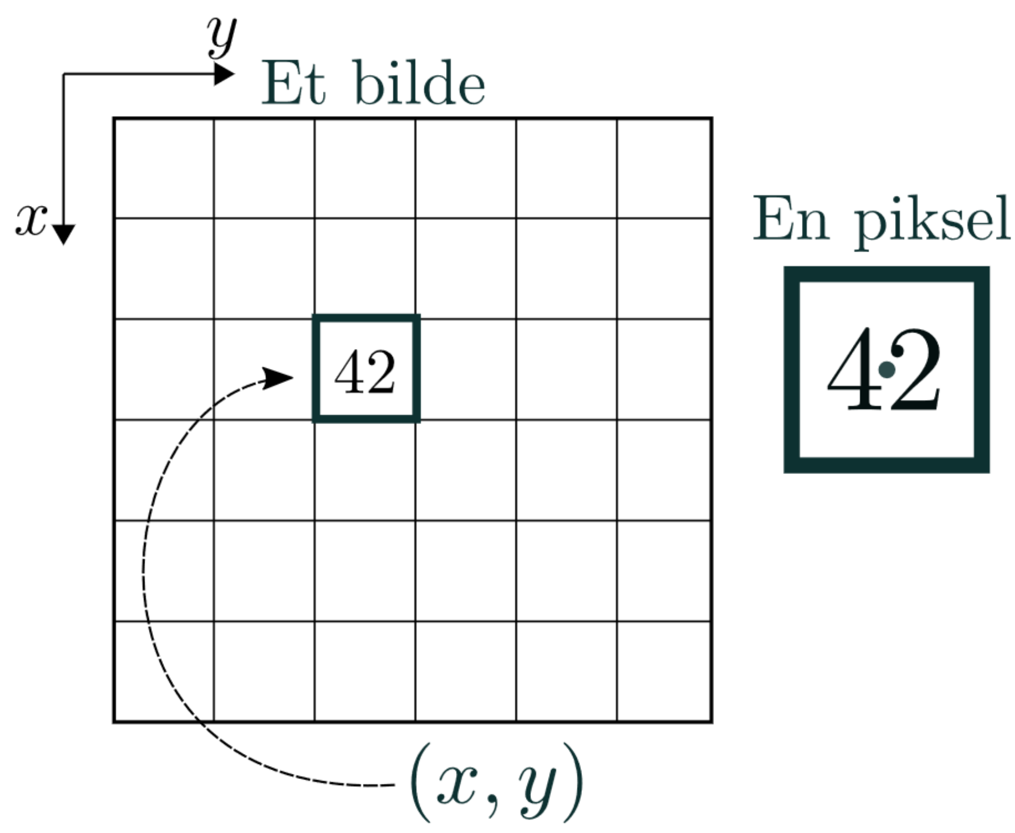
IN2070 – Februar 2022 – Ukens temaer

(Kap 2.4.4 og 2.6.5 i DIP)

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



y
Et bilde
x
42
En piksel
42
(x, y)

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.

Anvendelser

- Forstørre deler av bilder for visuell inspeksjon («zoome»)

- Rette opp geometriske feil som oppstår under avbildningen
  - Rotasjon
  - Fiskeøyelinse
  - Avbildning av terreng / ortorektifisering
  - Generell linsekorrigering
  - ...

- Samregistrere bilder
  - .. fra ulike sensorer / modaliteter (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 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:

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

Affine transformer kont.

- Eksempler på affine transformasjoner:
  - Translasjon (forflytning)
  - Rotasjon
  - ”Shearing”
  - Refleksjon
  - Skalering

Samt kombinasjoner av disse!

- Egenskaper
  - Rette linjer bevares
  - Parallelle linjer forblir parallelle
  - Kan utrykkes ved matrisemultiplikasjon

Og følgelig enkle inverse,
jfr. resampling som
vi straks kommer til

Eksempler på transformkoeffisienter

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	$x'=\cos\theta x - \sin\theta y$ $y'=\sin\theta x + \cos\theta y$
Horisontal "shear" med faktor s	1	s	0	0	1	0	$x'=x+sy$ $y'=y$

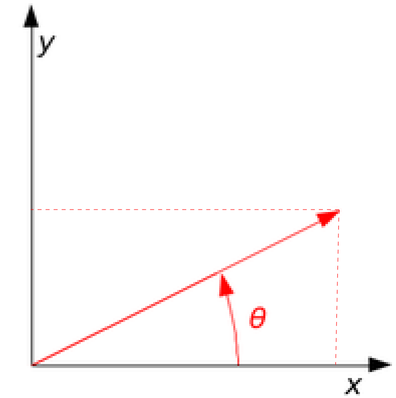
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_0 & a_1 & a_2 \\ b_0 & b_1 & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Litt mer om rotasjon

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

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$\mathbf{e}_1$ $\qquad$ $\mathbf{e}_2$



$$\boldsymbol{R}(\theta) = [\mathrm{T}_\theta(\mathbf{e}_1)\ \ \mathrm{T}_\theta(\mathbf{e}_2)]$$

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

Sammenslåing av affine transformer

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

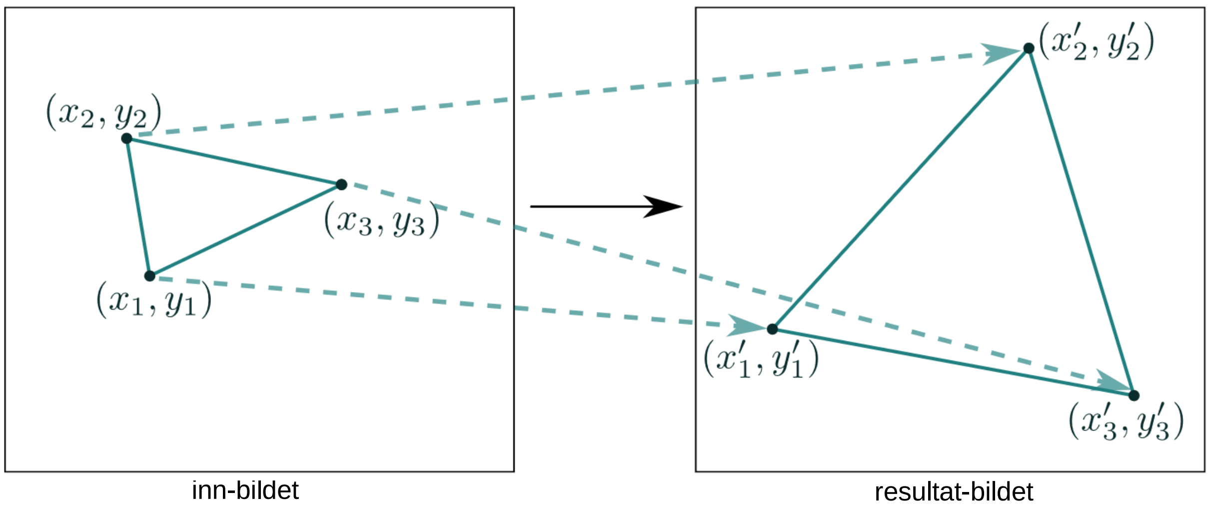
$$\begin{bmatrix} \\ rot. \\ \\ \end{bmatrix}\begin{bmatrix} \\ transl. \\ \\ \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix}$$

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

(etc.)

En annen, vanlig måte å bestemme transformkoeffisientene

- En affin transform kan bestemmes ved å spesifisere tre punkter og hvor de ønskes transformert til



- 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)

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

Transformer med høyere ordens polynomer

- Bilineære transformer beskrives ved:

$$x' = a_0x+a_1y+a_2+a_3xy$$
$$y' = b_0x+b_1y+b_2+b_3xy$$

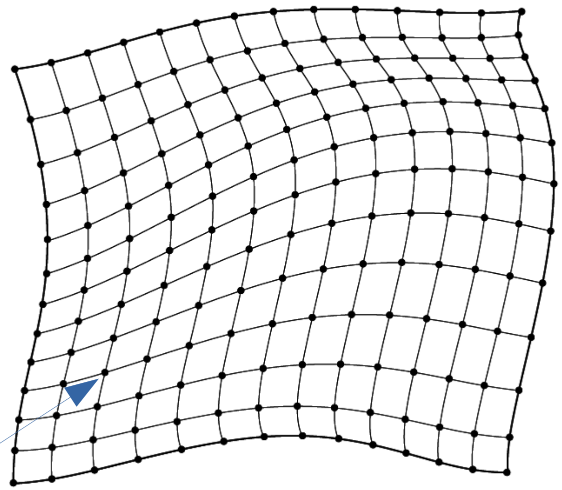
- Kvadratiske transformer:

$$x' = a_0x+a_1y+a_2+a_3xy+a_4x^2+a_{5y}y^2$$
$$y' = b_0x+b_1y+b_2+b_3xy+b_4x^2+b_{5y}y^2$$

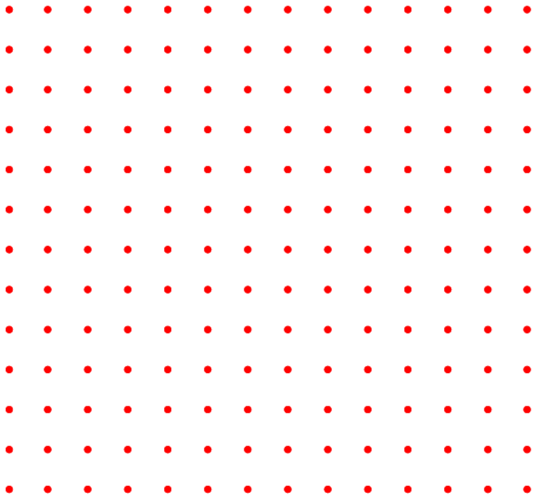
- Polynomer av høyere orden gir muligheter for å korrigere for mer «komplekse» avbildningsfeil

- Koeffisientene bestemmes ofte ved bruk av punktpar/kontrollpunkter (jfr. forrige side, dog med flere enn 6 koeffisienter)

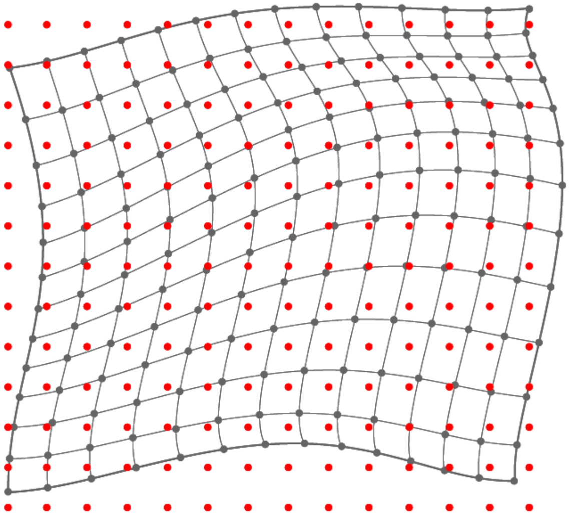
Resampling



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!

En noe naiv fremgangsmåte: Forlengs-mapping med koordinatavrunding

```
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
```

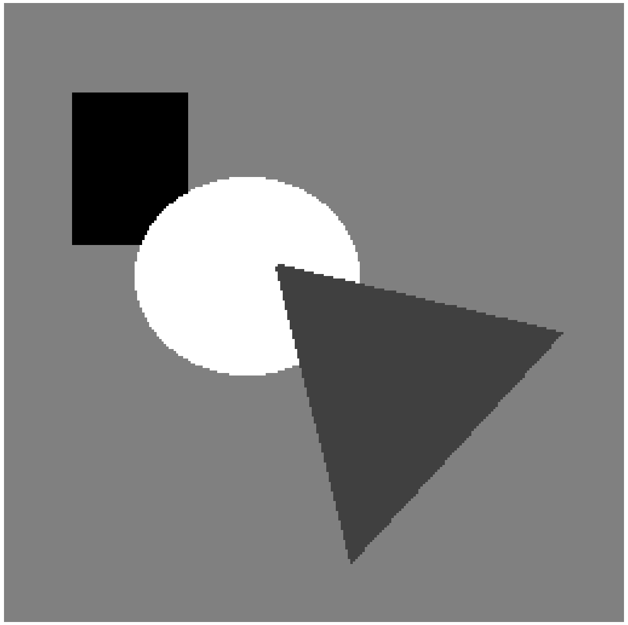
Eksempel:
Enkel rotasjon ved transformen:

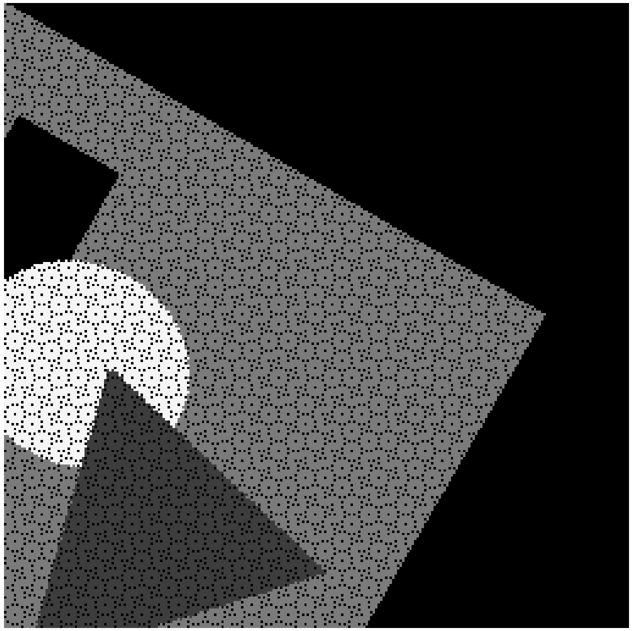
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

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

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

Forlengs-mapping, forts.





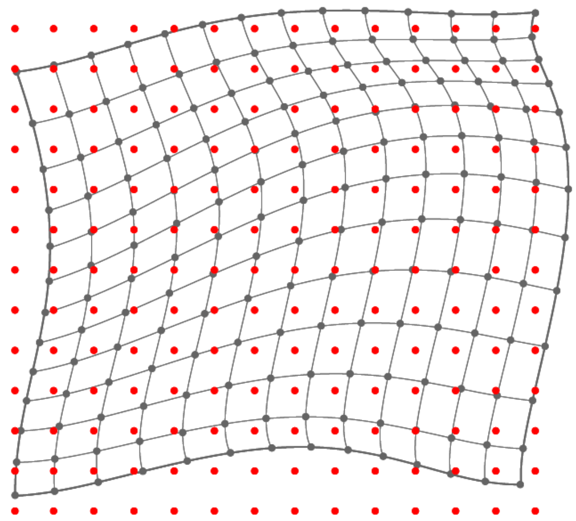
Problemer:

- 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
- (Små) geometriske unøyaktigheter

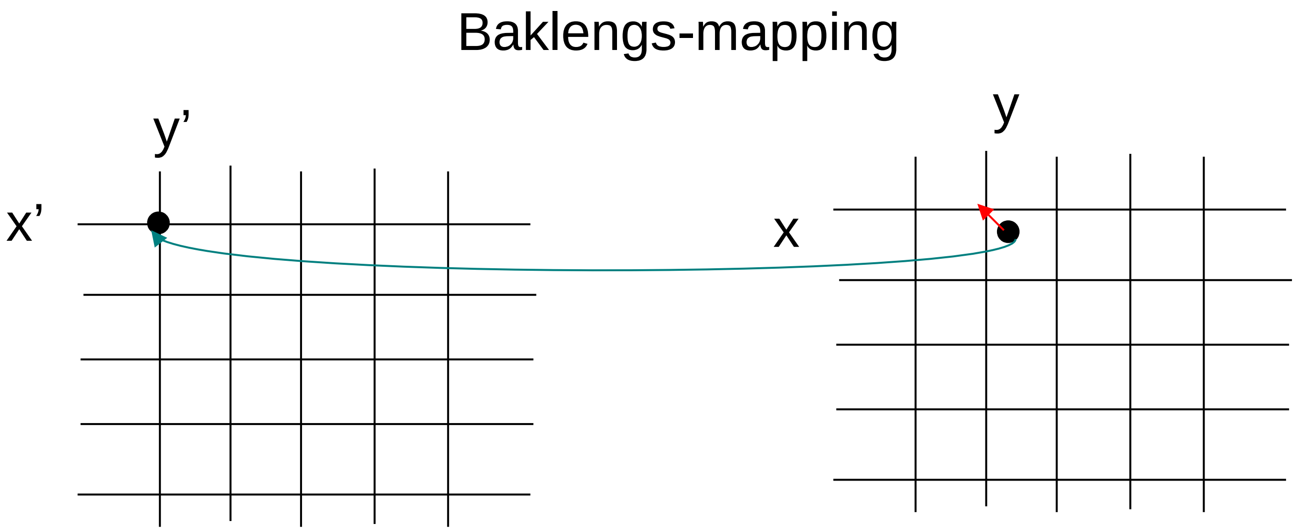
Baklengs-mapping (invers-mapping)

- Benytt den **omvendte** (inverse) transformen til å finne utbilde-koordinatenes (de røde prikkene på figuren) koordinat i **originalbildet**
- Benytt disse (originalbilde-)koordinatene til å finne pikslenes farge/intensitet

Husk at i originalbildet er pikslene sorterte og på et rektangulært grid



Interpolasjon – hvilke intensitetsverdier har bildet mellom sample-punktene?



Enkleste løsning:
Nullte-ordens interpolasjon eller nærmeste nabo-interpolasjon

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

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

Baklengs-mapping med nærmeste-nabo-interpolasjon

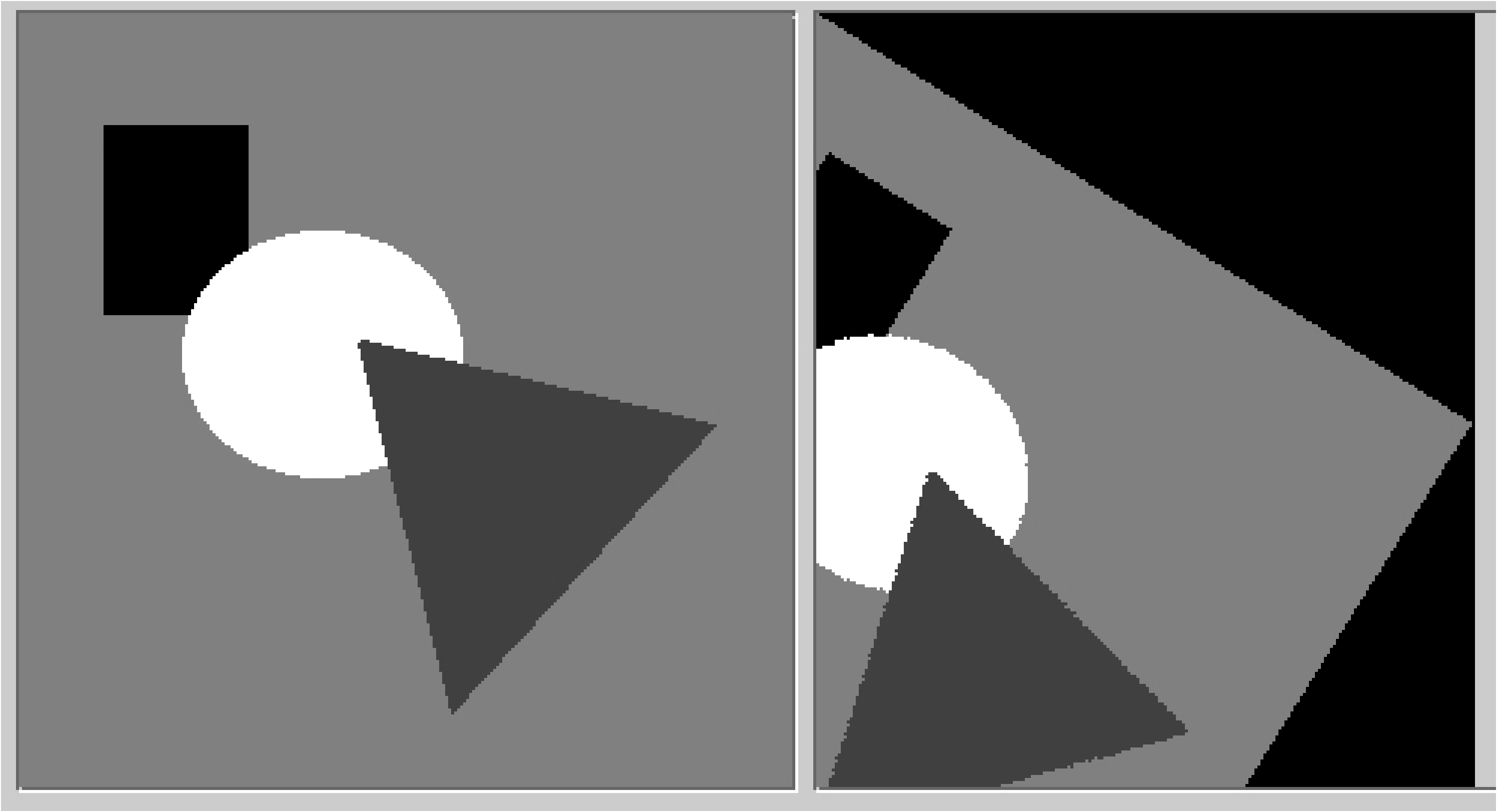
```
a0 = cos (-θ)
a1 = -sin (-θ)
b0 = sin (-θ)
b1 = cos (-θ)

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

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

Samme eksempel som ved forlengs-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 invertere transformmatrisen for å få koeffisientene.]

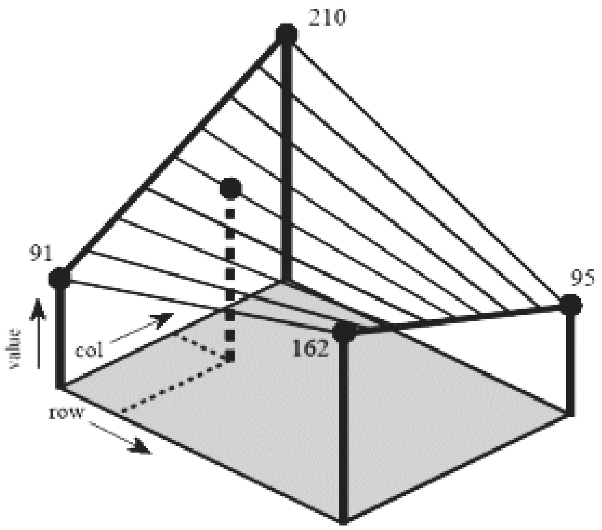
Baklengs-mapping, forts.



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:

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

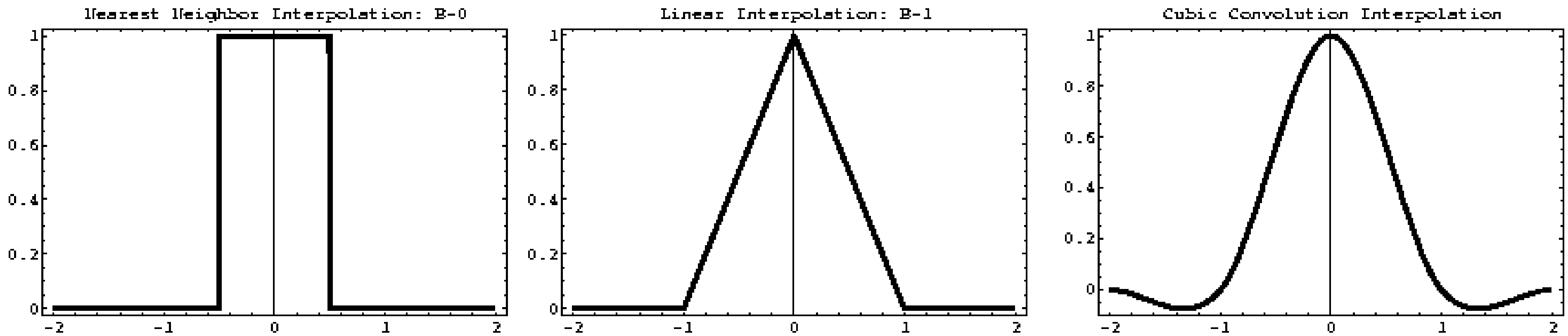


Praktisk algoritme:

```
x₀ = floor(x), y₀ = floor(y)
x₁ = ceil(x),  y₁ = ceil(y)

Δx = x - x₀
Δy = y - y₀
p = f(x₀,y₀)+[f(x₁,y₀)-f(x₀,y₀)] Δx
q = f(x₀,y₁)+[f(x₁,y₁)-f(x₀,y₁)] Δx
f(x',y') = p+(q-p)Δy
```

Høyere-ordens interpolasjon

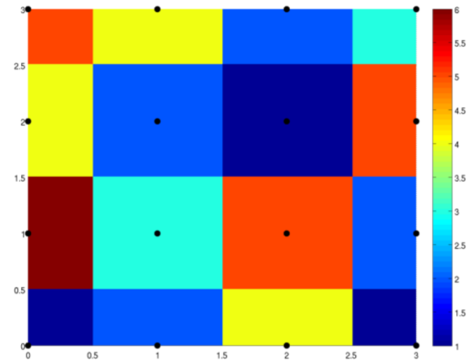
- Kubisk interpolasjon benytter et naboskap på 4×4 piksler
- Interpolasjon kan sees på som (kontinuerlig) konvolusjon med bestemte filtre



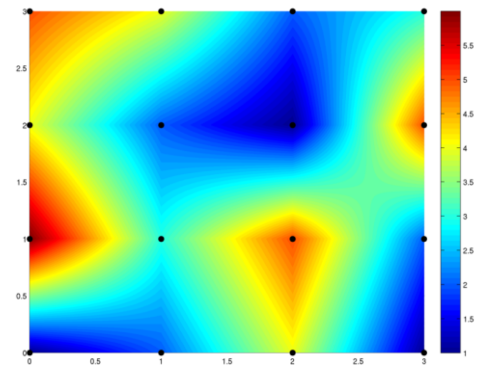
(1D-varianter av nærmeste nabo, lineær og kubisk interpolasjonskjerne)

Interpolasjon – en sammenligning

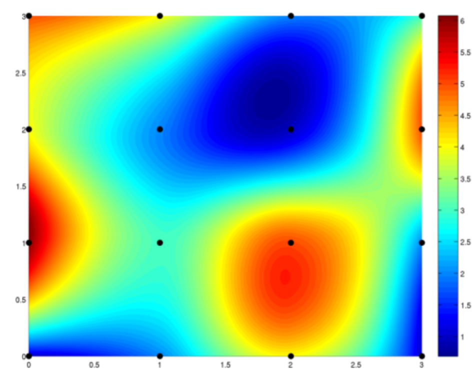
- Nærmeste nabo gir 2D trappefunksjon, med diskontinuitet midt mellom punktene



- Bi-lineær interpolasjon bruker 2 x 2 = 4 piksler. Derivert er ikke kontinuerlig over flaten



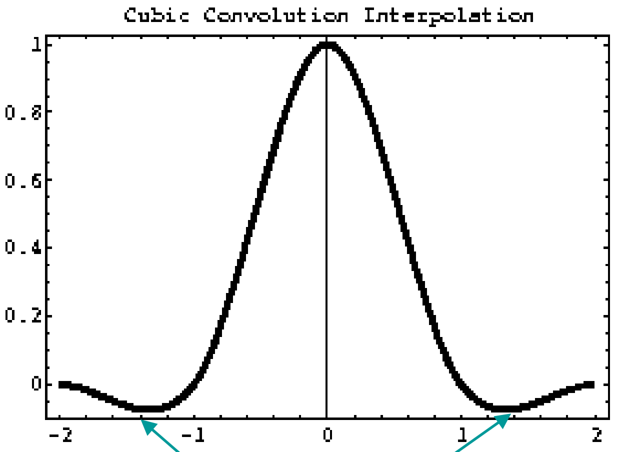
- Bi-kubisk interpolasjon gir også kontinuerlige deriverte, men er mer regnekrevende; bruker 4 x 4 = 16 piksler



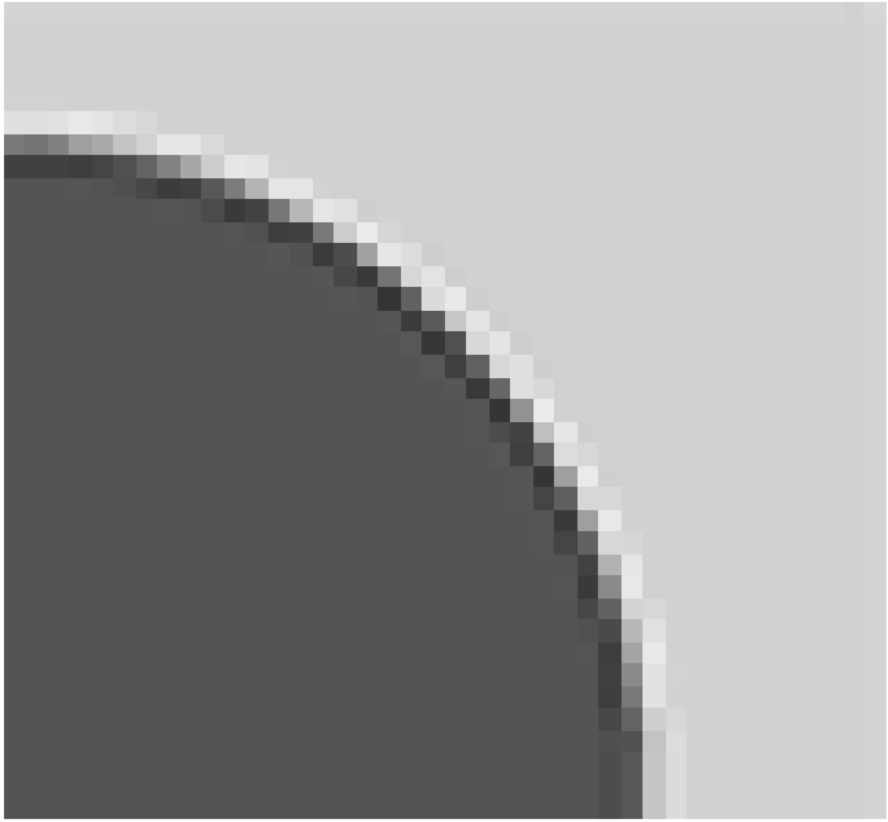
Interpolasjonsfunksjoner i praksis

- 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 segmenert 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
  - Noe skarpere bilder, men kan gi opphav til «kant-klorie-effekter»

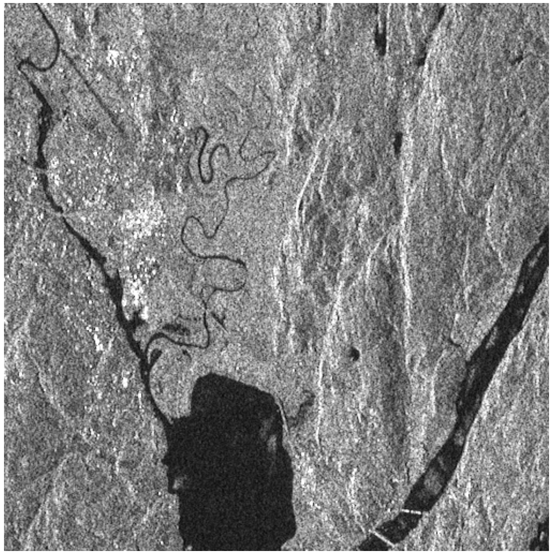
”Kant-glorie-effekter” / ”ringing” ved kubisk interpolasjon



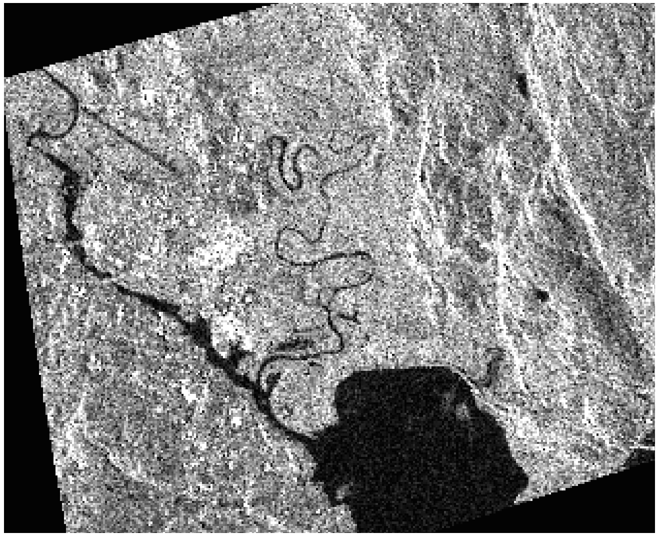
Negative «lobe»-verdier



Eksempel på bruk av geometriske transformer: Samregistrering av bilder



Original



Transformert

Ønsket bilde å samregistrere med

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 transformer 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 $(x_i,y_i),(x_i^r,y_i^r)$ («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

NB: Bytt om rekkefølgen på bildene for å få en inverstransform, jfr. resamplingen!

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

$$\underbrace{\begin{bmatrix} x_1^r \\ x_2^r \\ \vdots \\ x_M^r \end{bmatrix}}_{\text{d}} \quad \underbrace{\begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_M & y_M & 1 \end{bmatrix}}_{\text{G}} \underbrace{\begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix}}_{\text{a}}$$

Vi er på jakt etter koeffisientene i denne **a**-vektoren

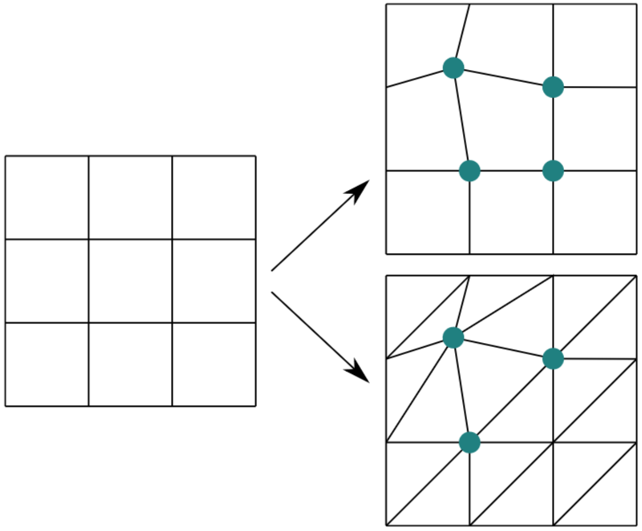
$$J_x = (d - Ga)^T (d - Ga) = d^T d + a^T G^T G a - 2a^T G^T d$$

$$\frac{\partial J_x}{\partial a^T} = 2G^T G a - 2G^T d = 0 \qquad \Rightarrow \qquad \boxed{a = \underbrace{(G^T G)^{-1} G^T}_{\text{Én enkelt matrise}} d}$$

Kursorisk pensum

Stykkevise transformer

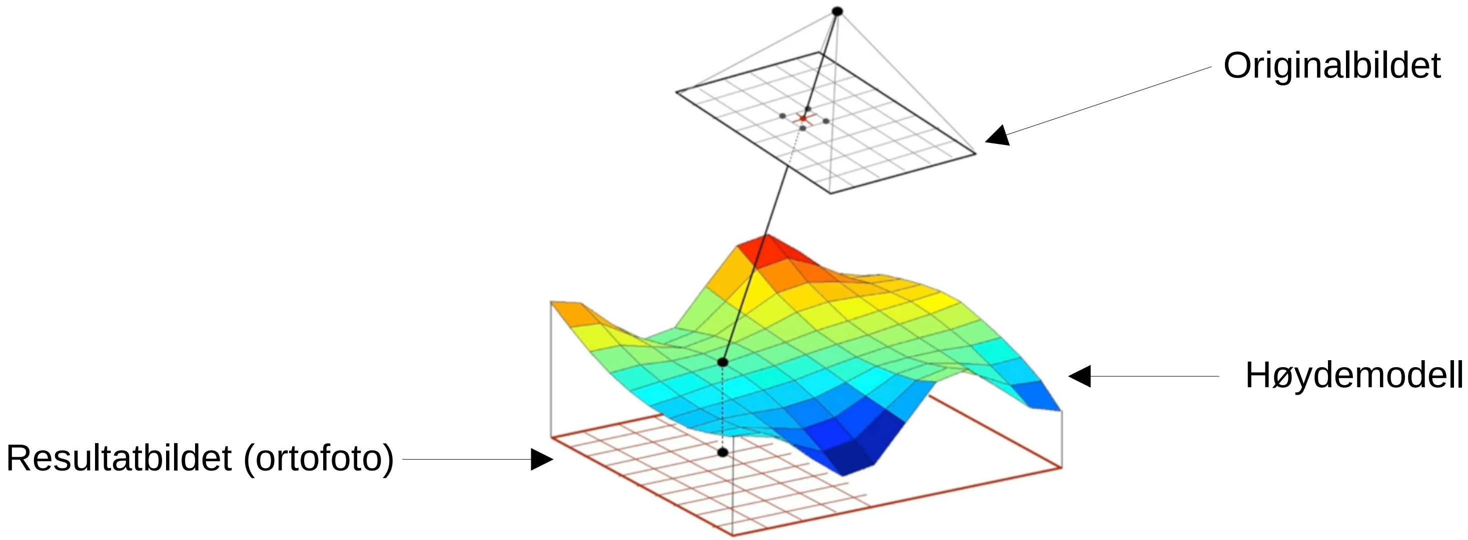
- Forskjellige transformer for ulike deler av bildet
- Eksempel: Bruken av et kontrollgrid som styrer hvordan de ulike delene skal endres



Kursorisk pensum

Ortorektifisering

- Ofte benyttet teknikk ved fjernmåling
- Projisere bildet på et kart
- Ønsker et bilde som om tatt uendelig langt fra og vinkelrett på overflaten
- Krever ideelt en høydemodell, T(x,y) enkel projeksjon om ikke



Oppsummering

- Transform/endring av pikslenes posisjoner (x-,y-koordinater)
  - Fokus på affine transformer

- Resampling
  - Forlengs- og baklengsmapping
  - Interpolasjonsmetoder
    - Nærmeste nabo-interpolasjon
    - Bilineær interpolasjon
    - Høyere-ordens interpolasjoner

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

- Bruk av geometriske operasjoner til å samregistrere bilder
  - Kontrollpunkter