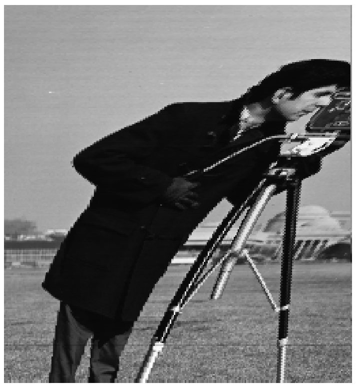
Bildverarbeitung und Algorithmen

Prof. Dr. Wolfgang Konen



Warping im Detail

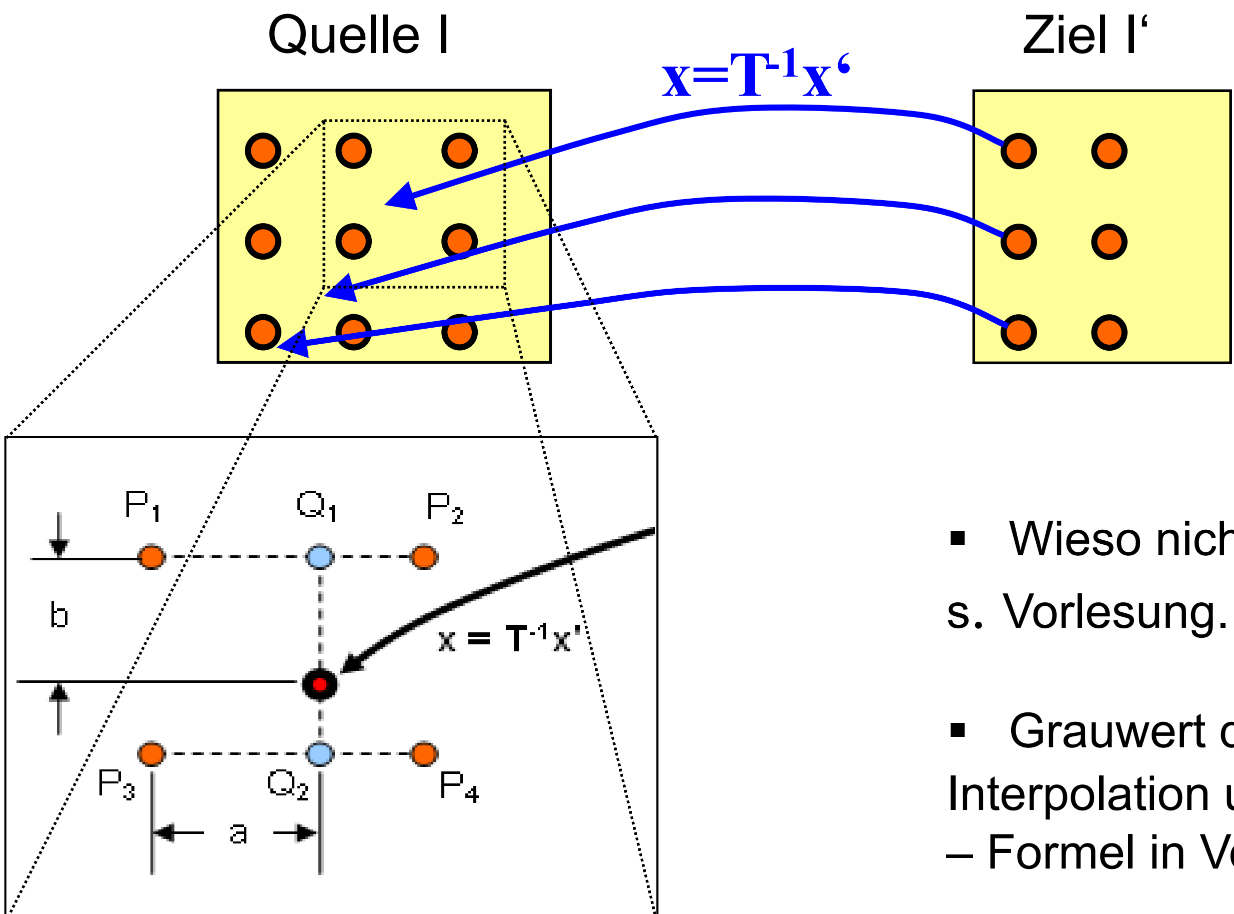
Die Schritte des Warping-Algorithmus

- **Schritt 1: Passpunkte finden**
  - automatisch oder manuell
- **Schritt 2: Suche zu den Passpunkten die beste Transformation**
  - analytisch (z.B. Vertreter aus Klasse der affinen Transformationen)
  - oder stückweise linear für jeden Pixelort **x'** im Zielbild (**Gridding**)
- **Schritt 3: Grauwerte der Zielpixel festlegen**
  - durch Interpolation am Quellort (nächster Nachbar, bilinear, bikubisch, ...)

Aus didaktischen Gründen gehen wir nachfolgend diese Schritte in umgekehrter Reihenfolge durch

Warping, Schritt 3
(Transformationen T und T^{-1} seien gegeben)

- **Backwards-Warp: Zielpixelort x' wird durch Quellort x koloriert**



Quelle: aufg_loes.htm

- Wieso nicht Vorwärts-Warp? – s. Vorlesung.
- Grauwert durch z.B. bilineare Interpolation um Quellort **x** herum – Formel in Vorlesung

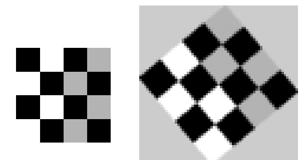
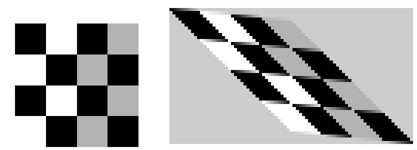
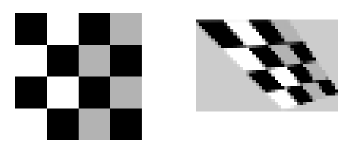
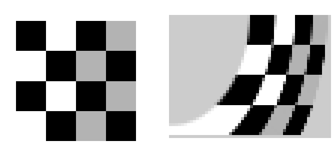
Warping, Schritt 2
Geometrische Transformation T

- **Typisierung analytischer Transformationen**

Euclidean / Procrustes	$x' = ax + by + t_x \quad , \quad a = s\cos\alpha$ $y' = -bx + ay + t_y \quad , \quad b = s\sin\alpha$
Affine / 1^{st} order polynomial	$x' = a_0 + a_1 x + a_2 y$ $y' = b_0 + b_1 y + b_2 y$
Bilinear	$x' = a_0 + a_1 xy + a_2 x + a_3 y$ $y' = b_0 + b_1 xy + b_2 x + b_3 y$
Perspective	$x' = (a_0 + a_1 x + a_2 y)/(c_0 x + c_1 y + 1)$ $y' = (b_0 + b_1 x + b_2 y)/(c_0 x + c_1 y + 1)$
2^{nd} order polynomial / Biquadratic	$x' = a_0 + a_1 x + a_2 y + a_3 x^2 + a_4 y^2 + a_5 xy$ $y' = b_0 + b_1 x + b_2 y + b_3 x^2 + b_4 y^2 + b_5 xy$
General polynomial	$x' = \sum_i \sum_j a_{ij} x^i y^j$ $y' = \sum_i \sum_j b_{ij} x^i y^j$

Warping, Schritt 2
Geometrische Transformation T

- **Typisierung analytischer Transformationen**

Euklid / Procrustes	Geraden bleiben Geraden, Quadrate bleiben Quadrate	
Affin	Geraden bleiben Geraden, Parallelen bleiben parallel, Rechteck $\rightarrow$ Parallelogramm	
Bilinear	Geraden bleiben Geraden, Rechteck $\rightarrow$ allg. Viereck	
perspektivisch (projektiv)	Geraden bleiben Geraden, Parallelen $\rightarrow$ Geraden mit gemeinsamen Fluchtpunkt	
Biquadratisch	Geraden werden zu Kurven	

Warping, Schritt 2
Geometrische Transformation T

- **Typisierung analytischer Transformationen**

Euklid / Procrustes	Kombination Translation + Rotation + globale Skalierung	mind. 2 Kontrollpunkte
Affin	Kombination Translation + Rotation + x-Skalierung + y-Skalierung	mind. 3 Kontrollpunkte
Bilinear		mind. 4 Kontrollpunkte
perspektivisch (projektiv)		mind. 5 Kontrollpunkte
Biquadratisch		mind. 6 Kontrollpunkte

Warping, Schritt 2
Geometrische Transformation T

- **Wie genau macht man den Warp in MATLAB?**
  1. wenn T gegeben ist
  2. wenn T aus Passpunkten zu berechnen ist
     a) beste Transformation im LS-Sinne aus bestimmter Klasse von Transformationen suchen >> s. Übung
     b) stückweise-linear interpolieren zwischen Passpunkten (**Gridding***)

* **Gridding** ist der Vorgang, bei dem eine Funktion, deren Werte nur an einigen Stellen gegeben sind, für alle Punkte in einem regelmäßigen Gitter (**Grid**) berechnet wird.

Warping, Schritt 2
2.1 vorgegebenes T

- **Zu 2.1.: Sei T als homogene 3x3-Matrix bekannt**

```
[h,w] = size(I);
[Xp,Yp] = ndgrid(1:w,1:h);
n = h*w;
X = T \ [Xp(:),YP(:),ones(n,1)]';
```

Xp

1	1	...
2	2	...
3	...	...

Yp

1	2	3
1	2	...
...	...	...

Lösung **X** für $$\mathbf{TX} = \begin{pmatrix} \mathrm{Xp1} & \mathrm{Xp2} & \ldots \\ \mathrm{Yp1} & \mathrm{Yp2} & \ldots \\ 1 & 1 & \ldots \end{pmatrix}$$

Warping, Schritt 2
2.2.a) Bestes T mit LS-Fit

- Zu 2.2.a): Seien mehr Passpunkte gegeben als man zur Bestimmung von T braucht. Bsp. Rotationsstreckung:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a & b \\ -b & a \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \quad \Leftrightarrow \quad \begin{cases} x' = ax + by \\ y' = -bx + ay \end{cases} \quad \Leftrightarrow \quad \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x & y \\ y & -x \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix}$$

Der Trick: Jeder freie Parameter a,b,... taucht genau 1x auf

- Übergang auf mehrere Passpunkte:

$$\begin{pmatrix} x_1' \\ \vdots \\ x_n' \\ y_1' \\ \vdots \\ y_n' \end{pmatrix} = \begin{pmatrix} x_1 & y_1 \\ \vdots & \vdots \\ x_n & y_n \\ y_1 & -x_1 \\ \vdots & \vdots \\ y_n & -x_n \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} \quad \Leftrightarrow \quad \mathbf{X'} = \mathbf{Zt}$$

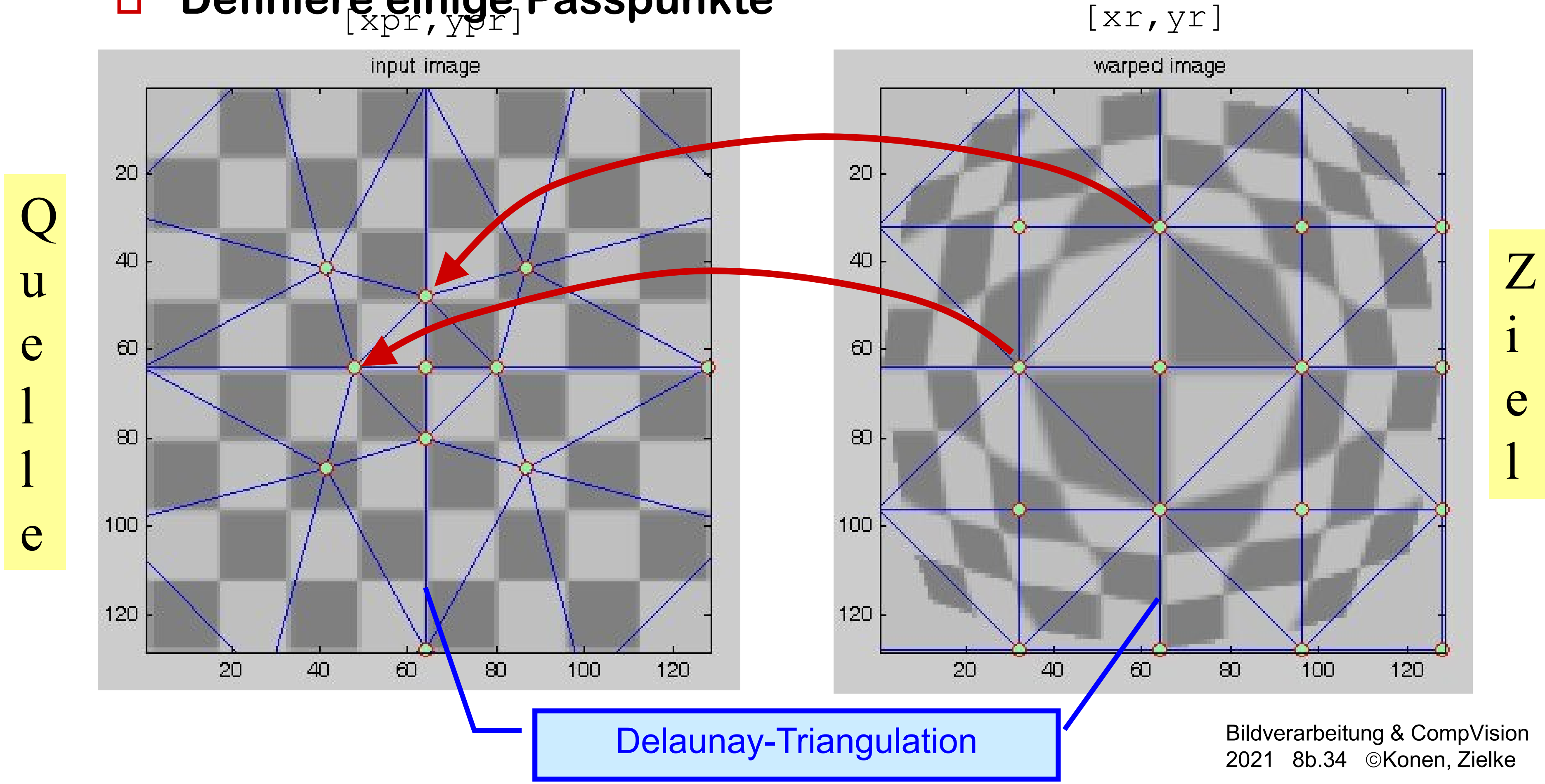
Dieses i.d.R. überbestimmte System für Parametervektor **t** =(a,b)^T wird durch den MATLAB-Befehl

```
t = Z \ Xp;
```

im LS-Sinne gelöst.

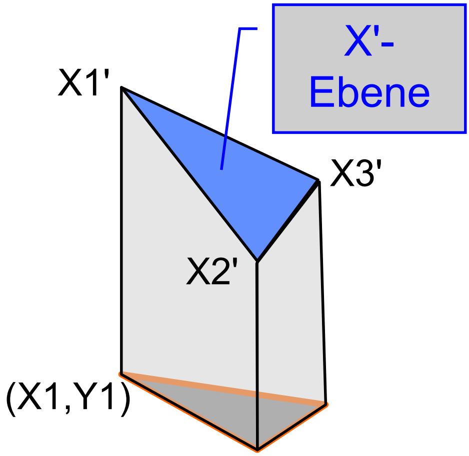
Warping, Schritt 2
2.2.b) stückweise lineare Transformation T

- **Zu 2.2.b): Aufgabe: Die Bildmitte vergrößert abbilden (Fovea)**
- **Definiere einige Passpunkte**



Warping, Schritt 2
2.2.b) stückweise lineare Transformation T

- **Zwischen den Dreieckspunkten**
  - linear interpolieren
  - an Dreieckspunkten ist durch (X1',X2',X3') die Ebene der X'-Koordinatenfunktion gegeben
  - Pixel (x,y) erhält Ebenenwert x'=X'(x,y)
  - analog für (Y1',Y2',Y3') und Y'-Koordinatenfunktion



- **Gridding in MATLAB (foveawarp.m):**

Y1,...,Yn

X1,...,Xn

X1',...,Xn'

gleiche Dim wie `xg`

```
[xg,yg] = ndgrid(1:128,1:128);
xI = griddata(xr,yr, xpr, xg,yg);
yI = griddata(xr,yr, ypr, xg,yg);
Ip(:,:,1)=interp2(I(:,:,1), yI, xI, 'bilinear',255);
Ip(:,:,2)=interp2(I(:,:,2), yI, xI, 'bilinear',255);
Ip(:,:,3)=interp2(I(:,:,3), yI, xI, 'bilinear',255);
```

`griddata` macht Delaunay-Triang. für alle (Xi,Yi) und interpoliert damit neue X'-Werte `xI` an Orten (`xg`,`yg`)

Warping, Schritt 2
Ergänzung

- **Fragen / Aufgaben**
  - Wieso muss es `"...,yI,xI,..."` heißen und nicht `"...,xI,yI,..."`?
  - Können Sie eine Fovealisierungs-Funktion angeben, d.h. eine Funktion, die aus dem Gitter `[xr,yr]` das Gitter `[xpr,ypr]` berechnet?
  - Aufgabe Warping: aufgaben.htm, Lsg.: euclideanwarp2.m

- **zum Schmunzeln**
  - foveawarp2.m

Warping, Schritt 1
Passpunkte finden

- **manuell**
  - o.k. für Bild**be**arbeitung, aber keine (automatisierte) Bild**ver**arbeitung
  - mühsam, wenn viele Bilder
  - optional: Passpunkte per Kreuzkorrelation verbessern
- **automatisch: möglich, wenn korrespondierende Punkte in 2 Bildern vorliegen**
- **diese Aufgabe heißt Matching oder Registrierung**
- **2 Teilprobleme:**
  - das Auffinden geeigneter Punkte im Bild
    - "salient points", Landmarken
    - >> Aperturproblem (!)
  - das genaue Wiederfinden im anderen Bild
    - Template Matching
    - s. Projekt Tracking

Anwendungen Warping

- **Kamerakalibrierung**
  - (Kissen- oder Tonnenverzeichnung korrigieren)
- **industrielle Prüfaufgaben für deformierbare Objekte**
- **Luftaufnahmen Landschaft zur Deckung bringen**
  - ... und dann Differenz, um Unterschiede zu sehen (Ökologie)
- **Inverse Perspektive**
- **medizinische Bildverarbeitung**
  - Matching und Registrierung bei multimodalen, multitemporalen Aufnahmen >> s. Projekt Matching
- **Zwischenbilder berechnen >> s. Projekt Morphing**
- **Facial Animation, 3D-MM (Morphable Models) [Blanz, Vetter] >> s. reanim_exercise.mpg**
- **Mosaicing (mehrere Bilder zusammensetzen) >> s. Projekt Panoramic View**