

INF2310 – 30. januar 2019 – Ukens temaer
(Kap 2.4.4 og 2.6.5 i DIP)

- Geometriske operasjoner
 - Lineære / affine transformasjoner
- Resampling og interpolasjon
- Samregistrering av bilder

Anvendelser

- Forstørre deler av bilder for visuell inspeksjon («zoom»)
 - Rotasjon
 - Fiskeøyelinse
 - (Radar)avbildning av terreng
 - Generell linsekorrigerings
 - ...
- Samregistrere bilder
 - .. fra ulike sensorer (f.eks. CT, MR, US)
 - .. tatt på ulike tidspunkt / vinkler
 - .. med kart i en bestemt kartprojeksjon
 - Eks ansiktsgjenkjenning: Finne ansiktene i et bilde og transformere bildet slik at ansiktene i bildet blir på samme sted, orientering og i samme størrelse som i referansebildene
- Generere bilder fra andre kameravinkler
- Spesialeffekter

Affine transformasjoner kont.

- Transformerer pikselkoordinatene (x,y) til (x',y')

$$x' = T_x(x,y)$$

$$y' = T_y(x,y)$$

- Affine transformasjoner beskrives ved:

$$x' = a_0x + a_1y + a_2$$

$$y' = b_0x + b_1y + b_2$$

- På matriseform:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_0 & a_1 & a_2 \\ b_0 & b_1 & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \text{ eller } \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a_0 & a_1 \\ b_0 & b_1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} a_2 \\ b_2 \end{bmatrix}$$

Geometriske operasjoner

- Endrer på pikslenes posisjoner
- Transformerer pikselkoordinatene (x,y) til (x',y')

$$x' = T_x(x,y)$$

$$y' = T_y(x,y)$$

- T_x og T_y ofte gitt som polynomer

Merk: Her er det ikke pikselverdiene, men piksel-koordinatene x og y som endres.

Affine transformasjoner

- Eksempler på affine transformasjoner:
 - Translasjon (forflytning)
 - Rotasjon
 - "Shearing"
 - Refleksjon
 - Skalering
 - **Kombinasjoner av disse (!)**

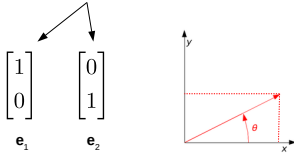
Eksempler på (enkle) transformasjoner

Transformasjon	a_0	a_1	a_2	b_0	b_1	b_2	Uttrykk
Translasjon med Δx og Δy	1	0	Δx	0	1	Δy	$x' = x + \Delta x$ $y' = y + \Delta y$
Skalering med faktor s	s	0	0	0	s	0	$x' = sx$ $y' = sy$
Rotasjon med θ	$\cos\theta$	$-\sin\theta$	0	$\sin\theta$	$\cos\theta$	0	$x' = \cos\theta x - \sin\theta y$ $y' = \sin\theta x + \cos\theta y$
Horizontal "shear" med faktor s	1	s	0	0	1	0	$x' = x + sy$ $y' = y$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_0 & a_1 & a_2 \\ b_0 & b_1 & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Litt mer om rotasjon

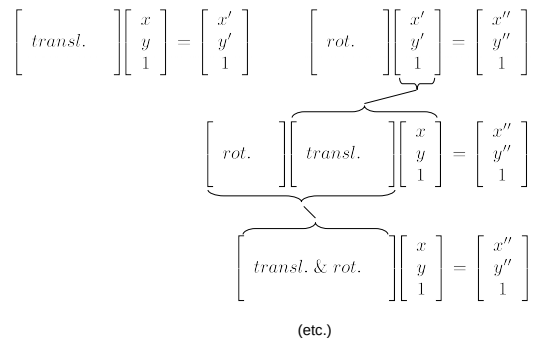
Hva vil vi skal skje med hver av disse basisvektorene ved en rotasjon med θ ?



$$R(\theta) = [T_1(e_1) \quad T_1(e_2)]$$

$$R(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

Sammenslåing av affine transformasjoner



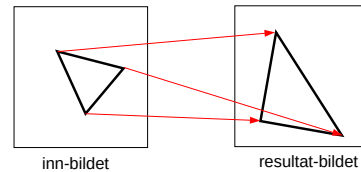
Egenskaper ved affine transformasjoner

- Rette linjer bevares
- Parallele linjer forblir parallelle
- Lineær transform med påfølgende translasjon
- Kan uttrykkes ved enkel matrisemultiplikasjon
 - Og følgelig svært enkle generelle inverse

Jfr. resampling som vi straks kommer til

Alternativ måte å finne transformkoeffisientene

- En affinn transform kan bestemmes ved å spesifisere tre punkter før og etter transformasjonen



- Med disse tre punktparene kan vi finne de 6 koeffisientene; $a_0, a_1, a_2, b_0, b_1, b_2$
- Med flere enn 3 punktpar velger man den transformasjonen som minimerer (f.eks. kvadrat-)feilen summert over alle punktene (mer om dette senere)

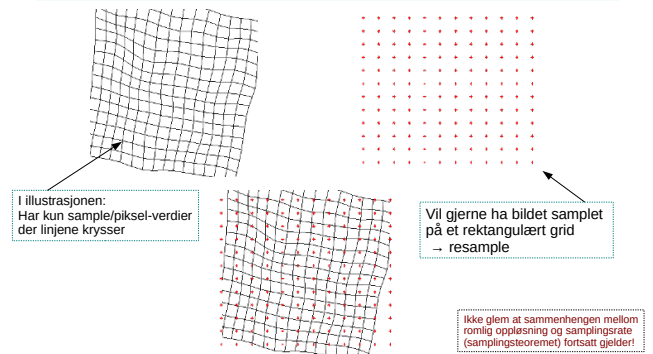
Transformer med høyere ordens polynomer

- Bilineære transformasjoner beskrives ved:

$$\begin{aligned} x' &= a_0x + a_1y + a_2 + a_3xy \\ y' &= b_0x + b_1y + b_2 + b_3xy \end{aligned}$$
- Kvadratiske transformasjoner:

$$\begin{aligned} x' &= a_0x + a_1y + a_2 + a_3xy + a_4x^2 + a_5y^2 \\ y' &= b_0x + b_1y + b_2 + b_3xy + b_4x^2 + b_5y^2 \end{aligned}$$
- Polynomer av høyere orden gir muligheter for å korrigere for mer «komplekse» avbildningsfeil
- Koeffisientene bestemmes ofte ved bruk av punktpar/kontrollpunkter (som vi kommer tilbake til senere i dag)

Resampling



Ikke glem at sammenhengen mellom romlig oppløsning og samplingsrate (samplingsteoremet) fortsatt gjelder!

En noe naiv fremgangsmåte: Forlengings-mapping

```

for all x',y' do g(x',y')=0

a0 = cos θ
a1 = -sin θ
b0 = sin θ
b1 = cos θ

for all x,y do
  x' = round(a0x+a1y)
  y' = round(b0x+b1y)
  if (x',y') inside g
    g(x',y') = f(x,y)
  end
end
    
```

Eksempel:

Enkel rotasjon ved transformen:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Flytter de posisjonstransformerte pikselverdiene til nærmeste pikselposisjon i utbildet (antar heltallige indekser)

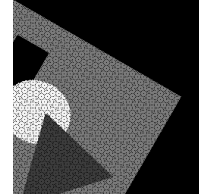
Skriver innbildets f(x,y) inn i g(x', y')

2019.01.30

INF2310

13 / 30

Forlengings-mapping, forts.



Problemer:

- Ikke alle utpikser får verdi (hullene i bildet)
- Unødig beregning av pikselkoordinater som allikevel ikke blir synlige (ender utenfor utbildet)
- Samme utbilde-piksel kan bli satt flere ganger

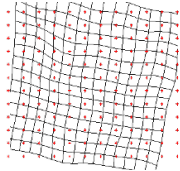
2019.01.30

INF2310

14 / 30

«Baklengs-mapping»

- Løp igjennom «utbilde»-pikselposisjonene (røde prikker på figuren) og finn ut hvilke verdier vi har der
- Resampling ved inverstransform



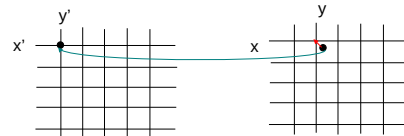
2019.01.30

INF2310

15 / 30

Interpolasjon – hvilke intensitetsverdier har bildet mellom sample-punktene?

Baklengs-mapping



Enkleste løsning:

Nullte-ordens interpolasjon eller nærmeste nabo-interpolasjon

$$g(x',y') = f(\text{round}(x), \text{round}(y))$$

[Intensiteten til g blir da ALLTID en av verdiene til f]

2019.01.30

INF2310

16 / 30

Baklengs-mapping med nærmeste-nabo-interpolasjon

```

a0 = cos (-θ)
a1 = -sin (-θ)
b0 = sin (-θ)
b1 = cos (-θ)

for alle x',y' do
  x = round(a0x'+a1y')
  y = round(b0x'+b1y')
  if (x,y) inside f
    g(x',y') = f(x,y)
  else
    g(x',y')=0
  end
end
    
```

Resample bildet. Her; for hvert utbilde-piksel, invers-transformér, og velg nærmeste piksel fra innbildet.

Samme eksempel som ved forlengings-mappingen.

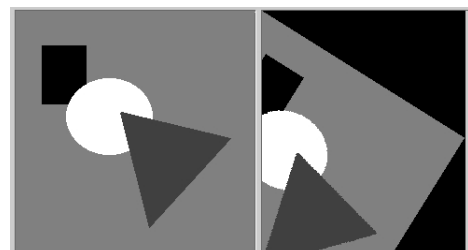
Husk: Hvis (x,y) roteres med θ og gir (x', y'), tilsvarer det at hvis (x', y') roteres med -θ får vi (x,y). [Kan selvfølgelig også bare inverttere transformmatrisen for å få koeffisientene.]

2019.01.30

INF2310

17 / 30

Baklengs-mapping, forts.



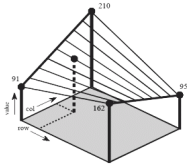
2019.01.30

INF2310

18 / 30

Første-ordens interpolasjon/ bilineær interpolasjon

- Intensiteten blir en kombinasjon av pikselverdiene i de fire pikslene som omgir punktet
- Bidragene fra hver av disse vektet med avstanden
- Interpolere i x- og y-intervallene mellom 0 og 1:
 $f(x, y) \approx f(0, 0)(1-x)(1-y) + f(1, 0)x(1-y) + f(0, 1)(1-x)y + f(1, 1)xy$



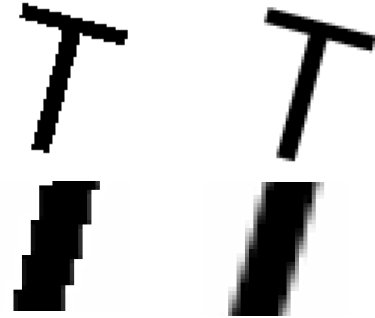
Praktisk algoritme:
 $x_0 = \text{floor}(x)$, $y_0 = \text{floor}(y)$
 $x_1 = \text{ceil}(x)$, $y_1 = \text{ceil}(y)$
 $\Delta x = x - x_0$
 $\Delta y = y - y_0$
 $p = f(x_0, y_0) + [f(x_1, y_0) - f(x_0, y_0)] \Delta x$
 $q = f(x_0, y_1) + [f(x_1, y_1) - f(x_0, y_1)] \Delta x$
 $f(x', y') = p + (q-p)\Delta y$

2019.01.30

INF2310

19 / 30

Bilineær interpolasjon, eksempel



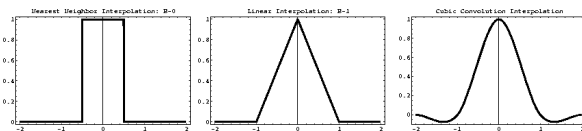
2019.01.30

INF2310

20 / 30

Høyere-ordens interpolasjon

- Kubisk interpolasjon benytter et naboskap på 4x4 piksler
- Interpolasjon kan sees på som (kontinuerlig) konvolusjon med bestemte filtre



(1D-varianter av nærmeste nabo, lineær og kubisk interpolasjonskjerne)

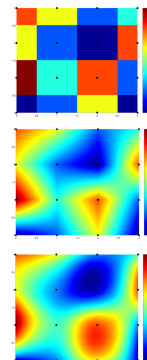
2019.01.30

INF2310

21 / 30

Interpolasjon – en sammenligning

- Nærmeste nabo gir 2D trappefunksjon, med diskontinuitet midt mellom punktene
- Bi-lineær interpolasjon bruker $2 \times 2 = 4$ piksler. Derivert er ikke kontinuerlig over flaten
- Bi-kubisk interpolasjon gir glattere flater enn bilineær, men er mer regnekrevende; bruker $4 \times 4 = 16$ piksler



2019.01.30

INF2310

22 / 30

Interpolasjonsfunksjoner i praksis

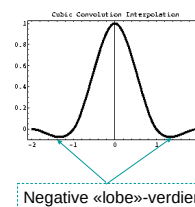
- Nærmeste nabo:
 - Taggete kanter (ikke kontinuerlige ut-bilder)
 - Hver ut-piksel har en verdi fra inn-bildet:
 - Ingen rekvantisering nødvendig, og en fordel hvis man vil bevare visse statistiske egenskaper i bildet (eller hvis bildet er segmentert i ulike klasser)
- Bilineær:
 - Kontinuerlige ut-bilder
 - Ofte visuelt mer behagelige enn nærmeste nabo
 - Noe mer regnekrevende
- Høyere-ordens interpolasjon (f.eks. bikubisk):
 - Kontinuerlige deriverte av ønsket orden
 - (Betydelig) mer regnekrevende
 - Kan gi opphav til «kant-klorie-effekter»

2019.01.30

INF2310

23 / 30

"Kant-klorie-effekter" / "ringing" ved kubisk interpolasjon

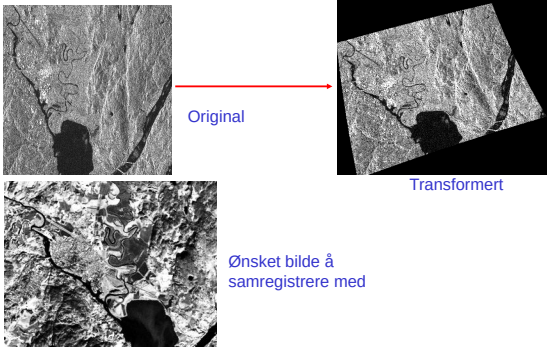


2019.01.30

INF2310

24 / 30

Bruk av geometriske transformasjoner: Samregistrering av bilder



Samregistrering II

- Om bildenes kartkoordinater er kjent kan disse benyttes til å finne transformkoeffisientene
- Hvis ikke, brukes gjerne kontrollpunkter:
 - Kontrollpunkter plukkes ut manuelt - lett identifiserbare punkter (landemerker) i begge bildene
 - Affine transformasjoner er unikt spesifisert med 3 punktpar (bestemmer $a_0, a_1, a_2, b_0, b_1, b_2$), bilineære med 4 punktpar (bi-kvadratiske med 6, etc.)
 - I praksis velges ofte mange flere punkter for å få en god transformasjon (se neste side)

Samregistrering III

- Ved flere kontrollpunkter enn nødvendig for å bestemme transformkoeffisientene, benyttes ofte summen av punktparenes kvadrerte feil som minimeringskriterium
- Gitt M kontrollpunkter $(x_i, y_i), (x'_i, y'_i)$ («r» indikerer referansebildet) og anta mappingen $(x, y) \rightarrow (x', y')$
- Polynomkoeffisientene settes til de som minimerer kvadratfeilen mellom kontrollpunktets sanne koordinater (x'_i, y'_i) og de transformerte koordinatene (x_i, y_i) :

$$J = \sum_{i=1}^M (x'_i - x_i)^2 + (y'_i - y_i)^2$$
- "Enkel" lineær algebra benyttes til å finne eksakt løsning

Kursorisk pensum

Samregistrering IV (Minimere kvadratfeilen)

$$J = \sum_{i=1}^M (x'_i - x_i)^2 + (y'_i - y_i)^2 = J_x + J_y \quad J_x = \sum_{i=1}^M (x'_i - x_i)^2$$

G og a er her basert på en affin transform

$$\begin{bmatrix} d \\ x'_1 \\ x'_2 \\ \vdots \\ x'_M \end{bmatrix} \quad \begin{bmatrix} G \\ x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_M & y_M & 1 \end{bmatrix} \quad \begin{bmatrix} a \\ a_0 \\ a_1 \\ a_2 \end{bmatrix}$$

Vi er på jakt etter koeffisientene i denne a-vektoren

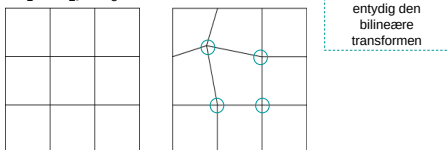
$$J_x = (d - Ga)^T (d - Ga) = d^T d + a^T G^T G a - 2a^T G^T d$$

$$\frac{\partial J_x}{\partial a^T} = 2G^T G a - 2G^T d = 0 \quad \Rightarrow \quad a = (G^T G)^{-1} G^T d$$

En enkelt matrise

Stykkevis transformasjoner

- Forskjellige transformasjoner for ulike deler av bildet
- Ofte bestemmes et kontrollgrid som styrer hvordan de ulike delene skal endres
- Bilineær transformasjon benyttes ofte:
 - $x' = a_0 x y + a_1 x + a_2 y + a_3$
 - $y' = b_0 x y + b_1 x + b_2 y + b_3$



Oppsummering

- Transform/ending av pikselenes posisjoner (x-, y-koordinater)
 - Fokus på affine transformasjoner
- Resampling
 - Forlengs- og baklengsmapping
 - Interpolasjonsmetoder
 - Nærmeste nabo-interpolasjon
 - Bilineær interpolasjon
 - Høyere-ordens interpolasjoner
- Bruk av geometriske operasjoner til å samregistrere bilder
 - Kontrollpunkter

Husk: Sammenhengen mellom romlig oppløsning og samplingsrate (samplingsteoremet) gjelder fortsatt!