fint siden F(u,v) er periodisk.



abs(F(u,v)) kan ha høye verdier med stor avstand mellom verdiene. Derfor kan det være vanlig å ta en logaritmisk transform, for vi er mer interessert i størrelsesforhold.
2D DFT er eperabel i to 1D DFTer

Linjestrukturen i en retning i bildedomenet blir representert som en linje normalt på retningen i fourierdomenet.
Skarpere kanter tilsvarer sum av mange sinusfunksjoner og skaper et bredt bånd i Fourier-domenet. Uskarpe kanter er færre sin-funksjoner og smalere bånd.

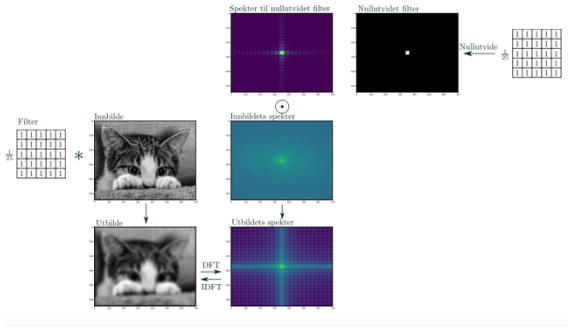
Bred struktur i bildedomenet = smal funksjon i Fourier-domenet. Og igjen, linjestrukturen går normalt på bildedomenet, i Fourier-domenet.

Fourier transformasjon II

https://www.uio.no/studier/emner/matnat/ifi/IN2070/v21/undervisningsmateriale/forelesning/inf2070-2021-fourier2.pdf

Konvolusjonsteoremet

Konvolusjonsteoremet sier at en elementvis multiplikasjon med filteret i frekvensdomene er det samme som en konvolusjon med samme filter i bildedommene.
Det gir vanligvis størst bidrag i spekteret for lave verdier av u og v, altså i midten av det filtrerte bildet. (ved sirkelkonvolusjon)
Filteret må nullutvites før multiplikasjon i frekvensdomenet.



Man kan designe et filter både i frekvensdomenet og i bildedomenet.

Filterdesign i Fourier-domenet

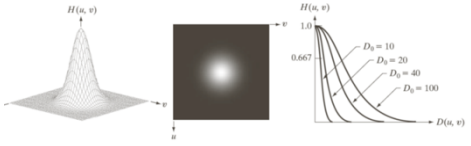
Ofte er filterets verdier mellom 0 og 1, 0 fjerner frekvens mens 1 bevarer.
Hvis nullfrekvensen (u = 0 og v = 0) i filteret er 1, bevares bildets middelverdi. Summen av gråtoner bevares.

Lavpass

Slipper gjennom lave frekvenser. Har en cut-off-frekvens, D_0 (en grense), som bestemmer at alle frekvenser lavere enn D_0 skal slippe gjennom. D_0 er ofte mellom 0 og 1.
Et ideelt filter vil sette alle verdier til 1 om $D(u,v) \leq D_0$ og 0 ellers, men er urealiserbart.

Ved å bruke et «ideelt» filter er det mulig å få «ringing»-effekt i bildet.

Butterworth lavpassfilter brukes som en glattere funksjon for å redusere ringing-effekten.
Et Gaussisk lavpassfilter har H(0,0) = 0 og er strengt avtagende i alle retninger.



Høypassfilter

Et høypassfilter kan defineres ut fra et lavpassfilter
hp(u,v) = 1-lp(u,v)

Båndpass og båndstopp

Båndpass slipper kun gjennom bidrag som hører til frekvenser i et bestemt intervall
Båndstoppfilter fjerner kun bidrag fra frekvenser.

Notchfilter

Slipper gjennom eller stopper filter i egendefinerte områder i Frourier-spekteret.

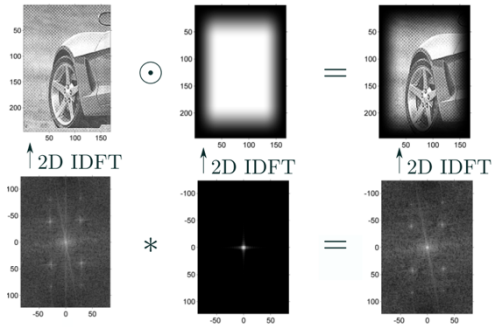
Filtrering i bildedomenet er kun raskere for små filtre.
Filtrering i frekvensdomenet er raskere når (m x n) >> $\log_2$(M x N) (M,N = bilde | m,n = filter)

Korrelasjonsteoremet

Korrelasjon i bildedomenet ⇔ elementvis multiplikasjon med F(u,v) kompleks konjugert.
Brukes f.eks. til å finne hvor i bildet visse mønstre eller objekter er.

Vindusfunksjon

Vindusfunksjoner brukes for å dempe effekten av bildekantene i spekteret. Bildekanten gir et «kunstig bidrag» langs aksene i spekteret grunnet diskontinuitet i bilderanden.
Det finnes mange forskjellige vindusfunksjoner.



Butterworth og Gaussisk er eksempel på vindusfunksjoner. Kan brukes i begge domener.

Kompresjon og koding

https://www.uio.no/studier/emner/matnat/ifi/IN2070/v21/undervisningsmateriale/forelesning/in2070_2021_11_kompresjon_i.pdf

Overføringshastighet betegnes med SI-prefikser (gange med hele 10'ere)
Filstørrelse er som regel oppgitt med binære prefikser.

I kompresjon vil vi komprimere data, sende det eller lagre det, og så dekompremere dataen. Alt etter hvordan dataen skal brukes er det viktig å tenke på hvor lang tid de kompresjon og dekompresjon tar.

Kompresjon

Kan deles inn i tre steg:
Transform – representere bildet mer kompakt (reverserbart)
kvantisering -avrund representasjonen (ikke reverserbart)
koding -produsere og bruke en kodebok (reverserbart)
Det kan gjøres både tapsfri/lossless og ikke-tapsfri/lossy

Redundans

I ikke optimal kode er det mulig å fjerne deler av koding uten å fjerne relevant informasjon.

Psykovisuell redudans

Fjerne informasjon vi ikke kan se/høre
Ikke tapsfri

Interbilde-redundans

Likhet mellom nabobilder i en tidssekvens. Lagre deler av bilde som tidssekvens og ellers differanser
Tapsfri

Intersampel-redundans

Likhet mellom nabopiksler. En rekke med like piksler kan representeres som ved antall like piksler.
Både tapsfri og ikke tapsfri

Koding-redundans

Enkeltsymboler/piksler blir ikke lagret optimalt. Gitt som gjennomsnittlig kodelengde minus et teoretisk minimum.
Både ikke tapsfri og tapsfri

Kompresjonsrate og redundans

$$CR = \frac{b}{c}$$

$$R = 1 - \frac{1}{CR} = 1 - \frac{c}{b}$$

- Også kalt plassbesparelse (eng. *space savings*).
- Ofte oppgitt i prosent.
- Prosentverdien kan kalles «percentage removed»:

$$PR = 100\,R = 100\left(1 - \frac{c}{b}\right)$$

b = faste antall biter per symbol i den ukomprimerte datamengden
c = gjennomsnittlig antall biter per symbol i den komprimerte datamengden.
c = $\sum_{i=0}^{G-1} b_i * p_i$

Entropi

$$H = \sum_{i=0}^{G-1} p_i I(s_i) = -\sum_{i=0}^{G-1} p_i \log_2 p_i$$

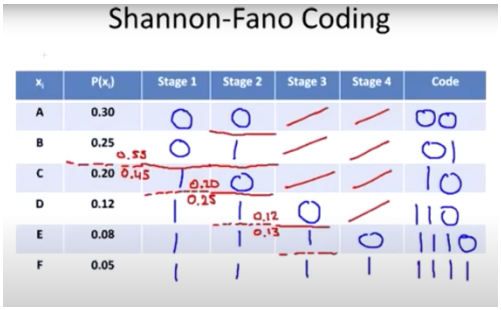
Gjennomsnittlig informasjon per symbol.
H = entropi, p_i = sannsynlighet for pikselverdi
Entropien setter en nedre grense for hvor kompakt en sekvens kan representeres.
Er sannsynlighet for alle piksler like, er entropien lik antall bites.
Er alle piksler like er H = 0

Koding

Alfabet er mengden av alle mulige symboler/alle mulige gråtoner.
hvert symbol får et kodeord, kodeboken beskriver alle kodeordene og deres betydning.
Det finnes mange typer for koding, bl.a. naturlig binærkoding, huffman-koding, shannon fano-koding osv..

Shannon-Fano

Ideen er å sortere rekken alfabetet etter sannsynligheten for hver forekomst. Så rekursivt dele det du arbeider med inn i to grupper med så lik sannsynlighet for hver den, på hver side. Så gi hver gruppe hvert sitt nye bit til alle har fått egne idividuelle bits.



Huffman-koding

Huffman-koding er optimal om vi skal kode symbol for symbol.
Ganske like prinsipp som Shannon-Fano, men her starter man å bygge treet fra bunnen av.
Man sorterer alle brukene/sannsynligheten for hver gråtoneverdi/bokstav og finner de to minste verdiene. Foreldrenoden til dette subtreet blir sannsynligheten til de nodene addert.
Hvis den nå laveste noden er mindre enn denne foreldrenoden vil den bli en del av subtreet og bygge videre rekursivt. Er den større vil den lage et eget subtre og fortsette.

For både SF og huff danner ingen kodeord prefiks i en annen kode. Så en kodesekvens er unik og instantant dekodbar. Hyppige symboler har kortere kodeord enn sjeldne symboler.

Huffmann har ingen redundans om $p_i = 1/2^k$

Aritmetisk koding

Er som SF og huffman taprsfri kompresjon og koder mer sannsynlige symboler mer kompakt. Aritmetisk koding lager ikke kodeord for enkeltsymboler. Men en sekvens av symboler kodes som et tall mellom to grenser. Tankegangen er at man representerer et symbol som et intervall mellom 0 og 1, så deler man det intervallet inn i flere intervall for å representere to symboler etter hverandre som starter med det første symbolet. En ulempe med dette er at for lengre symbolsekvenser krever god presisjon på flytall.

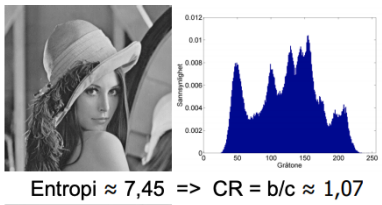
Aritmetisk er bedre for lengre symbolsekvenser, huffman er bedre for flere symboler i alfabetet. Aritmetisk komprimere ofte litt bedre for vanlige bilder, men er mer regnekrevende.

Kompresjon og koding II

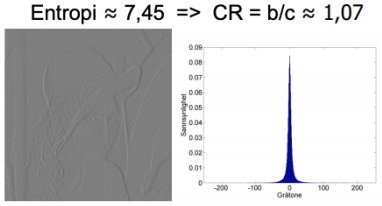
https://www.uio.no/studier/emner/matnat/ifi/IN2070/v21/undervisningsmateriale/forelesning/in2070_2021_12_kompresjon_ii.pdf

Kompresjonsmetoder - Tapsfri

Differansetransform



Entropi ≈ 7,45 => CR = b/c ≈ 1,07



Entropi ≈ 5,07 => CR = b/c ≈ 1,58

Transformer bildet med en reversibel transform som utnytter at horisontale nabopiksler ofte har ganske lik gråtone. Det transformerte bildet blir differansen mellom pikslene i originalbildet
Naturlig binærkoding av differansene er ikke optimalt.

Løpelengdetransform

Bruker også at de fleste bilder har objekter med lignende gråtoner. Denne metoden krever ekte likhet mellom pikselverdier, ikke bare omtrent likhet slik differansetransform. Fungerer spesielt bra for binære bilder. Transformen er reversibel.

Naturlig binærkoding

Alle kodeord er like lange, man legger til 0'ere foran slik at hver kode blir like lang.

Gray code

Denne kompresjonsmetoden ønsker å bruke at to gråtoneverdier som er i nærheten av hverandre stykemessig skal representeres som så like bit-verdier som mulig. Dermed kan koden kompremeres med løpelengdetransform. Her vil alltid bare én bit endres når gråtonen endres med 1. Det er ganske likt naturlig binærkode, bare at kodeordene som brukes er forskjellige. Hos begge er hvert kodeord like langt.
For transformasjon mellom binærkode og gray-kode se s 11.
Gray-kode har færre bitplan med støy enn BC.

Lempel-Ziv-Welch-transform

Denne metoden komprimerer ved hjelp av en LUT. Det som er spesielt med LZW er at den lager LUT'en fortløpende mens koden komprimeres. Dermed blir LUT'en optimalisert for gjeldende kode, mens LUT'en kan lages selv av mottakeren.
LZW er mye brukt, i bl.a. GIF. LWZ-koden som lages kan kodes videre med f.eks. Huffman-kode. Vi kan også lage faste prosedyrer for sletting av lite brukte/gamle koder.

Kompresjonsmetoder – ikke-tapsfri

For å få høyere kompresjonsrate kan man bruke ikke-tapsfrie metoder ved å f.eks. rekvantisere til færre antall gråtoner, resample til lavere romlig oppløsning, eller bruke filterbaserte metoder.

Når vi bruker ikke-tapsfrie metoder er det viktig at bildekvaliteten til det komprimerte bildet er god nok. Man kan måle feil i det nye bildet kontra originalbildet, men det er ikke alltid best mulig, fordi øynene våre oppdater ikke alle feil like godt. Derfor må vi ta i betraktning flere parametere; Feil rundt kanter er ille, feil i forgunnen er verre enn feil i bakgrunnen, manglende eller falske strukturer er ille. Dermed bør kompresjonsgraden variere rundt i bildet.

JPEG

Har både en tapsfri og en ikke-tapsfri variant

Ikke tapsfri

Bruker 2D diskret cos-transform (2D DCT)
Bildet deles inn i blokker på 8x8 piksler. Hver blokk kodes separat. Trekk fra 2^{b-1} (b=maks gråtoneverdi). Hver blokk vil deretter gjennomgå en 2D DCT. Blokken vil så punktutvides med en vektmatrise, så avrundes til nærmeste heltall. ´Vi vil få en AC og en DC-del fra 2D DCT. Disse blir behandlet hver for seg og komprimert henholdsvis med løpelengdetransform og differansetransform.
Disse prosessene kan så reverseres. Det vil ikke gi akkurat samme resultatblokk, men differansene vil ideelt være små.
For fargebilder deles bilde inn i 3 fargekanaler før en lik kompresjon foregår på hver fargekanal.
Rekonstruksjon av gråtonebilder etter JPEG-transform kan gi 8x8-piksel blokk-artefakter, glatting og ringinger. Graden av dette avhenger av vektmatrisen.

Tapsfri

I tapsfri JPEG brukes prediktiv koding der man prøver å predikere en piksel ved hjelp av naboer. For vanlige fargebilder er kompresjonsraten ≈ 2.
Brukes for det meste i medisinske anvendelser der kompresjonsfeil er uakseptabelt.

Digital video

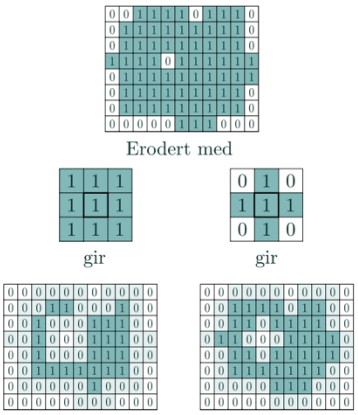
Bruker ofte både 2D DCT og prediktiv koding med bevegelse-kompensasjon der man ser på både tidligere og fremtidige bilder.

Morfologi

https://www.uio.no/studier/emner/matnat/ifi/IN2070/v21/undervisningsmateriale/forelesning/in2070-morfologi-2021.pdf

Modifiserer formen til objekter gjennom lokale operasjoner.
Kan brukes til å fjerne uønskede effekter som små støy-objekter og glatte ut omrisse, fylle ut eller lenke sammen objekter. Eller til å analysere objekter ved å tynne de ut og finne omrisse til objekter.
Er ofte enkelt og raskt.

Konsept



Passer = likhet
treffer = en piksel treffer
Man lager et strukturelement til et binært bilde med 0 eller 1. Origo velges. Vi utfører så en slags konvoulsjon med strukturelement som filter. Start slik at origo overlapper med bildet. S = strukturelement

Erosjon

Om S passer med inn-bildet blir origo til ut-bilde 1, ellers 0
Dette vil krymper/tynner ut objekter i bildet. Piksler fjernes innefra om objektet har hull. Større strukturelement gir mer erosjon. Man kan erodere et bilde flere ganger for større effekt.

Dilasjon

Roter S 180 og få S^. Om S^ treffer inn-bildet blir origo lik 1 i ut-bildet, ellers 0.
Dilasjon utvider objekter og fyller hull. En større S, gir større dilasjon.

for effekter se s.19

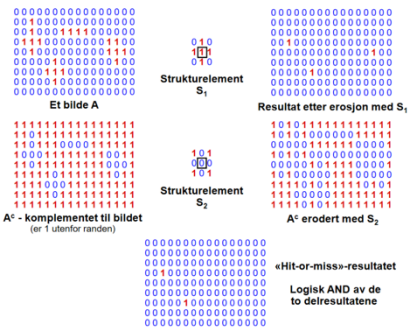
Både erosjon og dilasjon er dualiteter, dvs. vi kan beskrive de ved hjelp av hverandre.
Erosjon er ikke kommutativ eller assosiativ.

åpning bruker erosjon så dilasjon, lukking bruker dilasjon, så erosjon.

Granulometry

Brukes for å bestemme størrelsen til objekter i bilder. Antar at vi har objekter i en farge med bakgrunn i en annen farge. Regner så åpning mens man øker radius på SE hver gang og summer alle pikselverdiene hver gang. Plott differansen mellom summen, der differansen er størst forteller oss størrelsen til objektene i bildet.

Hit-or-miss-transformasjon



Hvis vi har et SE, S, som er definert ved $\{S_1,S_2\}$ som er to SE uten noe til felles. Da vil vi kunne ta f (erosjon) S_1 AND NOT(f) (erosjon) S_2. Vi vil da kunne finne eksakte mønstre.

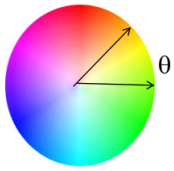
Morfologisk tynning
Repitisjon av erosjon helt til det ikke er endring lenger. Dette kan fjerne alle piksler bortsett fra det som definerer utstrekningen av et objekt.

Farger og fargerom
https://www.uio.no/studier/emner/matnat/ifi/IN2070/v21/undervisningsmateriale/forelesning/in2070-2021-farger.pdf
Vi er følsomme for lys mellom 350 og 760nm, og vi kan skille mellom ca. 100 rene farger og ca. 600 farger med intensitet og ca. 360 000 med metning.

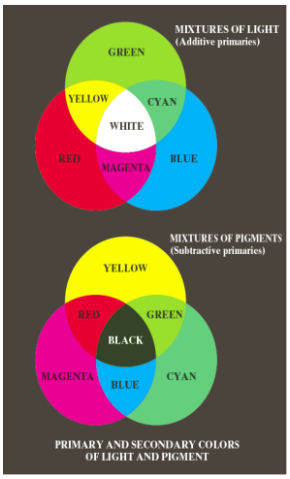
En farge kan representeres digitalt på mange forskjellige måter. RGB, HSI, CMY og mange flere.

Kromatisitet
Kromasitet og intensitet(lyshet) beskrive en farge, Kromasitet beskriver både dominerende bølgelengde og fargens metning. To gråtoner har samme kromasitet, men forskjellig intensitet.

- Tenk deg en sirkel der bølgelengden varierer med vinkelen θ.
  - Full metning ytterst ved radius r=1.
  - Minker r langs samme θ, så endres kun metningen.



Farger kan representeres med 3 variabler og man kan kalle disse X, Y og Z = 1. To av parameterne velger fargen, én velger intensitet.



Additive vs. subtraktive fargesystemer
Lys mikses additivt med primærfargene R, G, B og sekundærfargene Cyan, Magenta og gul. Øyet, kamera og skjermer er additive.

Maling/farger med pigment er subtraktive med primærfargene gul, cyan og magenta (CMY). Starter med svart og subtraherer farger. Brukes i printere

Fargemodeller
HSI – Hue, Saturation, Intensity
RGB og HIS har samme primær og sekundærfarger
Hue er ren farge og gir bølgelengden i det elektromagnetiske spektrum.
Saturation(metning) hvor mye grått inneholder fargen. Ved S = 0 blir fargen grå uavhengig av hue.
H og S beskriver sammen fargen (kromatisitet)
HSI er egnet til å beskrive farger, RGB er egnet til å generere farger.
Konversjon mellom disse to metodene er mulig se slides 24++

YIQ

Brukes som standard for TV og video i USA.
Y er iluminans, I og Q er krominanskomponentene. Samme signal kan brukes på farge og gråtoneskjermer.
Transform mellom YIQ og RGB kan uttrykkes ved matrisemultiplikasjon.

YCbCr-modellen

Vanlig fargemodell for digital TV og video
Y er iluminans (luma)
Cb er blå minus luma (B-Y)
Cr er rød minus luma (R-Y)
Mens RGB kan både være analog og digital er YCbCr kun digital men har en analog «tvilling» YPbPr

YUV-modellen

Brukes i analog TV, ganske likt konsept som YCbCr, Y = luma, U = B-Y, V = R-Y
Et videokamera må konvertere RGB data som regisreres i fokalplanet til enten YUV, YPbPr eller YCbCr for å så konvertere tilbake til RGB for å vise på mange typer skjermer.

Alfa-kanal

i RGBa spesifiserer alfakanalen om fargen er helt eller delvis transparent, der 0 er helt transparent og 255 er helt ugjennomsiktlig.

Printing

Gråtonebilder - I en printer med kun svart blekk vil gråtoner printes ut ved at hver piksel deles inn i flere deler, og printeren printer presist bare deler av pikselen sort.
Fargebilder – Bruker CMYK-modellen.

Dithering

er å tegne å gråtonebilde kun ved hjelp av sorte og hvite prikker der samling av sorte prikker oppfattes som sort, og desto flere hvite prikker i et område, desto lysere/gråere oppfattes det. Diffusjon hjelper å minske feil i dithering.

Pseudo-farger

er gråtonebilder der man har tilordnet hver gråtone en RGB-farge. Fremhever små gråtoneforskjeller.

Fargebilder og histogrammer

Egentlig er histogrammet til et fargebilde en 3D-kube, men vi pleier å skrive det på 2D-form som tre overlappende kurver.
Histogramutgjevning fungerer generelt dårlig på RGB-bilder, men om man transformerer bildet til HSI, gjør histogramutgjevning på I-komponenten og transformerer tilbake blir det bra resultat. Det samme gjelder for lavpassfiltrering.
Laplace-filtrering kan gjøres på alle fargene i RGB eller kun på I i HSI.

Terskling kan gjøres på hver kanal, så kombinere kanalene. I HSI lages en maske ved å terskle S-bilde til en valgt prosent, så multiplisere masken med H-kanalen og velge et intervall i H som svarer til ønsket farge.