INF 1040
Digital video – digital bildeanalyse

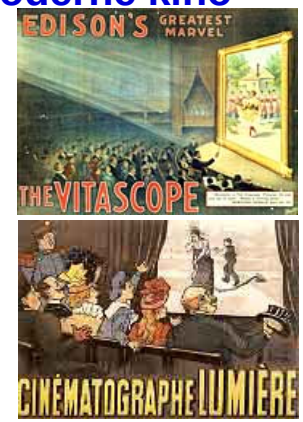
- **Tema i dag :**
  1. **Fra stillbilder til videosekvenser**
  2. **Digital video**
  3. **Hvorfor kan video komprimeres?**
  4. **Digitale videostandarder, lagring på DVD**
  5. **Digital bildebehandling og bildeanalyse**

- **Pensumlitteratur: Læreboka, kapittel 16 og 17.**
- **Neste gang: Kompresjon og koding**

Fra bilder til video

- **Når vi lukker øynene, tar det litt tid før "etter - bildet" forsvinner, spesielt hvis intensiteten er høy (i deler av) bildet.**
- **Bildet forsvinner gradvis (eksponensielt) fra retina.**
- **Det er en kort periode da vi ikke kan ta inn ny informasjon -- selv om øynene er åpne.**
- **Dette kan vi teste med periodiske lysblink.**
- **"Critical flicker fusion frequency" øker med økende luminans.**
  - **4 Hz for lave lysstyrker (bare stavene er virksomme)**
  - **ca 60 Hz for kraftig lysstyrke (bare tappene er virksomme)**
  - **konvergerer mot 80 Hz**

Fra Lumiere og Edison til moderne kino



- **Lumiere og Edison laget filmfremvisere med 10 bilder/sekund.**
- **Dette var (nesten) tilstrekkelig til å gi en illusjon om "kontinuerlig" bevegelse.**
- **Problemet med at tilskuerne ble slitne av flimmer ble løst ved å vise fram hvert bilde 3 ganger, en 30 "flash-per-second" presentasjon.**
- **Filmindustrien utviklet dette til 24 bilder/sek. som gir en mye bedre illusjon om kontinuerlig bevegelse.**
- **For å fjerne opplevelsen av flimmer blir bildene vist 2 ganger, noe som gir en illusjon om 48 bilder/sekund.**

Luminans-integrasjon

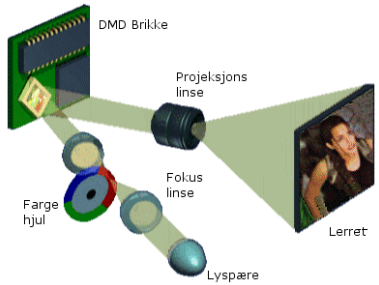
- **Når lysblinkene kommer tett nok etter hverandre i tid, vil de oppfattes som et jevnt lys, uten "flicker".**
- **Den effektive oppfattede luminans vil være gjennomsnittet over belysnings-sykelen, som jo egentlig består av en lys og en mørk del.**
- **Synssystemet vårt integrerer lysintensiteten.**
- **Dette betyr at vi kan variere den effektive luminans ved å holde den fysiske intensiteten til lyskilden konstant, og variere lengden av den mørke delen mellom lysblinkene.**

DMD – ”Digital Micromirror Device”

- Et stort antall bittesmå speil på en elektronisk brikke.
  - Hvert speil svarer til en piksel i det bildet vi skal projisere på lerretet.
  - Hvert speil vippes mot prosjektor-linsen (on), eller en ”heat-sink” (off).
  - Forholdet mellom tiden ”on” og tiden ”off” svarer til gråtoneverdien.
  - Dagens DMD brikker kan gi kontrastforhold 4000:1 (12 biter).
- Digital Cinema Initiatives spesifiserer to ”container” kino-formater
  - ”2K”: 2 048 x 1 080 (24 eller 48 Hz); ”4K”: 4 096 x 2 160 (24 Hz)
  - 3 x 12 biter XYZ fargerom.

DLP – ”Digital Light Processing”

- ”Single-chip” prosjektører:
  - Bruker roterende hjul med fargefiltre for rødt, grønt og blått lys.
  - Synkronisering: når det røde lyset faller på DMD’en blir vippingen av speilene bestemt av innholdet av R-kanalen i bildet, osv.



- ”Three-chip” prosjektører:
  - Tre adskilte DMD-brikker, en for hver kanal i RGB-bildet
  - Lysstrålene fra de tre DMD’ene kombineres gjennom faste fargefiltre eller et prisme, og projiseres til lerretet.
- Ulike fordeler og ulemper med de to teknikkene.

TV og video

- TV og video er en sekvens av stillbilder.
- Et nytt stillbilde sendes så ofte at øyet vårt oppfatter det som et kontinuerlig signal / kontinuerlig film i tid.
  - Vi ser ikke skiftene mellom stillbildene.
- Hvis neste bilde viser et objekt som har flyttet seg litt, oppfatter øyet det som om vi ser bevegelse.
- Skal vi se TV-bilder med naturlig bevegelse, må de skifte med en frekvens på minst 50-60 Hz.
- Sampling av bevegelse kan gi aliasing:
  - F.eks. hjul som roterer baklengs på film / TV

Interlacing

- Interlacing betyr at hvert bilde deles i to separate ”felter”
  - Første felt består av linjene 1,3,5,7,9,… (oddetalls linjer)
  - Andre felt består av linjene 2,4,6,8,10,… (partalls linjer)

- 30 ganger i sekundet skifter linje 1, 3, 5,...

  og midt mellom hvert av disse skiftene skifter linje 2, 4, 6... .
- Tilsammen skiftes bildet 60 ganger i sekundet, men man trenger ikke å overføre mer enn 30 bilder pr sekund til TV-skjermen.
- Dette er en teknikk for flimmerfri framvisning med halv båndbredde.
- Hvert felt kan splittes i odde- og partalls kolonner (“double interlace”).
- Bruk ikke klær med fiskebeins-mønster på TV !

Analoge videostandarder

- Component video
  - "high-end" video system med tre separate signaler for YPbPr
  - krever mer båndbredde, god synkronisering, gir skarpest bilde.
- Composite video
  - mikser farge og intensitet til ett signal.
  - alt bortsett fra audio og sync-signalet sendes på én kabel.
  - Ulempen er noe interferens mellom luminans og kromatisitet.
- S-Video
  - et kompromiss mellom Component- og Composite video,
  - to kabler, en for luminans og en for kromatisitets-signalet.
  - mindre "crosstalk" mellom farge og gråtone,
  - gir skarpere bilde enn Composite video.

Analog TV og video

- Øyet har høyere geometrisk oppløsning i intensitet enn i farge.
  - Vi kan ha lavere geometrisk oppløsning i kromatisitet enn i luminans.
  - Vi kan sende mindre detaljert fargeinformasjon.
- NTSC *(National Television System Committee)*
  - 525 scan-linjer per bilde, 30 bilder per sekund.
  - YIQ som fargemodell.
- PAL *(Phase Alternating Line)*
  SECAM *(Systéme Electronique Couleur Avec Mémoire)*
  - 625 scan-linjer per bilde, 25 bilder per sekund
  - YUV som fargemodell.

Digital video

- Fordelene med digital video er mange.
- Videosekvensene kan lagres på digitale media.
- Gjentatte opptak på samme media uten redusert kvalitet.
- Man kan gjøre bildebehandling og redigering.
- Videosekvenser integreres i multimedia applikasjoner.
- Datamengden reduseres ved kompresjon og koding.
- Det er enklere å beskytte opptakene med kryptering.
  - Beskytte mot uautorisert innsyn
  - Beskytte forretningsinteresser

Optimal digital video

- Digital video består av en tids-sekvens av digitale still-bilder.
- Geometrisk oppløsning:
  - Husk Nyquist-teoremet og Rayleigh-kriteriet !
- Fargekamera: For hvert piksel måles lysintensitet i tre separate bånd i det elektromagnetiske spekteret.
- For hvert bånd (red, green, blue) skal vi :
  - 1. beregne gjennomsnittsverdien i hver rute
  - 2. skalere slik at den passer innenfor det tall-området vi skal bruke
  - 3. kvantiserer verdiene til nærmeste heltalls verdi i tall-området
- Bilde-frekvensen må være høy nok (50-60 Hz) til å fange opp all den tids-informasjon som vi er i stand til å oppfatte.

Et overslag over datamengder

- Gitt en fargeskjerm med 1024 ·768 piksler
- Vi vil se på en bildesekvens som varer i 90 minutter
- Vi skal vise fram 60 bilder/sekund, 24 biters RGB pr piksel.
- Hvor stor er den filen som vi henter disse bildene fra?

  1024 · 768 · 8 · 3 · 60 · 60 · 90 / (1024 · 1024 · 1024 · 8) = 712 GiB
- Selv med en smart 8 biters LUT, vil denne filmen kreve

  712 / 3 = 237.3 GiB
- Dette er så store datamengder at vi er nødt til å bruke kompresjon og koding.

YCbCr - igjen

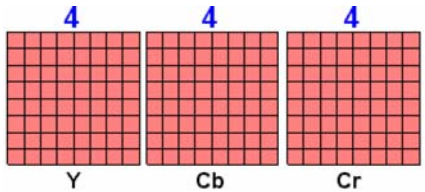
- YCbCr er det samme som YUV i digital versjon.
- Vi trenger ikke å sample kromatisiteten (Cb og Cr) med like høy rate som intensitets-komponenten Y.
- Konsekvensen er en kompresjon, fordi vi ikke samler på data som vi allikevel ikke er i stand til å se.

Chroma 4:4:4, 4:2:2, 4:1:1 eller 4:2:0

- Subsamplingen i YCbCr betegnes med talltrippelet 4:n:n
- n angir hvor mange av 4 originale piksler i Cb og Cr som faktisk sendes.
- 8 · 8 piksler er en "blokk" i et videobilde.
- En blokk er grunnsteinen i JPEG-kompresjon av digitale bilder.

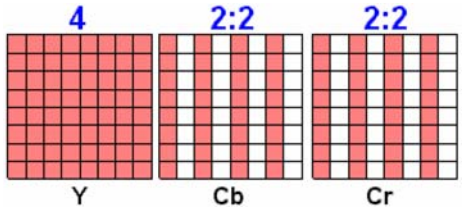
4:4:4

- Her er Cb og Cr samplet med samme rate som Y.
  - ingen chroma subsampling
  - alle 4 piksler i Y, Cb og Cr blir sendt.
  - MPEG-2 støtter 4:4:4 koding.
  - Dette er egentlig en ”oversampling” av Cb og Cr
  - Video resamples til 4:4:4 før konvertering mellom fargerom



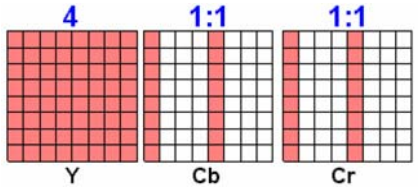
4:2:2

- horisontal subsampling av Cb og Cr med en faktor 2.
  - ”Co-sited” betyr at Cb og Cr samples på samme tidspunkt som Y.
  - Alternativet er at to og to piksler midles.
  - 4:2:2 sampling er mye brukt i profesjonell digital video, i DV, Digital Betacam og DVCPRO 50.
  - 4:2:2 er en opsjon i MPEG-2.



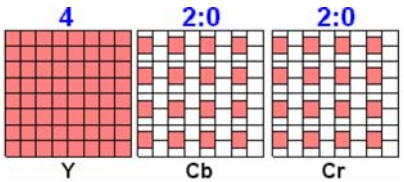
4:1:1

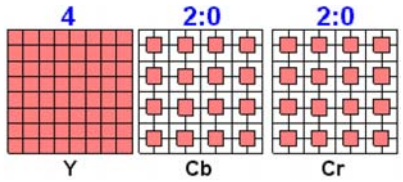
- Cb og Cr samples med en fjerdedel av den horisontale oppløsningen til Y.
  - ”Co-sited” betyr at Cb og Cr samples på samme tid som første sampel i Y.
  - Co-sited 4:1:1 brukes i DV, DVCAM og DVCPRO formatene.



4:2:0

- Subsampling med faktor 2 både horisontalt og vertikalt
  - enten en subsampling horisontalt og en midling vertikalt,
  - eller en midling begge veier.
  - MPEG-1 bruker den første metoden, MPEG-2 den andre.
  - JPEG benytter 4:2:0.



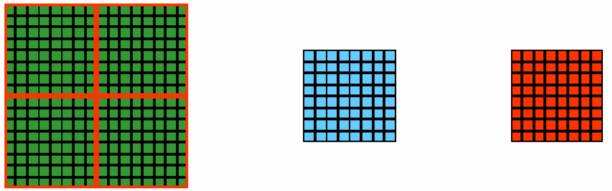


HDTV

- HDTV’s historie
  - Dreier seg mest om 16:9 istedenfor 4:3
- 1990: full-oppløsnings HDTV – uten interlacing
  - interlacing gir sagtakkede kanter, flimrende horisontale kanter.
- 1993: Digital HDTV.
- Mange kombinasjoner av parametre i ATSC-standarden.
- Flere HDTV-systemer er allerede i bruk.
- Nytt segment: kringkastet digital TV til mobile mottakere.
  - Bedre mobil-skjermer => kamp om markedsandeler og kunder

Makroblokker

- Videostandarden H.261 bruker
  - en ”makroblokk” på 16 · 16 piksler i intensitets-komponenten Y, en 8 · 8 blokk i Cb og en 8 · 8 blokk i Cr.
- 6 stk 8 · 8 blokker gir oss all informasjon fra en makroblokk på 16 · 16 piksler i bildet.
  - En makroblokk betegnes med MB
  - Dette har ikke noe med forkortelsen for MegaByte å gjøre.



Enkeltbilder og bildeserier

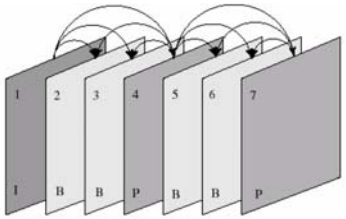
- JPEG bruker 8 · 8 blokker for å komprimere enkeltbilder.
- Vi kunne brukt slik koding av hvert bilde i en bildeserie.
- Men i en tidssekvens er nabo-bildene som regel svært like.
- Forandringer fra bilde til bilde skyldes at
  - objekter i bildet har flyttet seg,
  - kamera har beveget seg (pan)
  - vi har endret fokallengden (zoom)
  - - som regel skjer disse forandringene langsomt

Prediksjon og prediksjonsfeil

- I alle fall er det en korrelasjon mellom bildene.
- Vi kan da
  - modellere eller prediktere et bilde fra det forrige bildet,
  - subtrahere det faktiske bildet fra det modellerte,
  - bare ta vare på denne prediksjons-feilen
    - Prediksjons-feil finnes sannsynligvis bare i deler av bildet
    - Der den finnes er den sikkert ikke så stor,
      - så den kan lagres ganske kompakt –
      - med langt færre biter enn selve bildet.
- Prediksjon kan gjøres både forover og bakover i bildesekvensen.
  - Dette kalles *inter* mode.

Inter kontra intra

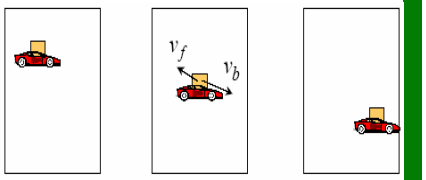
- Men med jevne mellomrom må vi kode virkelige bilder, ikke bare prediksjonsfeil. Det samme gjelder hvis prediksjonsfeilen blir for stor. Dette kalles *intra* mode.
  - Bilde 4 er predikert fra bilde 1,
  - Bilde 2 og 3 er predikert både forover og bakover fra bilde 1 og 4



- Kodes her i rekkefølgen 1 4 2 3 7 5 6
  - 1 kodes direkte (intra),
  - resten kodes som prediksjonsfeil (inter).

Bevegelses-estimering

- Prediksjonen blir bedre hvis vi kan kompensere for bevegelse.
- Gjøres for hver makroblokk i det bildet vi skal predikere.
  - Vi søker gjennom et referansebilde både foran og bak i sekvensen
    - til vi finner den makroblokken som ligner mest.
  - Undersøker om makroblokkene må forskyves
  - Maksimum forskyvning er 15 piksler.
- Estimering for hver makroblokk
  - beskriver forflytning med få vektorer
    - istedenfor feil i mange piksler.
- Forskjellige video standarder håndterer dette problemet på forskjellige måter.



Digitale video standarder - 1

- H.261 var den første standarden for video koding,
  - mest beregnet på video-konferanser på ISDN nettverk
  - fast båndbredde på 64 – 30 · 64 Kbit/s.
  - Bevegelseskompensasjon i heltalls piksler.
- H.263: videokonferanser over normale telefonnett og mobile nettverk,
  - og gir økt kvalitet ved lavere bitrate (under 54 Kbit/s).
  - Bevegelseskompensasjon med presisjon på ½ piksel.
- MPEG-1 video er optimert for CD-ROM og video på internett, og
  - gir god kvalitet ved 1.5 Mbit/s.
  - Bevegelseskompensasjon forlengs og baklengs, oppløsning ½ piksel.
- MPEG-2 video er beregnet å håndtere TV-sendinger,
  - både i progressiv og interlaced mode.
  - Den skal håndtere både 4:2:2 og 4:4:4 subsampling av YCbCr,
  - og en datarate på opptil 15 Mbits/s.

Digitale video standarder - 2

- MPEG-4 åpner for interactive TV og ”virtual reality”
  - ved at objekter kan manipuleres og scener bygges opp i dekoderen.
  - Bygger på objekt-basert video-koding
  - Gir muligheter for blanding av bilder, grafikk og animasjon.
- MPEG-7 åpner for søk i multimedia dokumenter,
  - Innholdsbaserte søk i videosekvenser og bildedatabaser.
- H.264 øker kodingseffektiviteten.
- MPEG-21går videre fra MPEG-7,
  - og er ment å gi tilgang uavhengig av type nett eller type plattform,

Digital Versatile Disc

- Ligner på CD-plater, lagrer lyd, video og data i ett digitalt format.
- Laser med kortere bølgelengde, gir høyere lagringskapasitet
  - bitene kan lagres tettere langs spiralsporet
  - spiralsporet kan vikles tettere.
- Redusert datamengde til feildeteksjon og korreksjon i forhold til CD.
- Det meste av kapasitetsøkningen skyldes
  - bruker begge sider av platen,
  - to informasjons-lag per side
    - ett delvis reflekterende og ett reflekterende.
    - Ett lag har en kapasitet på omtrent 4,7 milliarder byte, omtrent 7 ganger det man får inn på en konvensjonell CD.
    - Med to lag per side kommer man opp i 17 milliarder byte.

Objektgjenkjenning i digitale bilder

Motivasjon: det vi egentlig trenger å lagre fra et bilde, er ofte informasjon om akkurat hva det inneholder.

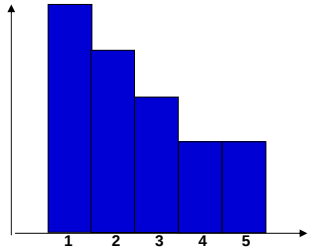
Noen steg i et enkelt system for objektgjenkjenning:

- **Kontrastendring**
- **Filtering for å fjerne støy**
- **Segmentering av objekter og bakgrunn**
  - **Kantdeteksjon**
  - **Terskling**
- **Beregne egenskaper fra objektene**
- **Klassifisere objektene som ulike typer**

Hva er et histogram?

- **Anta at vi har et gråtonebilde med $m \cdot n$ piksler**
  - **hvert piksel har ordlengde b biter => vi har 2^b mulige gråtoner.**
- **Ser på gråtonen i hvert piksel, teller opp hvor mange piksler det finnes for hver pikselverdi, får vi et *gråtone-histogram*.**
- **Histogrammet *h(v)* er en tabell over antall forekomster av verdien v.**

1	3	2	1
5	4	5	3
4	1	1	2
2	3	2	1



Mer om histogrammet

- **Summen av *h(v)* for *v* fra 0 til $2^b - 1$, er antall piksler i bildet, $m \cdot n$.**
- **Dividerer vi alle *h(v)* med antall piksler i bildet, får vi et *normalisert histogram*, *p(v)*.**
- ***p(v)* er *sannsynligheten* for at vi finner pikselverdien *v* hvis vi velger et vilkårlig piksel i bildet.**
- **Summen av alle *p(v)* for v fra 0 til $2^b - 1$ er 1.**
  - **(summen av alle sannsynlighetene for at vi finner en pikselverdi mellom 0 og $2^b - 1$).**

Endring av kontrast

- **Vi forandrer på hvor lyst eller mørkt et digitalt bilde er ved å addere en konstant – en bias – til alle pikselverdiene:**

$$g(x,y) = f(x,y) + b$$

  - **Hvis $b > 0$ økes alle pikselverdiene i bildet, slik at et gråtonebilde vil bli lysere.**
  - **Hvis $b < 0$ blir bildet mørkere.**
- **Vi justerer kontrasten ved å multiplisere hvert piksel med en faktor:**

$$g(x,y) = a\ f(x,y)$$

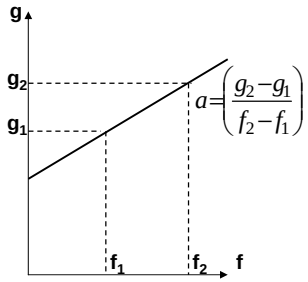
  - **Hvis $a > 1$ vil kontrasten øke, mens den minskes hvis $a < 1$.**
- **Et mer generelt uttrykk for endring av gjennomsnitt og kontrast:**

$$g(x,y) = a\ \ f(x,y) + b$$

Lineær gråtonemapping

- Vi vil at pikselverdier i området [f_1, f_2] i inn-bildet skal havne i gråtone-området [g_1,g_2] i ut-bildet.
- Dette er en lineær mapping fra f til g

$$g(x,y) = g_1 + \left(\frac{g_2 - g_1}{f_2 - f_1}\right)[f(x,y) - f_1]$$



altså en rett linje med stigningstall

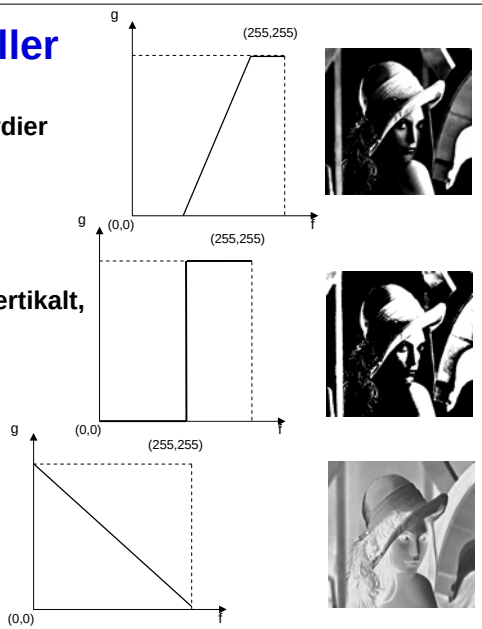
$a = (g_2-g_1)/(f_2-f_1)$

Sjekk at f_1 og f_2 mappes til hhv. g_1 og g_2

Et eksempel-bilde

Spesialtilfeller

- Et lite område av pikselverdier mappes til å bruke hele gråtone-skalaen.
- Hvis den skrå linjen står vertikalt, svarer det til en terskling av gråtoneverdiene.
- Vi kan invertere gråtonene og få et negativt bilde.

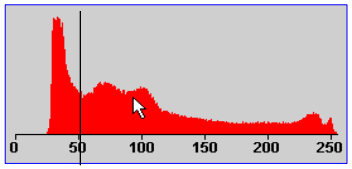


Terskling for å finne objektene i bildet

- Terskling:
  - sett alle piksler med pikselverdi < T til 0,
  - og alle piksler med pikselverdi >= T til 255.
- Utfordring: bestemme den beste verdien av T automatisk
  - Globale metoder bruker samme terskel for hele bildet.
  - Lokale metoder bruker en terskel som varierer over bildet.
  - I begge tilfeller brukes histogrammet til å finne terskelen T.

Terskelen T settes gjerne i lokale minimumspunkter i histogrammet.

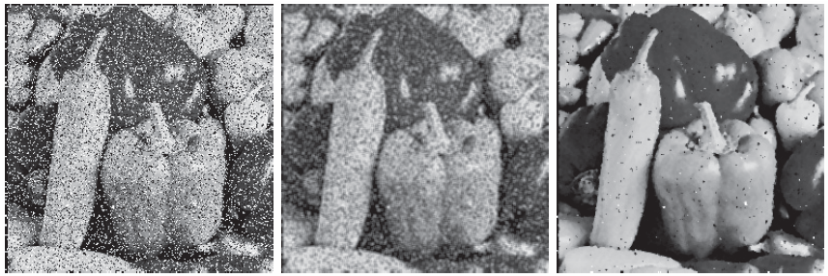
Kan vi finne T automatisk?



Støyfiltrering – middelverdi eller median

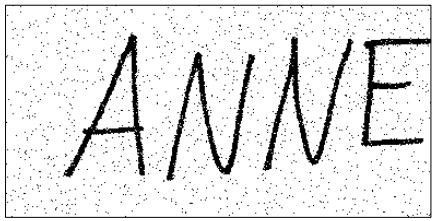
To enkle metoder for å filtrere et bilde:

- **Middelverdifilter: erstatt pikselverdien i et punkt med gjennomsnittet av pikselverdiene til nabopikslene.**
- **Medianfilter: erstatt pikselverdien i et punkt med medianen av pikselverdiene til nabopikslene.**
  - **Bevarer kanter i bildet bedre enn middelverdifilteret.**



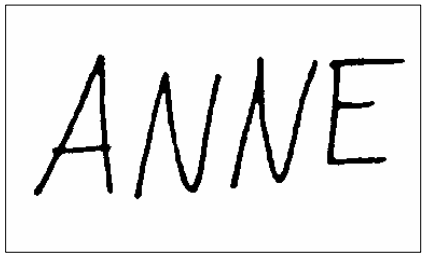
Terskling med og uten støyfjerning

Terskling uten filtering



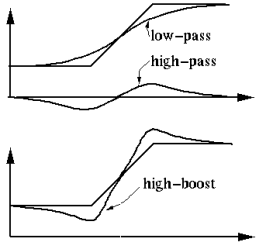
$$v_{out}(x,y)=\begin{cases}255 & hvis\ v_{in}(x,y)>T\\ 0 & hvis\ v_{in}(x,y)\le T\end{cases}$$

Terskling etter medianfiltering



Bildeforbedring: ”Unsharp masking”

- **Middelverdi-filtrering gjør bildet uskarpt.**
- **Vi kan subtrahere det uskarpe bildet fra originalen, og addere differansen til originalen.**
- **Resultatbildet vil virke skarpere enn originalen.**
- **Alternativ: original + høypass = ”high-boost”**



Bildeforbedring – hva er realistisk?

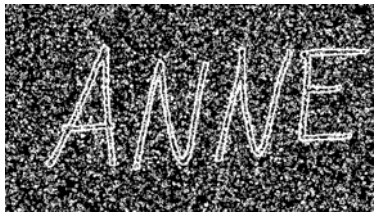
- **Hvis vi kjenner punktspredningsfunksjonen til kameraoptikken, kan vi i prinsipp regne oss tilbake til et skarpere bilde.**
  - **Men hvis det samtidig er støy i bildet, risikerer vi å forsterke denne støyen også.**
  - **Så dette er slett ikke enkelt!**
- **Det er ikke sant**
  - **at man kan gjenkjenne ansikter på skrå fra satellitt.**
  - **at grumsete bilder med noen få piksler kan gjøres fantastisk skarpe.**
  - **at et bilde av iris tatt på skrå fra relativt lang avstand kan brukes til identifikasjon og verifikasjon .**
- **Det merkelige er at veldig mange tror at alt dette er mulig.**
- **Vi blir dynget ned av ”tekno-babbel”.**

Kantdeteksjon i bilder

- Synssystemet er basert på deteksjon av kanter og linjer.
- Kanter i bildet gir oss ofte overganger mellom objekter.
- Kanter svarer til høy verdi av den deriverte i x- og y-retningen.
- Den deriverte approksimeres ved differanser,
  f.eks. $\delta f/\delta x = |f(x+1,y) - f(x-1,y)|$ i x-retningen
  og $\delta f/\delta y = |f(x,y+1) - f(x,y-1)|$ i y-retningen.
- Slike enkle operatorer er følsomme for tilfeldig støy i bildet.
- Medføre at vi finner falske kanter
  som ikke har noe med objektene å gjøre.
- Vi kan fjerne støy før kant-deteksjon.

Kantdeteksjon

Kantdeteksjon uten støyfjerning først



Kantdeteksjon etter medianfiltrering

Kantdeteksjon ved konvolusjon

- Støyfjerning ”smører ut” detaljer, kanter og linjer i bildet.
- Kombinerer de to operasjonenene,
  - midler i y-retning og deriverer i x-retning, og omvendt.
  - Bruker to operatorer – eller filtre, for eksempel:

$$h_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, \quad h_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

- Legger operatormasken oppå det digitale bildet,
  - multipliserer hvert underliggende piksel med verdien i masken,
  - summerer, og setter svaret i tilsvarende posisjon i ut-bildet.
  - Flytter masken til neste piksel i bildet og gjentar prosedyren,
  - Dette kalles *konvolusjon*, og skrives gjerne g(x,y) = f(x,y) * h

Gradientmagnitude og retning

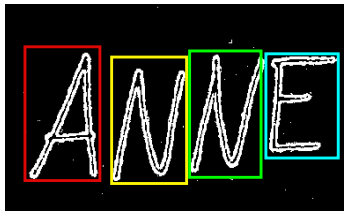
- Med to slike filtre får vi to ut-bilder som gir oss hhv.
  - gråtonegradienten i x-retning $g_x(x,y) = f(x,y) * h_x$
  - gråtonegradienten i y-retning. $g_y(x,y) = f(x,y) * h_y$
    - symbolet * betegner *konvolusjon.*
- Vi lager et gradient-magnitude bilde G(x,y)
  - som viser hvor ”bratte” kantene er i bildet :

$$G(x,y) = \sqrt{[g_x(x,y)]^2 + [g_y(x,y)]^2}$$

- Derivasjoner ikke bare abstrakt matematikk.
- *VI KAN DERIVERE DIGITALE BILDER,*
  - og dermed detektere kanter og linjer i bilder,
  - på samme måte som synssystemet vårt gjør.

Fra segmentert bilde til objekter

- Lag et objekt for hver region i bildet.



- Vi lager oss en datastruktur som inneholder en liste over objekter.
- For hvert objekt har vi en liste over alle piksler dette inneholder.
- Hvis vi vet noe om størrelsen på objektene,
  kan vi fjerne små objekter som vi regner med er støy.

Egenskapsuttrekking for objekter

- For hvert objekt, må vi beregne egenskaper
  - som skiller det fra andre objekter,
  - og som er felles for alle objekter av samme type.
- Stikkordet er er: hvordan beskriver vi formen til et objekt?
  - F.eks. Hva skiller en B fra en D, eller E fra F?
- Noen form-egenskaper er basert på alle pikslene i objektet
  - areal, tyngdepunkt, treghets-momenter (fysikk)
- Andre egenskaper er basert på objektets omriss
  - lengden av objektets omriss
  - ”bounding-box” (minste rektangel som inneholder objektet).
  - beskrive omrisset med f.eks. med splines eller Fourier-koeffisienter.

Klassifikasjon av objekter

- Fra et sett kjente objekter, bygger vi en database med beskrivelse for hver objekttype (en beskrivelse for hver bokstav).
- Databasen må bygge på tusenvis av objekter av hver klasse.
- Finner egenskapene for hvert objekt, matcher mot databasen
  og finner den objektklassen (bokstaven) det ligner mest på.
- Enkle metoder for å finne den klassen det ligner mest på:
  - Finn de 5 objektene den ligner mest på,
    velg objektklassen som forekommer oftest blant disse 5.
  - Regn ut korrelasjonen mellom objektet og alle objektene i databasen
  - Tilpass en statistisk sannsynlighetsfordeling (f.eks. normalfordeling) til hver klasse. Velg klassen med størst sannsynlighet.

Tekstgjenkjenning

- La oss si at vi skal kjenne igjen bokstaver i et bilde.
- Vi har segmentert bildet i objekter vha. terskling.
- Små og store bokstaver er forskjellige, vi må bruke en av delene eller trene systemet på begge deler.

A a B b C c

- Hvordan matcher vi bokstaver med ulik fontstørrelse?
  - Vi kan trene systemet på bokstaver i alle størrelser.
  - Vi kan skalere objektet slik at objektet har en viss størrelse

Skalering av objektet slik at det passer i en boks på 32 · 32 piksler

Tekstgjenkjenning - ulike fonter

- En bokstav kan jo se helt ulik ut i ulike fonter:

  A *A* *A* A Ulik stil, samme font

  A A A A A A A A A A A A A A A A Ulike fonter

- Matcher vi piksel for piksel, vil vi få dårlig gjenkjenningsrate.
- Håndskrevne symboler gjør det enda vanskeligere

Tekstgjenkjenning - flere utfordringer

- Ofte vil en bokstav ikke være helt sammenhengende etter segmentering, men fragmentert i flere delobjekter.
- To bokstaver kan også henge sammen.
- Noen bokstaver består av flere objekter, f.eks. å og i

Kontekst

- *Konteksten* kan hjelpe oss i den visuelle tolkningen av symbolet,
  - dette er vanskelig å automatisere.

  T /-\ E C /-\ T

  kan med noe velvilje leses som ”THE CAT”
- Statistikk over hyppigheten av bokstavene i engelsk tekst:
  - H mest sannsynlige bokstav etter T,
  - H mest sannsynlige bokstav foran E,
  - A mest sannsynlige bokstav foran T.
- Er ordet sannsynlig?
  - Sjekk i en digital ordbok (eksempel sms)
  - ordboka må være ganske omfattende for at resultatet skal bli godt.
- Alternativ: Sjekk sannsynligheten for at foreslått ord kommer etter de ordene man allerede har (veldig omfattende).

Ord som bilder

- ”Dgitial rerpsenetajson av tesketr, tall, lyd, bidler og viedo” er et ineterssnat krus som inenhloder mye rart!
- Hvis de første og de siste bokstavene i hvert ord er på rett plass, så kan vi blande de øvrige bokstavene og fortsatt lese og forstå teksten – dog noe langsommere!
- Vi leser ordene som ”bilder” og retter feilene.
- Dette klarer vi fordi vi assosierer med ord vi kjenner.
- Kan en PC lære å gjøre dette? !KNIHT

Tekstgjenkjenning i praksis

- Moralen er: robust tekstgjenkjenning er vanskelig å programmere.
  - Maskinen kan ikke konkurrere med øyet / hjernen i fleksibilitet.
  - Men mennesker gjør også feil når vi blir trøtte.
  - Maskiner er overlegne på store volumer og høy hastighet.
- Dette er et veldig viktig anvendelsesområde.
- Tekst som skal maskinavleses kan erstattes av strek-koder, men må da ofte gis som kombinert strekkode og tekst.
- Automatisk mønstergjenkjenning er vanskelig, men gøy:
  - For de som vil gå i dybden:
    - INF 2310 Digital bildebehandling
    - INF 4300 Digital bildeanalyse

Viktige punkter

- Lagringsbehovet for en RGB videosekvens kan reduseres ved
  - interlacing
  - fargeoppslagstabeller (LUT)
  - subsampling i YCbCr-fargerommet
  - koding av differanse mellom predikert bilde og bildet selv
  - bevegelseskompensasjon for hver makroblokk i bildet
- Det vi trenger å lagre, er en kompakt representasjon av objekter.
- Bildebehandling kan gi oss et bedre bilde
  - Støyfjerning, filtrering, ...
- Bildeanalyse kan redusere data til nyttig informasjon
  - Segmentering, egenskapsuttrekking, klassifikasjon, ...