

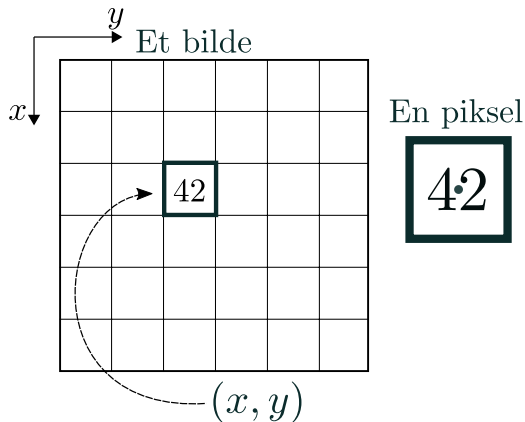
INF2310 - Geometriske operasjoner

29. januar 2020

- Affine transformasjoner
- Resampling og interpolasjon
- Samregistrering av bilder

Oppsummering: Hva et digitalt bilde er

- En matrise med verdier
- Hver piksel har en verdi og en koordinat (x, y)
- Øvre, venstre hjørne er ofte element med koordinat $(0, 0)$



Hva er en geometrisk operasjon?

- Endrer på posisjoner til piksler

Hva er en geometrisk operasjon?

- Endrer på posisjoner til piksler
- Posisjonen (x, y) til hvert piksel blir flyttet på gjennom transformasjonene T_x og T_y ,

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

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

Hva er en geometrisk operasjon?

- Endrer på posisjoner til piksler
- Posisjonen (x, y) til hvert piksel blir flyttet på gjennom transformasjonene T_x og T_y ,

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

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

- T_x og T_y er ofte polynomer

- Forstørre deler av bildet for visuell inspeksjon (zoome)
- Rette opp for geometriske feil som oppstår under avbildning
 - Rotasjon
 - Fiskeøyelinse
 - Avbildning av terreng (radar/sonar)
 - Linsekorrigerings
 - ...
- Samregistrering av bilder
 - ... fra ulike sensorer (f.eks CT, MR, ultralyd)
 - ... tatt på ulike tidspunkt / vinkler
 - ... kart i bestemt kartprojeksjon
 - ... f.eks i ansiktsgjenkjenning der ansiktene skal være på samme plass som et referansebilde
 - ...
- Spesialeffekter

- Transformasjon av koordinatene (x, y) til (x', y') ,

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

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

Affine transformasjoner

- Transformasjon av koordinatene (x, y) til (x', y') ,

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

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

- Affine transformasjoner beskrives ved

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

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

Affine transformasjoner

- Transformasjon av koordinatene (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 matrise form:

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

Affine transformasjoner

- Transformasjon av koordinatene (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 matrise form:

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

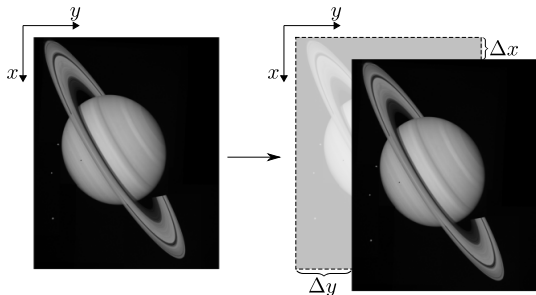
Egenskaper til affine transformasjoner

- Rette linjer bevares
- Parallelle linjer forblir parallelle
- Lineær transformasjon + translasjon
- Kan uttrykkes ved matrisemultiplikasjon

Eksempler på affine transformasjoner

- Affine transformasjoner:

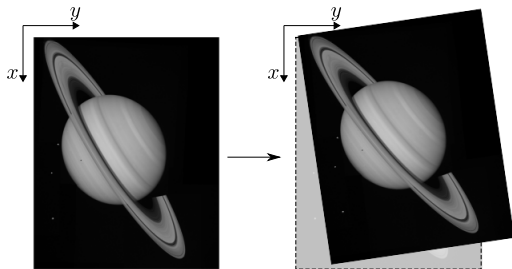
- Translasjon (forflytning)
- Rotasjon
- "Shearing"
- Refleksjon
- Skalering



Eksempler på affine transformasjoner

- Affine transformasjoner:

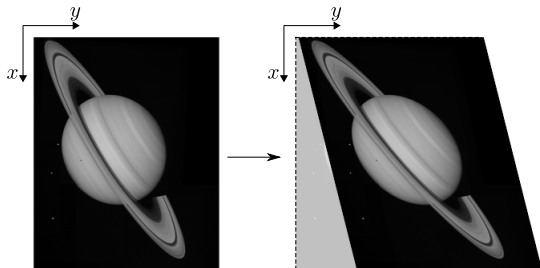
- Translasjon (forflytning)
- Rotasjon
- "Shearing"
- Refleksjon
- Skalering



Eksempler på affine transformasjoner

- Affine transformasjoner:

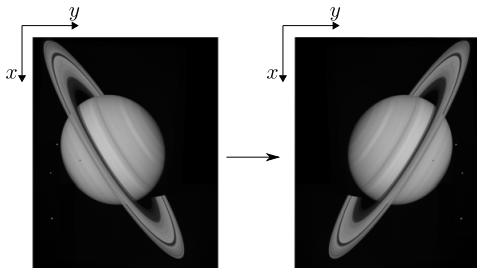
- Translasjon (forflytning)
- Rotasjon
- "Shearing"
- Refleksjon
- Skalering



Eksempler på affine transformasjoner

- Affine transformasjoner:

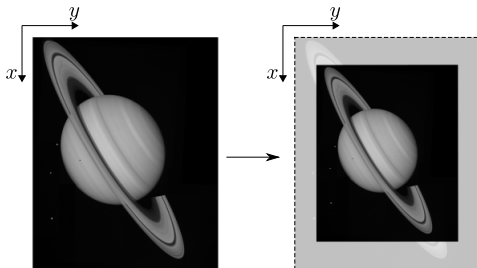
- Translasjon (forflytning)
- Rotasjon
- "Shearing"
- Refleksjon
- Skalering



Eksempler på affine transformasjoner

- Affine transformasjoner:

- Translasjon (forflytning)
- Rotasjon
- "Shearing"
- Refleksjon
- Skalering







- Affine transformasjoner:
 - Translasjon (forflytning)
 - Rotasjon
 - "Shearing"
 - Refleksjon
 - Skalering

Disse transformasjonene kan kombineres til å gi andre affine transformasjoner!

Eksempler på affine transformasjoner

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

Transformasjon	Transformasjonsmatrise	Uttrykk	Eksempel
Translasjon	$\begin{bmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{bmatrix}$	$x' = x + \Delta x$ $y' = y + \Delta y$	
Rotasjon	$\begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x' = x \cos \theta - y \sin \theta$ $y' = x \sin \theta + y \cos \theta$	
"Shear"	$\begin{bmatrix} 1 & s_y & 0 \\ s_x & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x' = x + s_y y$ $y' = y + s_x x$	
Skalering	$\begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x' = c_x x$ $y' = c_y y$	

Sammenslåing av affine transformasjoner

$$\left[\begin{array}{c} \text{translasjon} \end{array} \right] \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \quad \left[\begin{array}{c} \text{rotasjon} \end{array} \right] \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix}$$



$$\left[\begin{array}{c} \text{rotasjon} \end{array} \right] \left[\underbrace{\left[\begin{array}{c} \text{translasjon} \end{array} \right] \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}_{\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}} \right] = \begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$



$$\left[\begin{array}{c} \text{translasjon \& rotasjon} \end{array} \right] \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix}$$

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

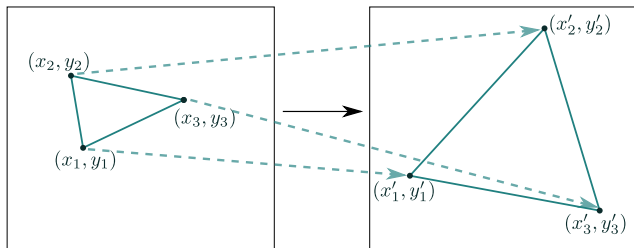
Ønsker å finne a_0, a_1, a_2, b_0, b_1 og b_2 .

Finne transformkoeffisientene

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

Ønsker å finne a_0, a_1, a_2, b_0, b_1 og b_2 .

- Kan bestemmes ved å spesifisere tre punkter før og etter transformasjonen



Andre former for geometriske transformasjoner

- Bilineære transformasjoner:

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

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

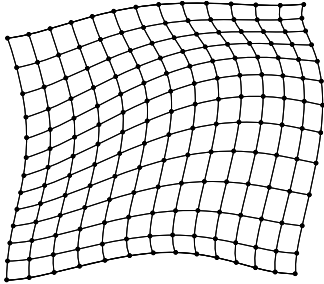
- Kvadratiske former:

$$x' = a_0x + a_1y + a_2 + a_3xy + a_4x^2 + a_5y^2$$

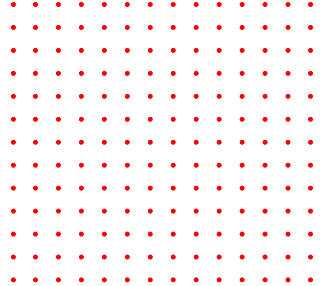
$$y' = b_0x + b_1y + b_2 + b_3xy + b_4x^2 + b_5y^2$$

- Kan brukes til å korrigere for mer komplekse avbildningsfeil
- Flere koeffisienter betyr flere punktpar som må brukes

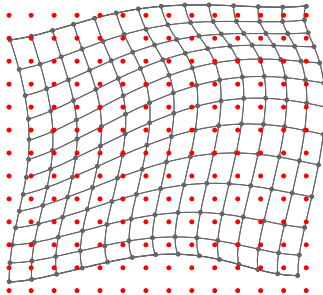
Resampling



Kun sampleverdier der
linjene krysser

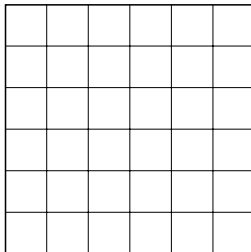


Ønsker bildet samlet i et
regulært grid \rightarrow resample

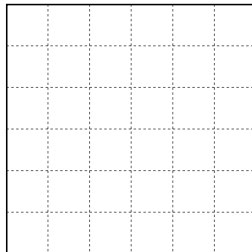


Gå gjennom alle piksler i **innbildet** og flytt dem til **utbildet**

Innbilde



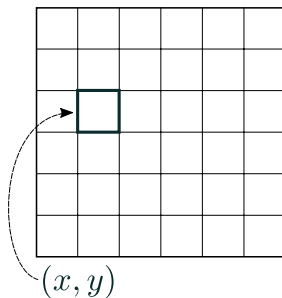
Utbilde



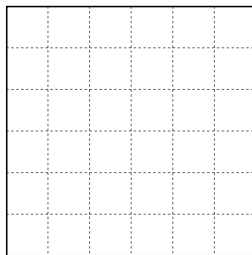
Forlengings-mapping

Gå gjennom alle piksler i **innbildet** og flytt dem til utbildet

Innbilde

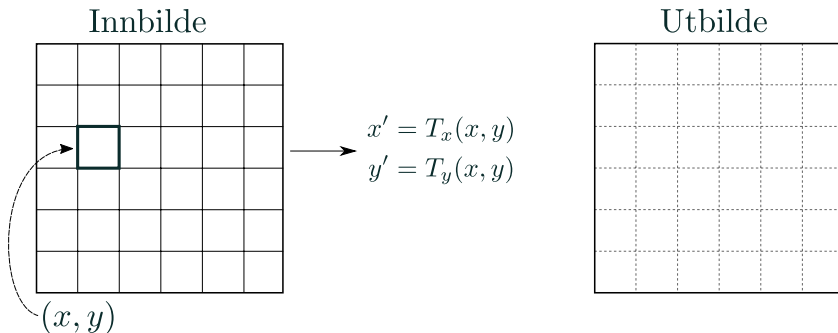


Utbilde

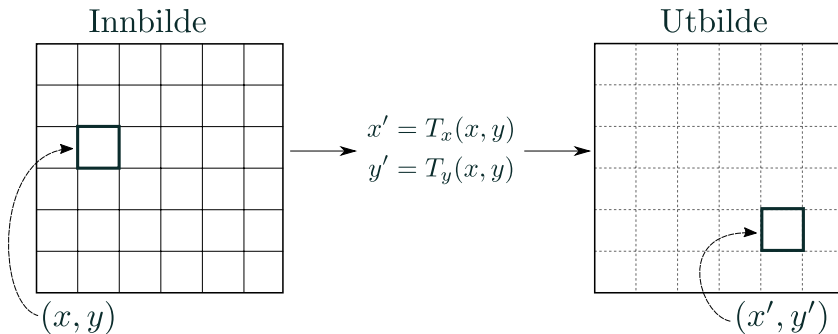


Forlengings-mapping

Gå gjennom alle piksler i **innbildet** og flytt dem til utbildet



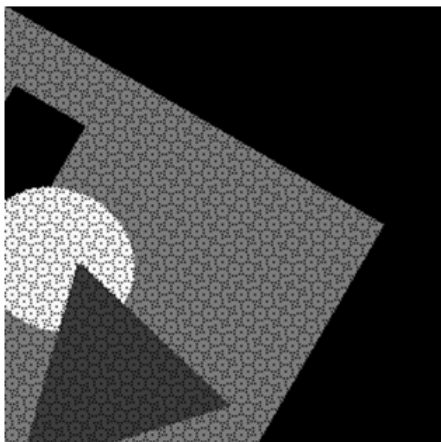
Gå gjennom alle piksler i **innbildet** og flytt dem til utbildet



Algorithm 1: Forlengs-mapping

```
1: for alle  $(x, y)$  i INNBILDE do  
2:    $x' = T_x(x, y)$   
3:    $y' = T_y(x, y)$   
4:   if  $(x', y')$  er i UTBILDE then  
5:     UTBILDE $(x', y') =$  INNBILDE $(x, y)$   
6:   end if  
7: end for
```

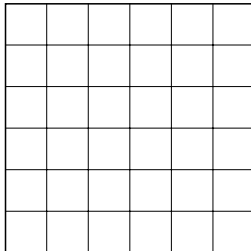
Forlengs-mapping



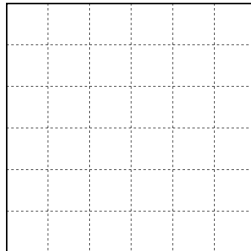
Baklengs-mapping (invers mapping)

Gå gjennom alle piksler i **utbildet** og finn ut hvilken piksel verdi fra innbildet vi må hente

Innbilde



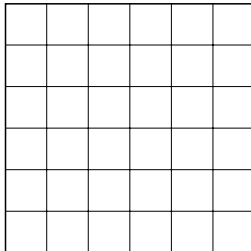
Utbilde



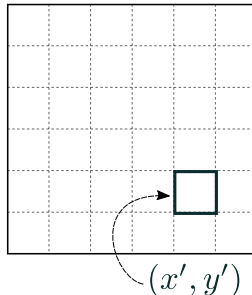
Baklengs-mapping (invers mapping)

Gå gjennom alle piksler i **utbildet** og finn ut hvilken piksel verdi fra innbildet vi må hente

Innbilde

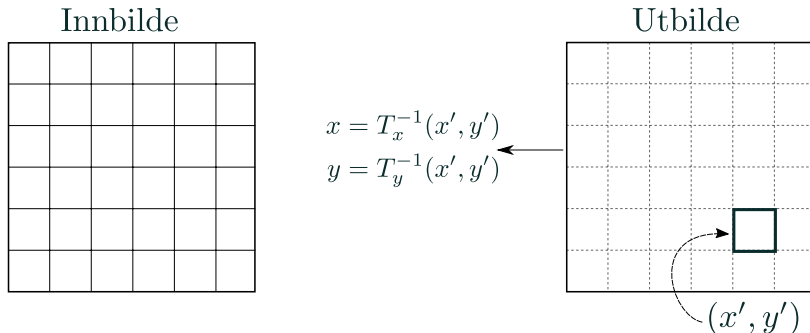


Utbilde



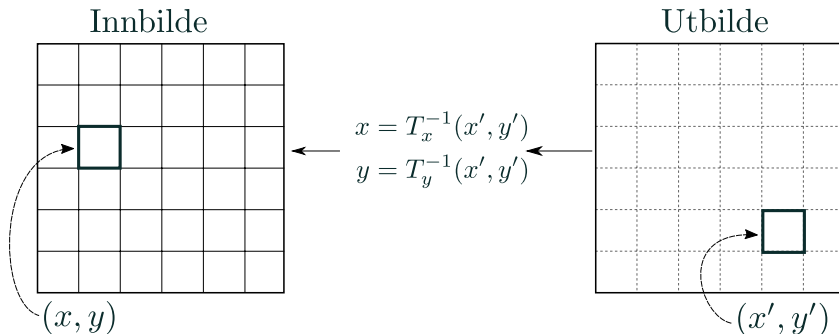
Baklengs-mapping (invers mapping)

Gå gjennom alle piksler i **utbildet** og finn ut hvilken piksel verdi fra innbildet vi må hente



Baklengs-mapping (invers mapping)

Gå gjennom alle piksler i **utbildet** og finn ut hvilken piksel verdi fra innbildet vi må hente



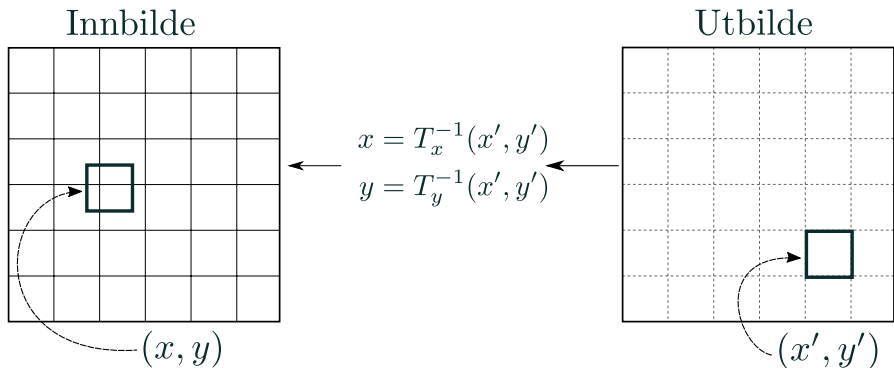
Algorithm 2: Baklengs-mapping

- 1: **for** alle (x', y') i UTBILDE **do**
 - 2: $x = T_x^{-1}(x', y')$
 - 3: $y = T_y^{-1}(x', y')$
 - 4: **if** (x, y) er i INNBILDE **then**
 - 5: UTBILDE (x', y') = interpolert verdi til INNBILDE (x, y)
 - 6: **end if**
 - 7: **end for**
-

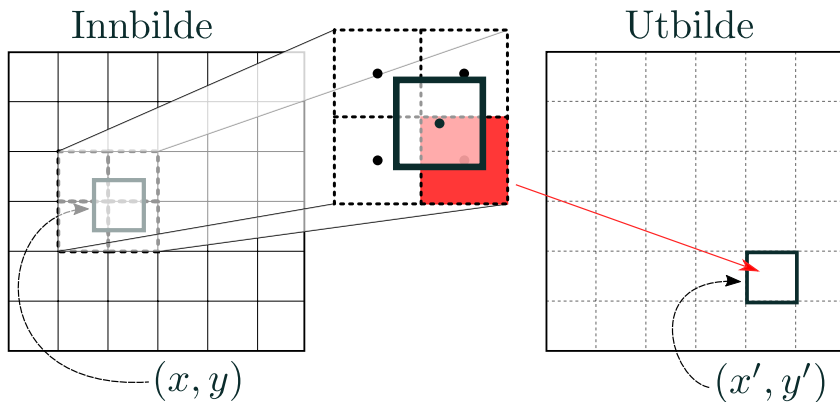
Baklengsmapping

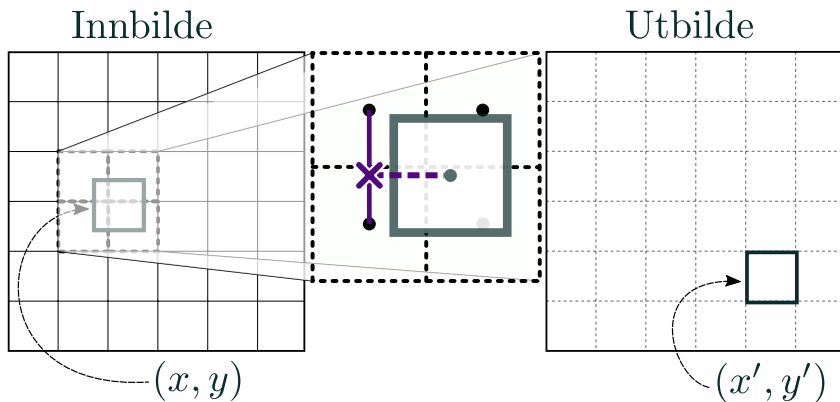


Hvorfor interpolasjon i baklengs-mapping?

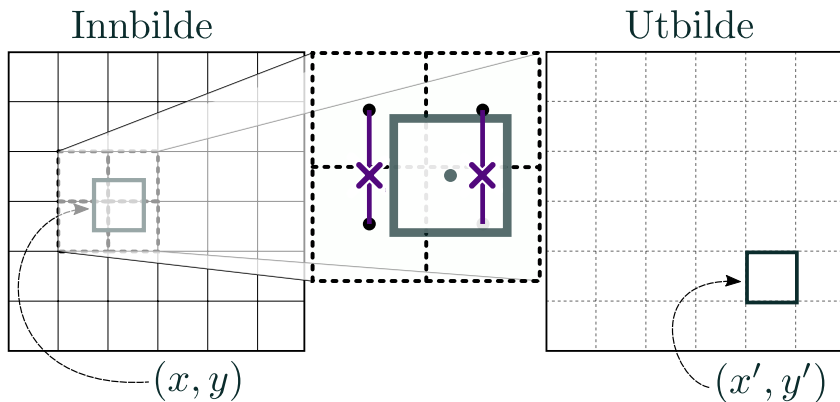


Nærmeste nabo interpolasjon

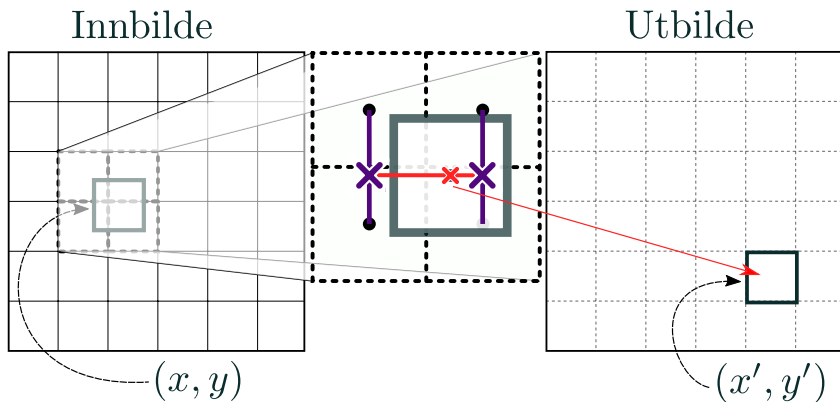




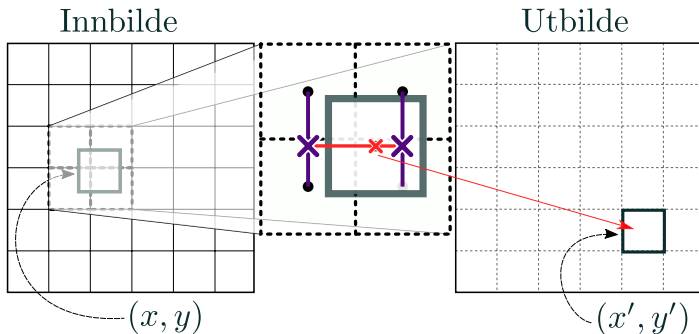
Bilinear interpolasjon



Bilinear interpolasjon



Bilinear interpolasjon



$$x_0 = \text{floor}(x); y_0 = \text{floor}(y)$$

$$x_1 = \text{ceil}(x); y_1 = \text{ceil}(y)$$

$$\Delta x = x - x_0$$

$$\Delta y = y - y_0$$

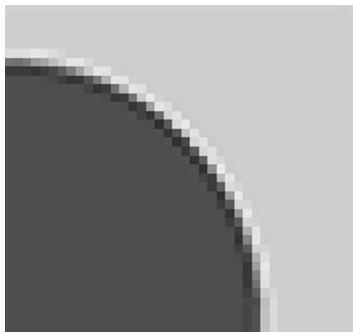
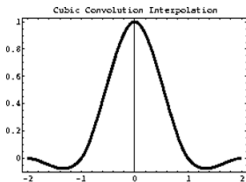
$$p = f_{\text{INN}}(x_0, y_0) + (f_{\text{INN}}(x_1, y_0) - f_{\text{INN}}(x_0, y_0))\Delta x$$

$$q = f_{\text{INN}}(x_0, y_1) + (f_{\text{INN}}(x_1, y_1) - f_{\text{INN}}(x_0, y_1))\Delta x$$

$$f_{\text{UT}}(x', y') = p + (q - p)\Delta y$$

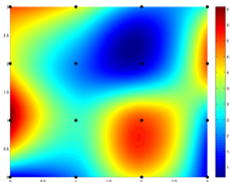
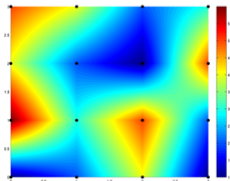
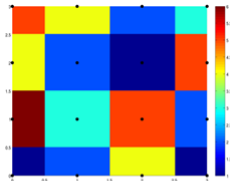
Høyere-ordens interpolasjon

- Kontinuerlige deriverte av høyere orden
- Mer regnekrevende
- Kan gi opphav til "kant-glorie" effekter



Baklengsmapping med ulike interpolasjonsmetoder

- Nærmeste nabo gir en trappefunksjon i 2D og skaper diskontinuitet midt mellom punktene.
Bruker ett piksel pr. piksel
- Bilineær interpolasjon gir kontinuerlig overflate, men diskontinuerlig derivert.
Bruker $2 \times 2 = 4$ piksler pr. piksel.
- Bikubisk interpolasjon gir glatte og kontinuerlige flater, men er mer regnekrevende.
Bruker $4 \times 4 = 16$ piksler pr. piksel.



- **Nærmeste nabo:**
 - Taggete kanter
 - Hver piksel bruker verdi fra innbildet

- **Nærmeste nabo:**
 - Taggete kanter
 - Hver piksel bruker verdi fra innbildet

- **Bilineær:**
 - Kontinuerlige resultat
 - Noe mer regnekrevende

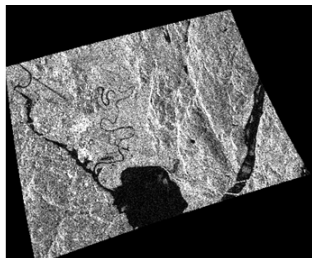
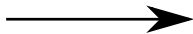
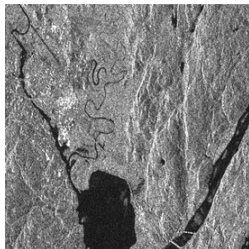
- **Nærmeste nabo:**
 - Taggete kanter
 - Hver piksel bruker verdi fra innbildet

- **Bilineær:**
 - Kontinuerlige resultat
 - Noe mer regnekrevende

- **Høyere-ordens interpolasjon (f.eks bikubisk):**
 - Kontinuerlig derivert av ønsket orden
 - Betydelig mer regnekrevende
 - kan gi opphav til "kant-glorie" effekter

Bruk av geometriske transformasjoner: Samregistrering

Innbilde



Transformert bilde



Ønsket bilde å
samregistrere med

- Hvis kartkoordinatene er kjent i bildene, kan de brukes til å finne transformasjonskoeffisientene
- Hvis ikke, kan vi velge kontrollpunktene selv

- Kan bruke flere kontrollpunkter enn nødvendig for å bestemme transformasjonskoeffisientene

- Kan bruke flere kontrollpunkter enn nødvendig for å bestemme transformasjonskoeffisientene
- Finner ofte transformasjonskoeffisienter som minimerer sum av kvadrat feil ved de valgte punktparene i inn- og referansebilde

- Kan bruke flere kontrollpunkter enn nødvendig for å bestemme transformasjonskoeffisientene
- Finner ofte transformasjonskoeffisienter som minimerer sum av kvadrat feil ved de valgte punktparene i inn- og referansebilde
- Anta N punkt i innbilde (x_i, y_i) og i referansebilde $(x_i^{(r)}, y_i^{(r)})$

- Kan bruke flere kontrollpunkter enn nødvendig for å bestemme transformasjonskoeffisientene
- Finner ofte transformasjonskoeffisienter som minimerer sum av kvadrat feil ved de valgte punktparene i inn- og referansebilde
- Anta N punkt i innbilde (x_i, y_i) og i referansebilde $(x_i^{(r)}, y_i^{(r)})$
- Vi ønsker å finne transformasjonen $(x_i, y_i) \rightarrow (x_i^{(r)}, y_i^{(r)})$ som minimerer

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

Samregistrering ved minimering av kvadratfeil

Ønsker å minimere:

Utleddingen av hvordan vi får vektoren \mathbf{a} er kursorisk!

$$J = \sum_{i=1}^N \left(x'_i - x_i^{(r)} \right)^2 + \left(y'_i - y_i^{(r)} \right)^2 = J_x + J_y$$

La

$$\mathbf{d} = \begin{bmatrix} x_1^{(r)} \\ x_2^{(r)} \\ \vdots \\ x_N^{(r)} \end{bmatrix} \quad G = \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ & \vdots & \\ x_N & y_N & 1 \end{bmatrix} \quad \mathbf{a} = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix}$$

Ser på $J_x = \sum_{i=1}^N \left(x'_i - x_i^{(r)} \right)^2$ (tilsvarende tankegang for J_y):

$$J_x = (\mathbf{d} - G\mathbf{a})^T (\mathbf{d} - G\mathbf{a}) = \mathbf{d}^T \mathbf{d} - 2\mathbf{a}^T G^T \mathbf{d} + \mathbf{a}^T G^T G \mathbf{a}$$

$$\nabla_{\mathbf{a}} J_x = 2G^T G \mathbf{a} - 2G^T \mathbf{d} = \mathbf{0}$$

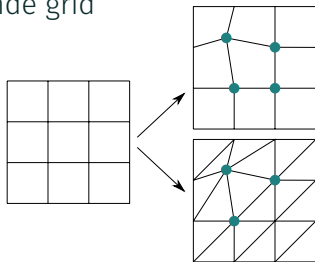
$$\Rightarrow \mathbf{a} = \underline{\underline{(G^T G)^{-1} G^T \mathbf{d}}}$$

Stykkevis transformasjoner

- Forskjellige transformasjoner på ulike deler av bildet
- Må bestemme struktur på underliggende grid (kvadrater, rektangler, trekanter, ...)
- Grid bestemmer hvordan de ulike delene av bildet skal endres
- Bilineær transformasjon kan brukes,

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

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



- Transformasjoner av pikslenes koordinater
- Resampling
 - Forlengs-mapping
 - Baklengs-mapping
- Interpolasjonsmetoder
 - Nærmeste nabo interpolasjon
 - Bilineær interpolasjon
 - Høyere-ordens interpolasjon
- Bruk av geometriske operasjoner til å samregistrere bilder