INF 2310 – Digital bildebehandling

Forelesning 3

Geometriske operasjoner

Fritz Albregtsen

Temaer i dag

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

- Pensum: Kap. 2.4.4 og 2.6.5 i DIP

Geometriske operasjoner

- Endrer på pikslenes posisjoner
- Første steg i denne prosessen:
  - Transformer pikselkoordinatene (x,y) til (x',y'):

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

  - T_x og T_y er ofte gitt som polynomer.
- Siden pikselkoordinatene må være heltall, må vi deretter bruke interpolasjon til å finne pikselverdien (gråtonen) i den nye posisjonen.

Anvendelser

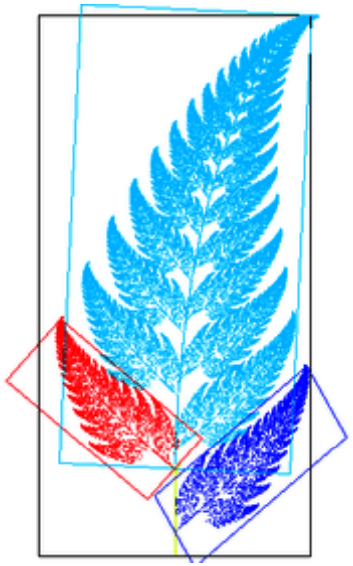
- Forstørre deler av bilder for visuell inspeksjon («zoome»)
- Rette opp geometriske feil som oppstår under avbildningen
  - Rotasjon i bildeplanet
  - Fiskeøyelinse
  - Radaravbildning av terreng
  - Medisinsk ultralyd
  - …
- Samregistrere bilder
  - Samregistrere bilder fra ulike sensorer (CT, MR, US)
  - Samregistrere bilder tatt på ulike tidspunkt.
  - Samregistrere bilder med kart i en bestemt kartprojeksjon.
  - Eksempel: Ansiktsgjenkjenning: Finne alle ansiktene i et bilde, ”klipp” ut ansiktene, transformer del-bildene slik at ansiktene i bildet blir på samme sted, har samme orientering og samme størrelse som i referansebildene.
- Generere bilder fra andre kameravinkler
- Spesialeffekter

Affine transformer

- Transformerer pikselkoordinatene (x,y) til (x′,y′):

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

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



- Affine transformer beskrives ved:

$$x' = a_0x + a_1y + a_2$$
$$y' = b_0x + b_1y + b_2$$

- På matriseform:

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

Egenskaper ved affine transformer

- Rette linjer bevares (se ukeoppgave)
- Parallelle linjer forblir parallelle
- Utrykkes ved enkel matrisemultiplikasjon

- Eksempler på affine transformasjoner:
  - Translasjon
  - Rotasjon
  - ”Shearing”
  - Skalering
  - Kombinasjoner av disse (!)

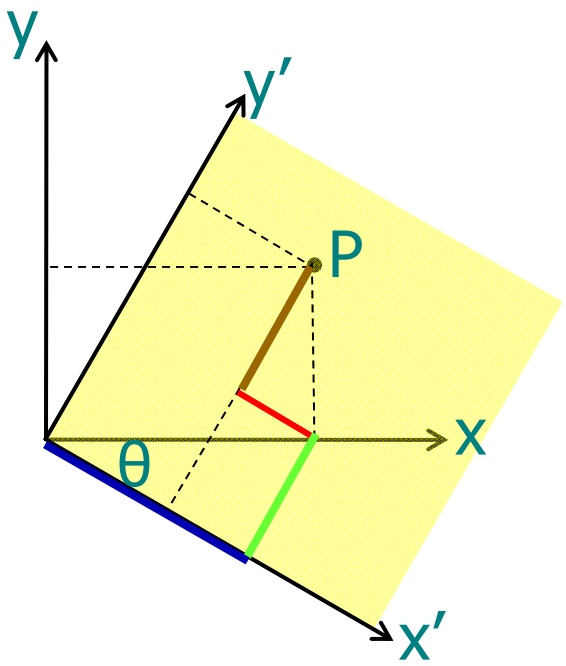
Litt om rotasjon …

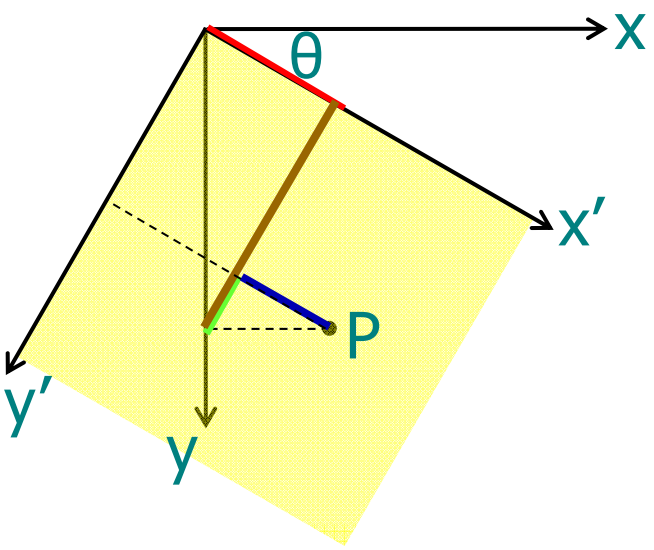
- Høyrehånds koordinatsystem, rotasjon positiv med urviseren:

  $x' = x\cos\theta - y\sin\theta$; $y' = x\sin\theta + y\cos\theta$

- Venstrehånds koordinatsystem, rotasjon positiv med urviseren:

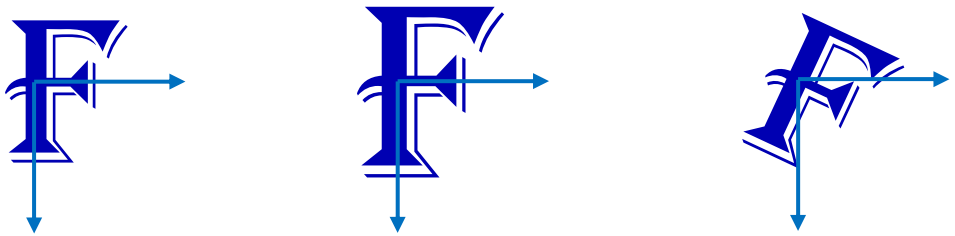
  $x' = x\cos\theta + y\sin\theta$; $y' = -x\sin\theta + y\cos\theta$





Eksempler på enkle transformer - I

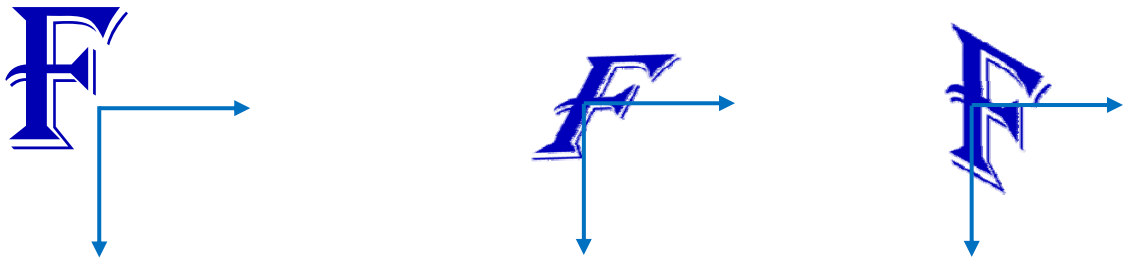
Transformasjon	a_0	a_1	a_2	b_0	b_1	b_2	Uttrykk
Identitet	1	0	0	0	1	0	$x' = x$ $y' = y$
Skalering	s_1	0	0	0	s_2	0	$x' = s_1x$ $y' = s_2y$
Rotasjon	cosθ	-sinθ	0	sinθ	cosθ	0	$x' = x\cos\theta - y\sin\theta$ $y' = x\sin\theta + y\cos\theta$



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

Eksempler på enkle transformer - II

Transformasjon	a_0	a_1	a_2	b_0	b_1	b_2	Uttrykk
Translasjon	1	0	Δx	0	1	Δy	$x' = x+\Delta x$ $y' = y+\Delta y$
Horisontal "shear" med faktor s_1	1	s_1	0	0	1	0	$x' = x+s_1 y$ $y' = y$
Vertikal "shear" med faktor s_2	1	0	0	s2	1	0	$x' = x$ $y' = s_2 x+y$



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

Sammenslåing av affine transformer

$$\begin{bmatrix} \\ transl. \\ \\ \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \qquad \begin{bmatrix} \\ rot. \\ \\ \end{bmatrix}\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix}$$

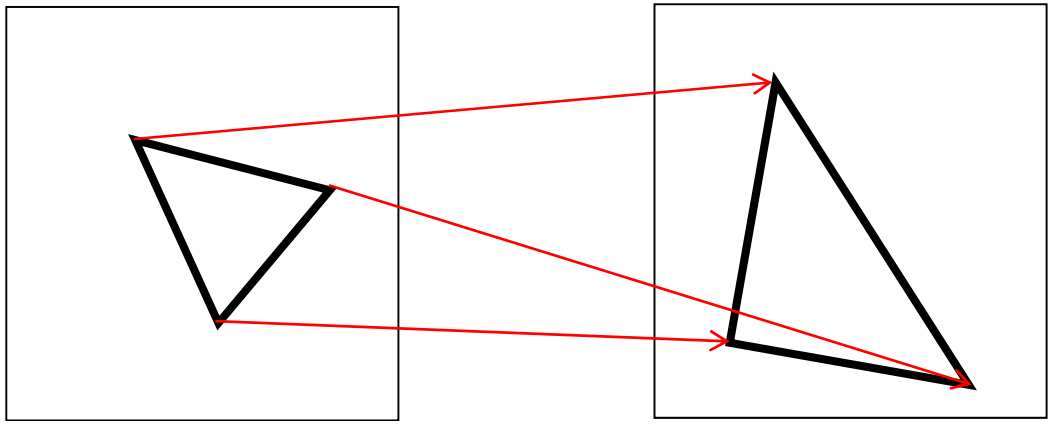
$$\begin{bmatrix} \\ rot. \\ \\ \end{bmatrix}\begin{bmatrix} \\ transl. \\ \\ \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} \\ transl.\ \&\ rot. \\ \\ \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix}$$

(etc.)

Alternativ måte å finne transformkoeffisientene

- En affin transform kan bestemmes ved å spesifisere tre punkter før og etter avbildningen



- 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 (kvadrat-)feilen summert over alle punktene (mer om dette senere)

Transformer med høyere ordens polynomer

- Bilineære transformer beskrives ved:

$$x' = a_0x + a_1y + a_2 + a_3xy$$

$$y' = b_0x + b_1y + b_2 + b_3xy$$

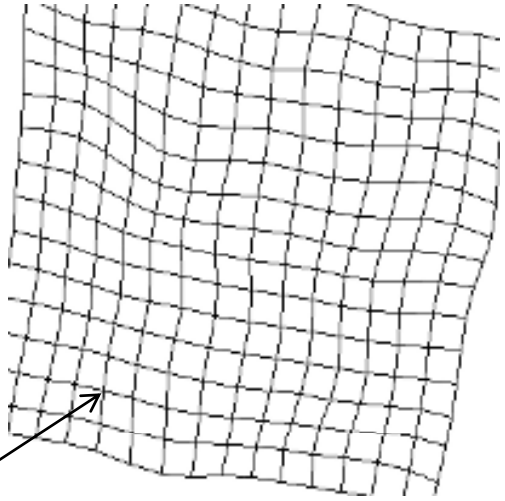
- Kvadratiske transformer:

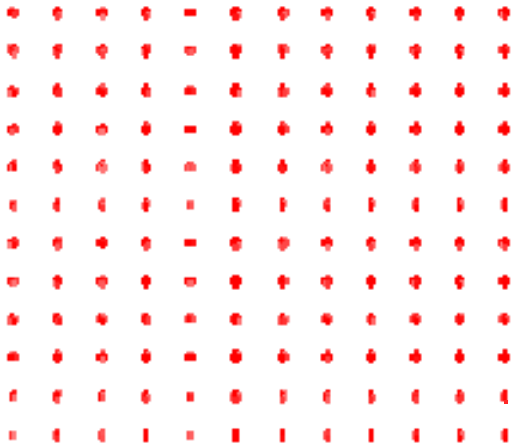
$$x' = a_0x + a_1y + a_2 + a_3xy + a_4x^2 + a_{5y}y^2$$

$$y' = b_0x + b_1y + b_2 + b_3xy + b_4x^2 + b_{5y}y^2$$

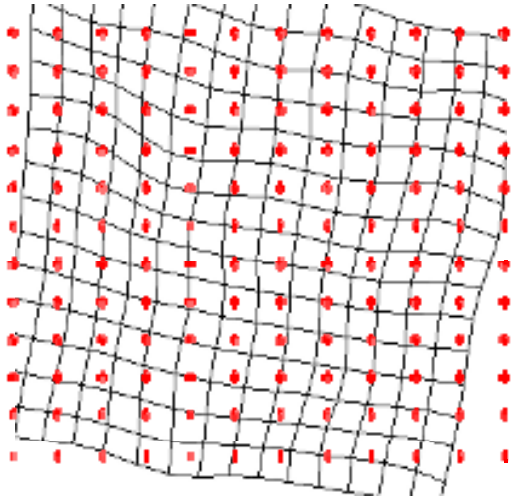
- Polynomer av høyere orden gir muligheter for å korrigere for mer «komplekse» avbildningsfeil.

Resampling





I illustrasjonen:
Har kun sample/piksel-verdier
der linjene krysser



Vil gjerne ha bildet samplet
på et rektangulært grid
-> resample

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

Forlengs-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
```

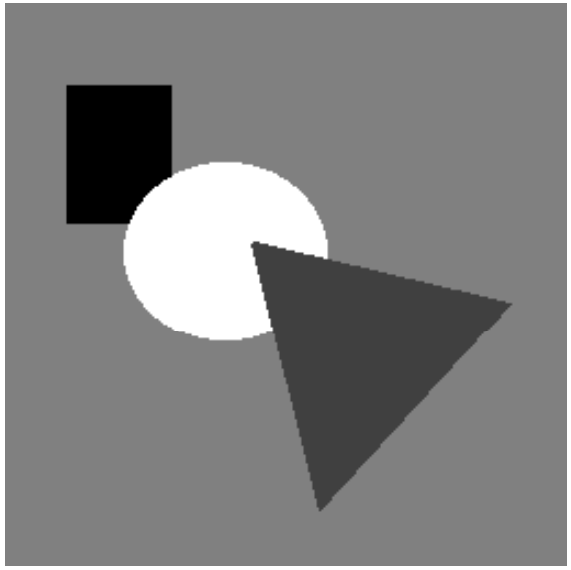
Eksempel:
Enkel rotasjon ved transformen:

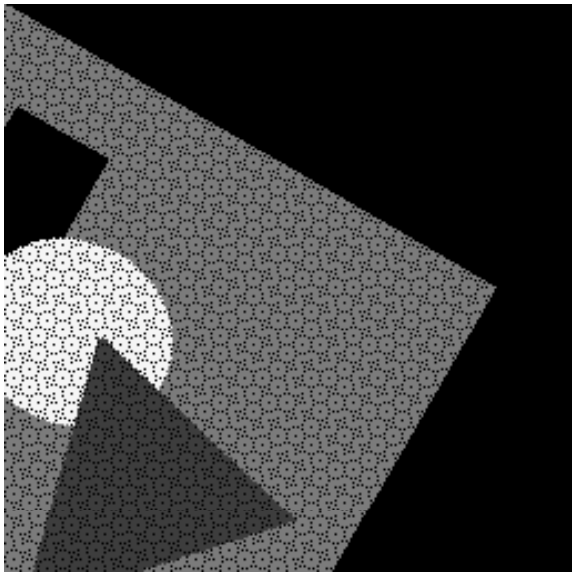
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Flytter de posisjonstransformerte pikselposisjonene til nærmeste pikselposisjon i utbildet.

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

Forlengs-mapping, forts.





Problemer:

- Ikke alle utpiksler 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

Baklengs-mapping

$a_0 = \cos(-\theta)$
$a_1 = -\sin(-\theta)$
$b_0 = \sin(-\theta)$
$b_1 = \cos(-\theta)$

```
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
```

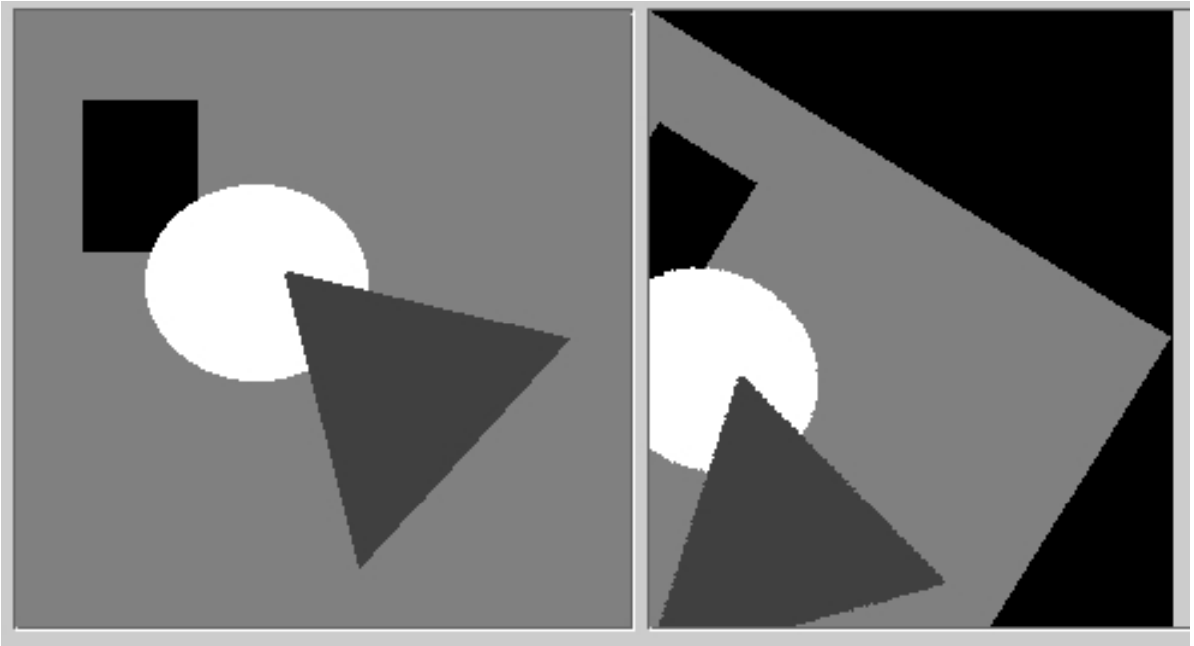
Samme eksempel som ved forlengs-mappingen.

NB: (x,y) rotert med θ ga (x′,y′)
(x′,y′) rotert med -θ gir (x,y)

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

For hver pikselposisjon i ut-bildet:
Hent pikselverdi fra innbildet.

Baklengs-mapping, forts.



GARANTI:

• Alle utpiksler får verdi (ingen hull i ut-bildet).

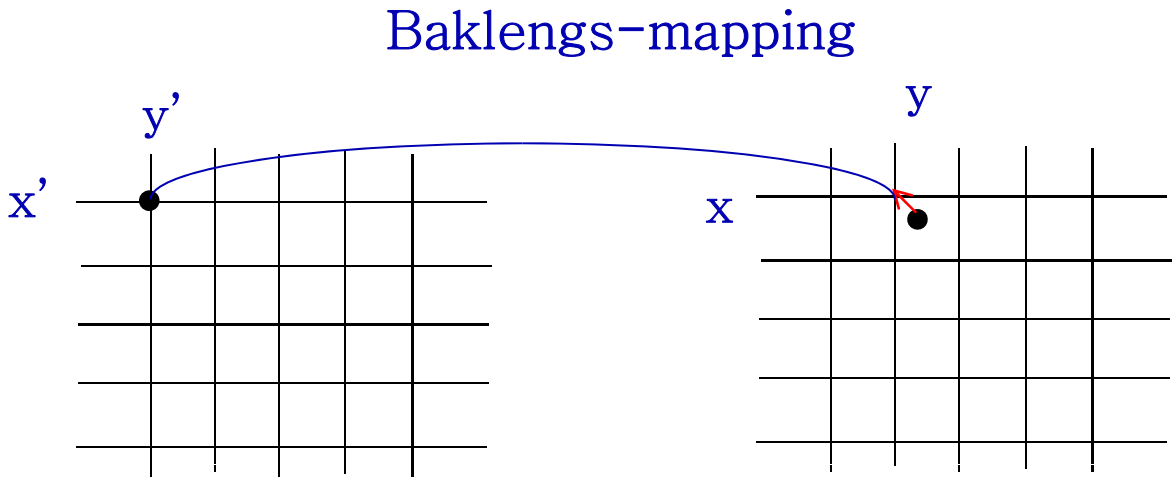
• Ingen unødig beregning av pikselkoordinater som allikevel ikke blir synlige (ender utenfor utbildet).

• Ingen utbilde-piksel vil bli satt flere ganger.

Og så : INTERPOLASJON

- Vi har utført den geometriske operasjonen ”baklengs mapping”, som er en re-sampling.
- For hvert piksel i utbildet har vi valgt pikselverdien fra nærmeste piksel i innbildet, gitt transformen.
- Er dette det beste vi kan få til?
- Eller finnes det metoder som gir et bedre resultat?

Interpolasjon :
Hvilken intensitetsverdi skal pikselen få?



Nullte-ordens interpolasjon eller nærmeste nabo-interpolasjon
g(x',y') = f(round(x) , round(y))

=> Intensiteten til g(x',y') blir en av verdiene fra f

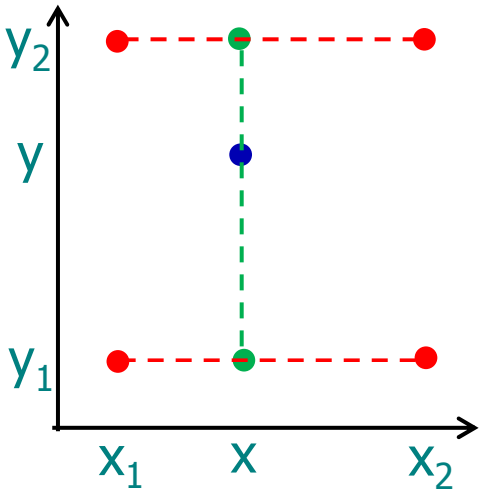
Bilineær interpolasjon - I

- Anta at vi kjenner gråtoneverdien i fire nabo-punkter
- Ønsker å estimere gråtonen i et mellomliggende punkt.
- Gjør to lineære interpolasjoner i én retning først, f.eks i x-retning:

$$f(x, y_1) \approx \frac{(x_2 - x)}{(x_2 - x_1)} f(x_1, y_1) + \frac{(x - x_1)}{(x_2 - x_1)} f(x_2, y_1)$$

og

$$f(x, y_2) \approx \frac{(x_2 - x)}{(x_2 - x_1)} f(x_1, y_2) + \frac{(x - x_1)}{(x_2 - x_1)} f(x_2, y_2)$$



- Så én interpolasjon i ortogonal retning:

$$f(x, y) \approx \frac{(y_2 - y)}{(y_2 - y_1)} f(x, y_1) + \frac{(y - y_1)}{(y_2 - y_1)} f(x, y_2)$$

- Resultatet er uavhengig av rekkefølgen.
- Den interpolerte flaten er kvadratisk (krum), men lineær langs linjer parallelle med aksene.

Bilineær interpolasjon - II

- Bilineær interpolasjon når f er kjent i (0,0), (0,1), (1,0) og (1,1):

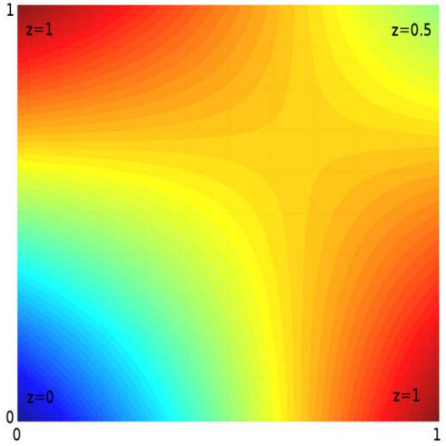
$$f(x,y) \approx (1-y) f(x,0) + y\, f(x,1)$$

der

$$f(x,0) \approx (1-x)\, f(0,0) + x\, f(1,0)$$

og

$$f(x,1) \approx (1-x)\, f(0,1) + x\, f(1,1)$$



$$\Rightarrow\ f(x,y) \approx (1-x)(1-y)\, f(0,0) + x(1-y)\, f(1,0) + (1-x)y\, f(0,1) + x\,y\, f(1,1)$$

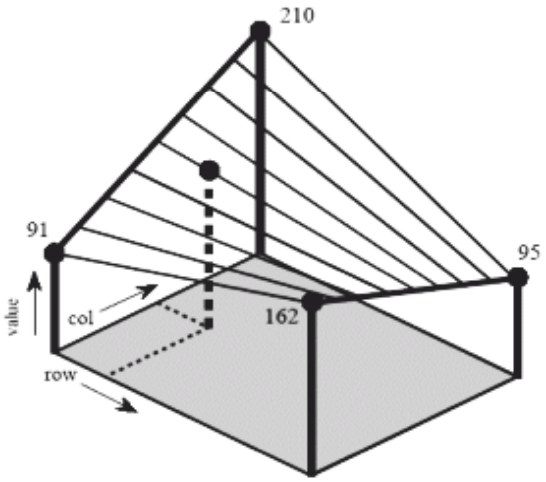
- Eller i matrisenotasjon:

$$f(x,y) \approx \begin{bmatrix}(1-x) & x\end{bmatrix}\begin{bmatrix} f(0,0) & f(0,1) \\ f(1,0) & f(1,1)\end{bmatrix}\begin{bmatrix}(1-y) \\ y\end{bmatrix}$$

Første-ordens interpolasjon/ bilineær interpolasjon

- Intensiteten blir en kombinasjon av pikselverdiene i de fire pikslene som omgir punktet
- Bidragene fra hver av disse vektes 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 = floor(x), y_0 = floor(y)

x_1 = ceil(x), y_1 = ceil(y)

$\Delta x = x - x_0$

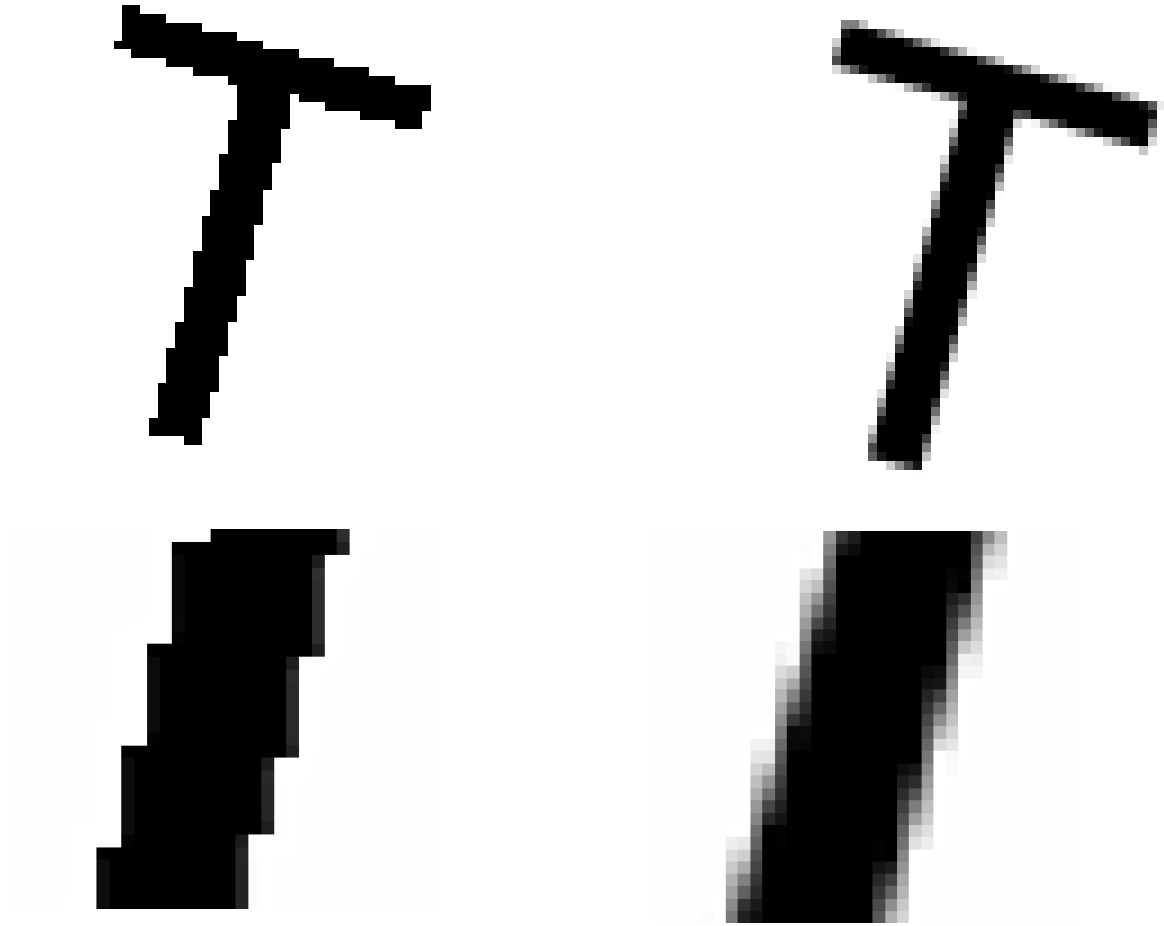
$\Delta y = y - y_0$

$p = f(x_0,y_0)+[f(x_1,y_0)-f(x_0,y_0)]\ \Delta x$

$q = f(x_0,y_1)+[f(x_1,y_1)-f(x_0,y_1)]\ \Delta x$

$f(x',y')=p+(q-p)\Delta y$

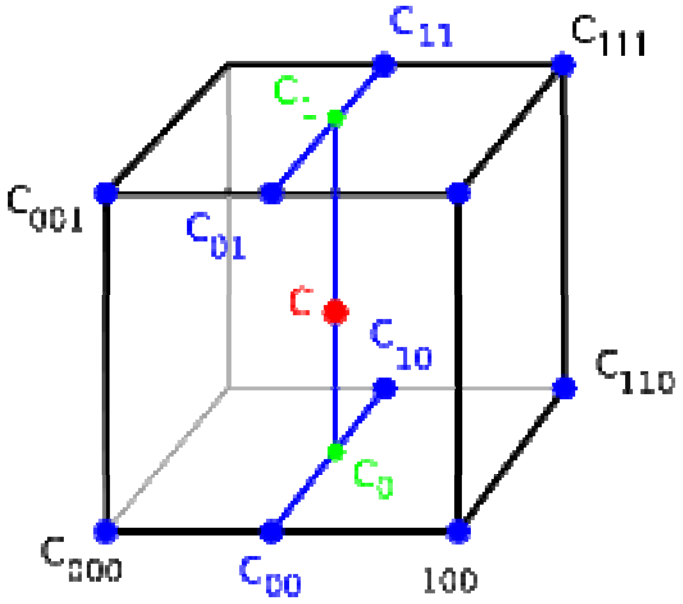
Bilineær interpolasjon, eksempel



Trilineær interpolasjon

- Utvidelsen fra 2D til 3D kalles *trilineær* interpolasjon, og er en lineær interpolasjon mellom resultatene av to bilineære interpolasjoner.

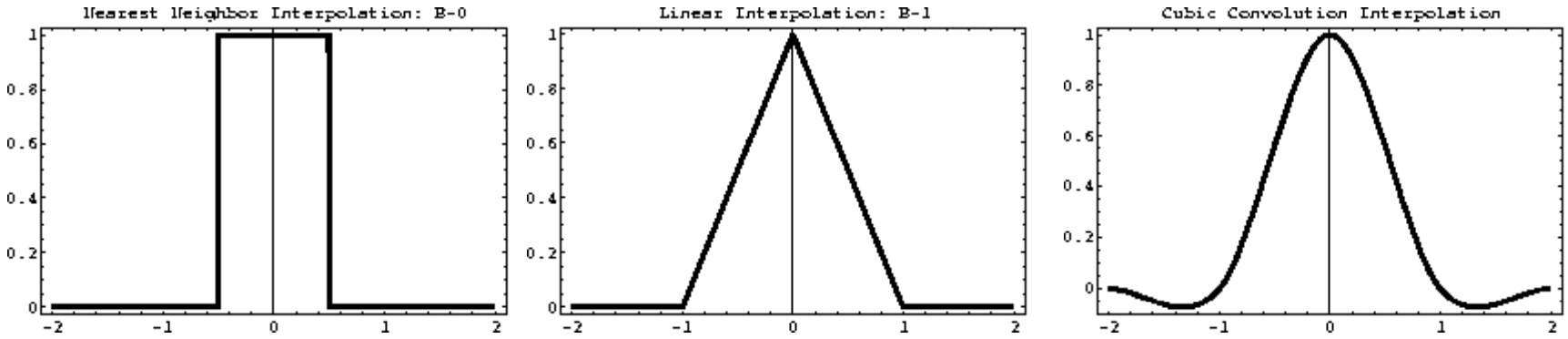
- Resultatet er igjen uavhengig av rekkefølgen.



Høyere-ordens interpolasjon

- Bi-kubisk interpolasjon benytter et naboskap på 4x4 piksler

- Interpolasjon kan sees på som (kontinuerlig) konvolusjon med bestemte filtre.

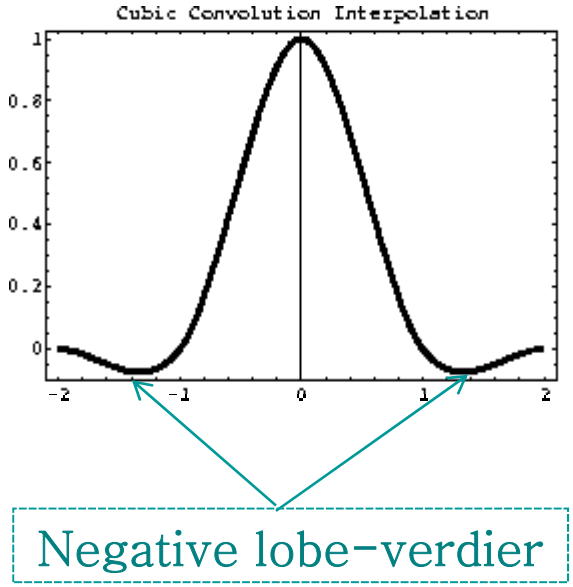


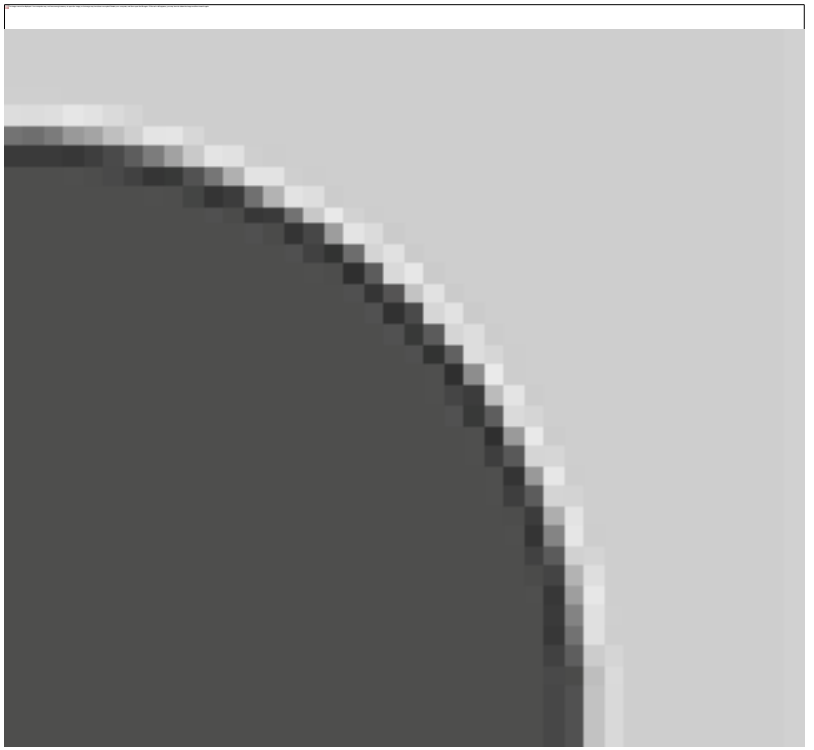
(1D-varianter av nærmeste nabo, lineær og kubisk interpolasjonskjerne)

Interpolasjonsfunksjoner i praksis

- Nærmeste nabo:
  - Taggete kanter og større totalfeil.
  - Hver ut-piksel har en verdi fra inn-bildet:
    - Ingen rekvantisering er nødvendig:
      - en fordel hvis man vil bevare visse statistiske egenskaper i bildet (eller hvis bildet er segmentert i ulike klasser)
- Bilineær interpolasjon og høyere-ordens interpolasjon:
  - er mer regnekrevende
- Bi-kubisk interpolasjon:
  - gir skarpere bilder og har kontinuerlige deriverte
  - Men er (mye) mer regnekrevende enn bilineær interpolasjon,
  - og kan gi opphav til ”kant-glorie-effekter”

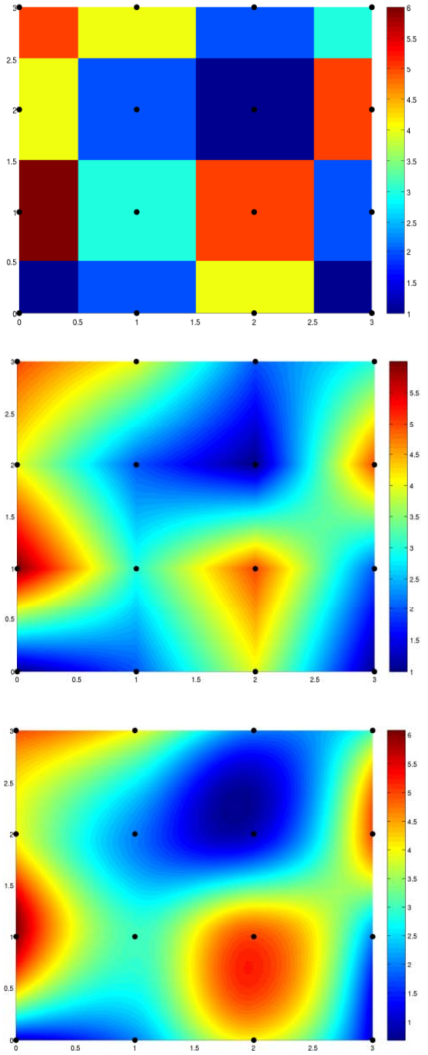
”Kant-glorie-effekter” / ”ringing” ved kubisk interpolasjon





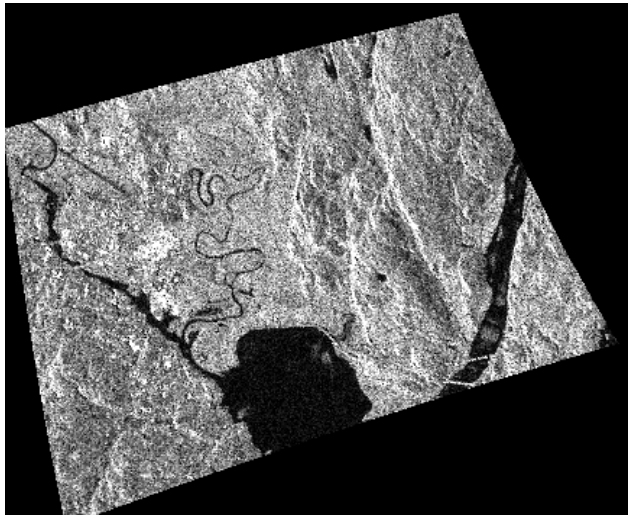
Interpolasjon – en sammenligning

- Nærmeste nabo gir 2D trappefunksjon, med diskontinuitet midt mellom punktene.
- Bi-lineær interpolasjon bruker 2x2=4 piksler. Derivert er ikke kontinuerlig over flaten.
- Bi-kubisk interpolasjon gir glattere flater enn bilineær, men er mer regnekrevende; bruker 4x4=16 piksler.



Bruk av geometriske transformer: Samregistrering av bilder

Original



Transformert

Ønsket bilde å samregistrere med

Samregistrering II

- Hvis bildenes kartkoordinater er kjent, benyttes de til å finne transformkoeffisientene.
- Hvis ikke, brukes gjerne kontrollpunkter:
  - Kontrollpunkter kan plukkes ut manuelt
    - lett identifiserbare punkter (landemerker) i begge bildene.
  - Affine transformer er unikt spesifisert med 3 punktpar
    - (bestemmer a_0, a_1, a_2, b_0, b_1, b_2)
  - Bi-lineære med 4 punktpar
  - Bi-kvadratiske med 6 punktpar
  - Bi-kubiske med 16 punktpar, …
- 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 kvadratfeilen som minimeringskriterium.

- Gitt M kontrollpunkter $(x_i,y_i),(x_i^r,y_i^r)$ («r» indikerer referansebildet) og anta mappingen (x_i,y_i) --> (x'_i,y'_i)

- Polynomkoeffisientene settes til de som minimerer kvadratfeilen mellom kontrollpunktets sanne koordinater (x_i^r,y_i^r) og de transformerte koordinatene (x_i',y_i'):

$$J = \sum_{i=1}^{M} (x_i' - x_i^r)^2 + (y_i' - y_i^r)^2$$

- "Enkel" lineæralgebra benyttes til å finne eksakt løsning.

Kursorisk pensum

Samregistrering IV (Minimere kvadratfeilen)

$$J = \sum_{i=1}^{M} (x_i' - x_i^r)^2 + (y_i' - y_i^r)^2 = J_x + J_y \qquad J_x = \sum_{i=1}^{M} (x_i' - x_i^r)^2$$

G og a er her basert på en affin transform

$$\begin{bmatrix} x_1^r \\ x_2^r \\ \vdots \\ x_M^r \end{bmatrix} \quad \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_M & y_M & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix}$$

d G a

Vi er på jakt etter koeffisientene i denne a-vektoren

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

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

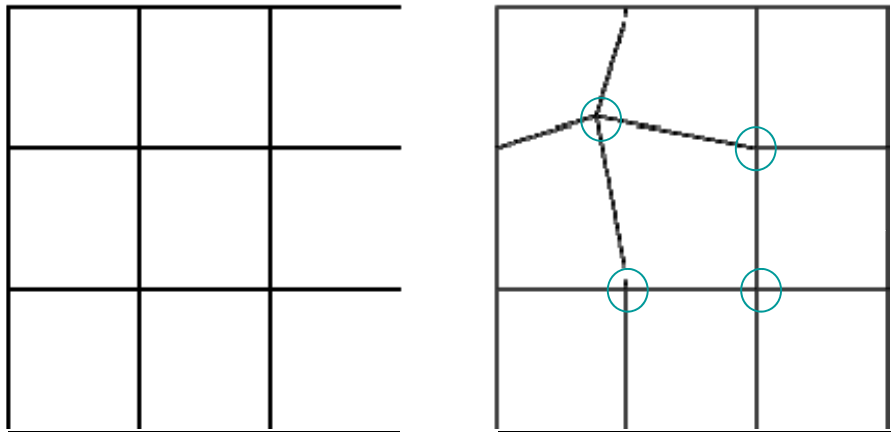
Én enkelt matrise

Stykkevise transformer

- Resampling:Forskjellige transformer for ulike deler av bildet
- Ofte bestemmes et kontrollgrid som styrer hvordan de ulike delene skal endres
- Bilineær transformasjon benyttes ofte:

$$x' = a_0xy + a_1x + a_2y + a_3$$

$$y' = b_0xy + b_1x + b_2y + b_3$$



Hver firkants fire hjørnepunkter bestemmer entydig den bilineære transformen

Oppsummering

- Transform/endring av pikslenes posisjoner (x-,y-koordinater)
  - Affine transformer
- Resampling:
  - Forlengs- og baklengsmapping
- Interpolasjonsmetoder for å bestemme gråtonene til de geometrisk transformerte pikslene:
  - Nærmeste nabo-interpolasjon
  - Bilineær interpolasjon
  - Høyere-ordens interpolasjoner
- Bruk av geometriske operasjoner til å samregistrere bilder
  - Kontrollpunkter
  - Løs ligningssett som minimerer kvadratfeil