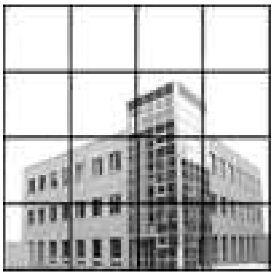
# Geometrische Transformationsverfahren und ihre Einsatzgebiete in der digitalen Bildverarbeitung

*Thomas Bocek s99-706-319 (tom@nope.ch)*

## 1. Was ist eine geometrische Transformation



**Basisbild**
Das Ziel ist die Umsetzung der geometrischen Transformation wie Translation, Rotation, Skalierung und Scherung.
Geometrische Transformationen sind also Koordinatentransformationen (Abbildungen von einer in eine andere Punktmenge)

## 2. Verwendungsmöglichkeiten

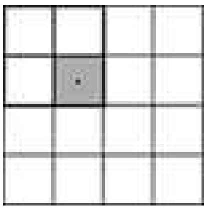
Es gibt viele Verwendungsmöglichkeiten für geometrische Transformationen z.B. Vermessung von realen Objekten mittels digitaler Bilder. Da die heute verwendeten CCD-Sensoren sehr genau sind, sind solche Vermessungen möglich.

Die meisten grafischen Computerspiele verwenden geometrische Transformation um Objekte am Bildschirm darzustellen. Bei 3D Computerspielen wird das fertige Bild zuvor in einer 3D Pipeline berechnet, die aus den drei Schritten besteht: Tesselation, geometrische Transformation und Rendering. Eine genaue Beschreibung würde den Rahmen dieser Arbeit sprengen. (Für weitere Informationen siehe „8.Quellen und weitere Informationen")

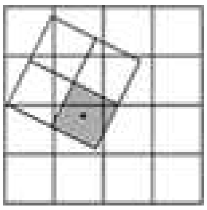
In der Medizin wird die geometrische Transformation häufig zur Diagnostik verwendet. Das heisst zur Aufbereitung und Auswertung von mikroskopischen oder endoskopischen Aufnahmen, Verarbeitung von Ultraschall und Röntgenbildern. Verschiedene Bilder können so angepasst werden, dass man die Bilder überlagern kann, um so eine bessere Diagnose zu erstellen.

## 3. Probleme der geometrischen Transformation

Ein grosses Problem entsteht durch die Tatsache, dass die Rasterungen von Inputbild und Outputbild im allgemeinen nicht übereinstimmen. Es treten zwei Teilprobleme auf:

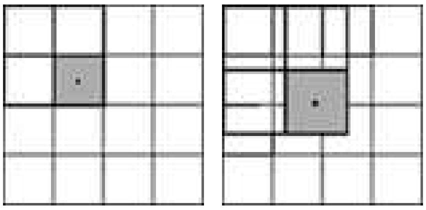


Input Bild



Output Bild.

Die Pixel des Inputbildes werden meistens nicht genau auf das Outputbild abgebildet.
Konkret ist hier die Frage wie das graue Feld im Input Bild nach der Rotation im Outputbild dargestellt werden soll. Welche Felder sollen im Output Bild grau werden.

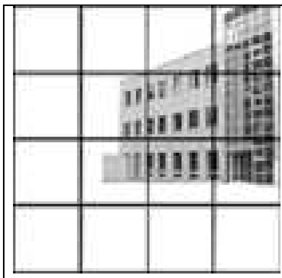


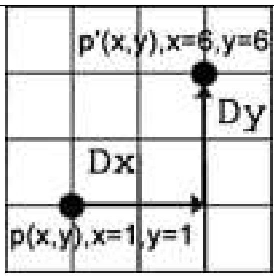
Input Bild Output Bild

Die Pixel des Inputbildes und die Pixel des Outputbildes sind verschieden gross. In diesem Fall muss ein Pixel des InputBildes mehreren Pixeln des Outputbildes zugeordnet werden.
Hier stellt sich die gleiche Frage wie im vorherigen Problem: Welche Felder sollen im Output Bild grau werden.

Diese Probleme werden mit Hilfe von Interpolation gelöst. Es gibt verschieden Interpolationsverfahren, auf die ich jedoch nicht genauer eingehen möchte (nearest neighbour, bilineare Interpolation, bikubische Interpolation; für weitere Informationen siehe „8.Quellen und weitere Informationen“)

## 4. Die Basistransformationen



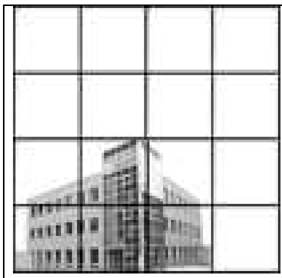


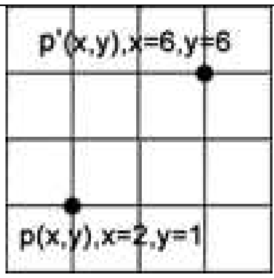

*Translation*
Die “Verschiebung “ des Bildes. Alle Punkte werden in die gleiche Richtung, mit der gleichen Länge verschoben. Die Translation ist durch eine Addition beschrieben

*Matrix-Form*

$$[x'\ y'\ 1]=[x\ y\ 1]=\begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ Dx & Dy & 1\end{pmatrix}$$

Die Translationswerte sind: Dx=6, Dy=6; x’=x+Dx; y’=y+Dy



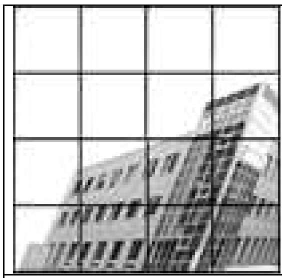


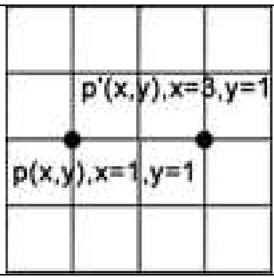

*Skalierung*
Die „Verkleinerung / Vergrösserung“ des Basisbildes. Ein Bild mit der Grösse von z.B. 100x100 Pixel wird auf eine Grösse von 200x200 Pixel vergrössert.

*Matrix-Form*

$$[x'\ y'\ 1]=[x\ y\ 1]=\begin{pmatrix}Sx & 0 & 0\\ 0 & Sy & 0\\ 0 & 0 & 1\end{pmatrix}$$

Die Skalierfaktoren sind: Sx=1/3, Sy=1/6; x’=x*Sx; y’=y*Sy



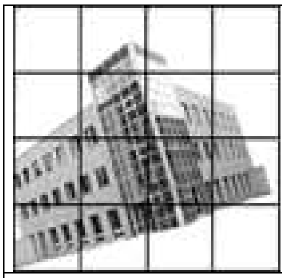


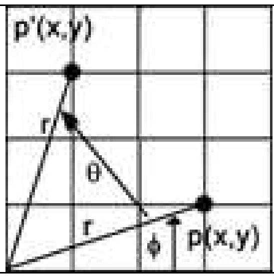

*Scherung*
Die Scherung verschiebt die Pixel nur Zeilen-, oder Spaltenweise. Es entsteht ein „verzerrtes“ Bild

*Matrix-Form*

$$[x'\ y'\ 1]=[x\ y\ 1]=\begin{pmatrix}1 & a & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{pmatrix}$$

Der Scherfaktor ist 2; *x’= x+(y*a); y’= y;* Scherung in *x* um *a* : *Sx*(*a*)






*Rotation (Drehung)*
Die Rotation (Drehung) dreht das Bild im Nullpunkt (0,0) mit dem Winkel θ.

Matrix-Form

$$[x'\ y'\ 1]=[x\ y\ 1]=\begin{pmatrix}\cos(\theta) & \sin(\theta) & 0\\ -\sin(\theta) & \cos(\theta) & 0\\ 0 & 0 & 1\end{pmatrix}$$

x= r*cos(Φ), y= r*sin(Φ)
x’=r*cos(Φ+θ)=r*cos(Φ)* cos(θ)-r * sin(Φ) * sin(θ) → x’= x*cos(θ)-y*sin(θ)
y’=r*sin(Φ+θ)=r*cos(Φ)* sin(θ)-r*sin(Φ)*cos(θ) → y’= y*sin(θ)+y*cos(θ)

## 5. Anmerkung zur Matrix Form

Die Matrizen sind alle in homogenen Koordinaten dargestellt, da die Rotation und die Skalierung durch eine Multiplikation, die Translation jedoch durch eine Addition beschrieben wird. Durch sogenannte *homogene Koordinaten* lässt sich die Translation ebenfalls durch eine Multiplikation berechnen.

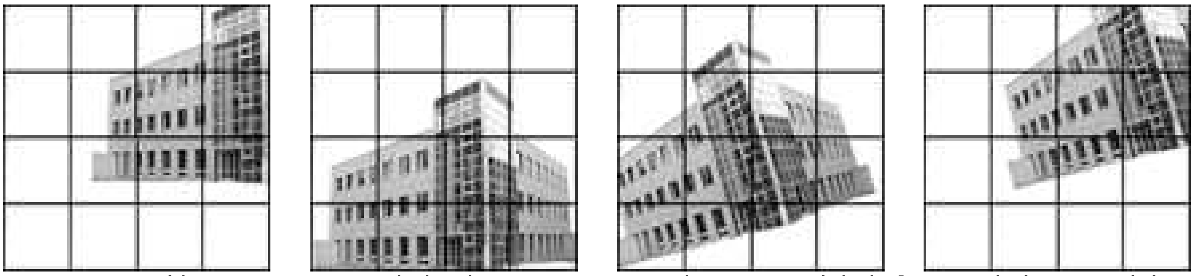
| | |
|---|---|
| Die Umwandlung von „Punkte in 2D“ in „Punkt in homogenen Koordinaten“<br>*P(x,y) → P(x,y,1). Bsp: P(2,3) → P(2,3,1)* | Die Umwandlung von „Punkt in homogenen Koordinaten“ in „Punkte in 2D“<br>*P(x,y,w) → P(x/w,y/w). Bsp P(4,6,2) → P(2,3)* |

## 6. Berechnungen mit der Matrix Form

Mit Hilfe der Matrix Form können nun „Composed“ Transformationen durchgeführt werden. Dazu muss man die Matrizen der einzelnen Transformationen multiplizieren.
Beispiel: Zweimal mal Skalieren mit (Sx1,Sy1) und (Sx2,Sy2)

$$\begin{pmatrix} Sx1 & 0 & 0 \\ 0 & Sy1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} Sx2 & 0 & 0 \\ 0 & Sy2 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} Sx1*Sx2 & 0 & 0 \\ 0 & Sy1*Sy2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Dieses Composed Transforming funktioniert mit allen 4 vorgestellten Transformationen. Solche Transformationen werden auch verwendet, um komplexe Transformationen zu berechnen z.B. eine Rotation um den Punkt P1



*Ausgangsposition* — *Transalation in Koordinatenursprung* — *Rotation um Winkel θ im Nullpunkt* — *Translation zurück in die Ausgangsposition*

Wie sieht die Matrix aus, mit der wir eine Rotation um den Punkt (x,y) durchführen. Diese Aufgabe lässt sich in 3 Basistransformationsschritten lösen. Wir erhalten am Schluss ein Matrix, mit der wir die Operation durchführen können

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -x & -y & 1 \end{pmatrix} \qquad \begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ x & y & 1 \end{pmatrix}$$

*Translation in Koordinatensystemursprung* — *Rotation um Winkel θ* — *Translation zurück in Ausgangsposition*

*Zuerst multiplizieren wir die ersten beiden Matrizen*

$$\begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ -x*\cos(\theta)+y*\sin(\theta) & -x*\sin(\theta)-y*\cos(\theta) & 1 \end{pmatrix} \qquad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ x & y & 1 \end{pmatrix}$$

*Danach multiplizieren wir das Resultat mit der dritten Matrix, und wir erhalten*

$$\begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ -x*\cos(\theta)+y*\sin(\theta)+x & -x*\sin(\theta)-y*\cos(\theta)+y & 1 \end{pmatrix}$$

*Danach vereinfachen wir die Matrix*

$$\begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ x(1-\cos(\theta))+y*\sin(\theta) & y(1-\cos(\theta))-x*\sin(\theta) & 1 \end{pmatrix}$$

## 7. 3D Geometrische Transformation

Wir betrachten die 3-Dimensionalen Transformation als eine Erweiterung der 2- Dimensionalen. Wir benötigen zusätzlich ein z. Die Umwandlung eines 3D Punktes in homogenen Koordinaten lautet (analog zur 2D Umwandlung)

| | |
|---|---|
| Die Umwandlung von „Punkte in 3D“ in „Punkt in homogenen Koordinaten“ *P(x,y,z) → P(x,y,z,1). Bsp: P(2,3,6) → P(2,3,6,1)* | Die Umwandlung von „Punkt in homogenen Koordinaten“ in „Punkte in 3D“ *P(x,y,z,w) → P(x/w,y/w,z/w). Bsp: P(4,6,12,2) → P(2,3,6)* |

Analog zu den Transformationen im 2D Fall, können wir die Matrizen im 3D Fall angeben

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ Dx & Dy & Dz & 1 \end{pmatrix}$$

*Translation*

$$\begin{pmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

*Skalierung*

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ a & 1 & 0 & 0 \\ b & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

*Scherung um die yz Achse*

$$\begin{pmatrix} 1 & a & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & b & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

*Scherung um die xz Achse*

$$\begin{pmatrix} 1 & 0 & a & 0 \\ 0 & 1 & b & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

*Scherung um die xy Achse*

$$\begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 & 0 \\ -\sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

*Rotation um die z-Achse*

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & \sin(\theta) & 0 \\ 0 & -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

*Rotation um die x-Achse*

$$\begin{pmatrix} \cos(\theta) & 0 & -\sin(\theta) & 0 \\ 0 & 1 & 0 & 1 \\ \mathrm{Sin}(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

*Rotation um die y-Achse*

## 8. Quellen und weitere Informationen


[1] Skript „Kernvorlesung Multimediale Systeme“ Prof. Dr. Peter Stucki
Kap. 9

[2] Seminar Grundlagen der Bildverarbeitung, SS 98
http://www.inf.fu-berlin.de/~vratisla/Bildverarbeitung/GeoTrans/GeoTrans.html

[3] 3D Concept - 3dimensionale Welten auf dem Computer - Wie funktionierts?
http://www.3dconcept.ch/artikel/3dhow/

[4] Wie funktioniert 3D
http://www.cs.uni-magdeburg.de/~elanger/graphik-wie.html

[5] Interpolation
http://www.micg.et.fh-stralsund.de/photopub/gruppe2/bildverarbeitung/interpol.html

[6] Interpolation: Digital Imaging: Glossary: Learn: Digital Photography Review
http://www.dpreview.com/learn/Glossary/Digital_Imaging/Interpolation_01.htm

[7] Grundlagen der Computergraphik
http://www.fh-wedel.de/~ko/Data/bba-handout1+2.pdf