Transformationen

Visual Computing

Winter Semester 2018-2019

Prof. Dr. A. Kuijper

Mathematical and Applied Visual Computing (MAVC)
Graphisch-Interaktive Systeme (GRIS)
Fraunhofer IGD
Fraunhoferstrasse 5
D - 64283 Darmstadt

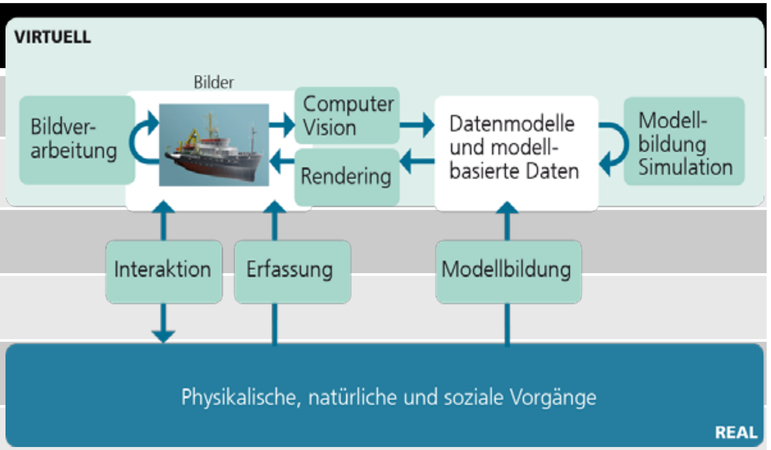
E-Mail: office@gris.tu-darmstadt.de
http://www.gris.tu-darmstadt.de
https://www.mavc.tu-darmstadt.de

$$411+5+7= 423 < 531 < 536$$

In vielen Studiengängen endet die Anmeldefrist am 17. Dezember 2018 - bitte informieren Sie sich rechtzeitig! Ihre Anmeldung nehmen Sie im TUCaN-Webportal im Bereich *Prüfungen* unter *Meine Prüfungen / Anmeldung zu Prüfungen* vor.

Semesterplan

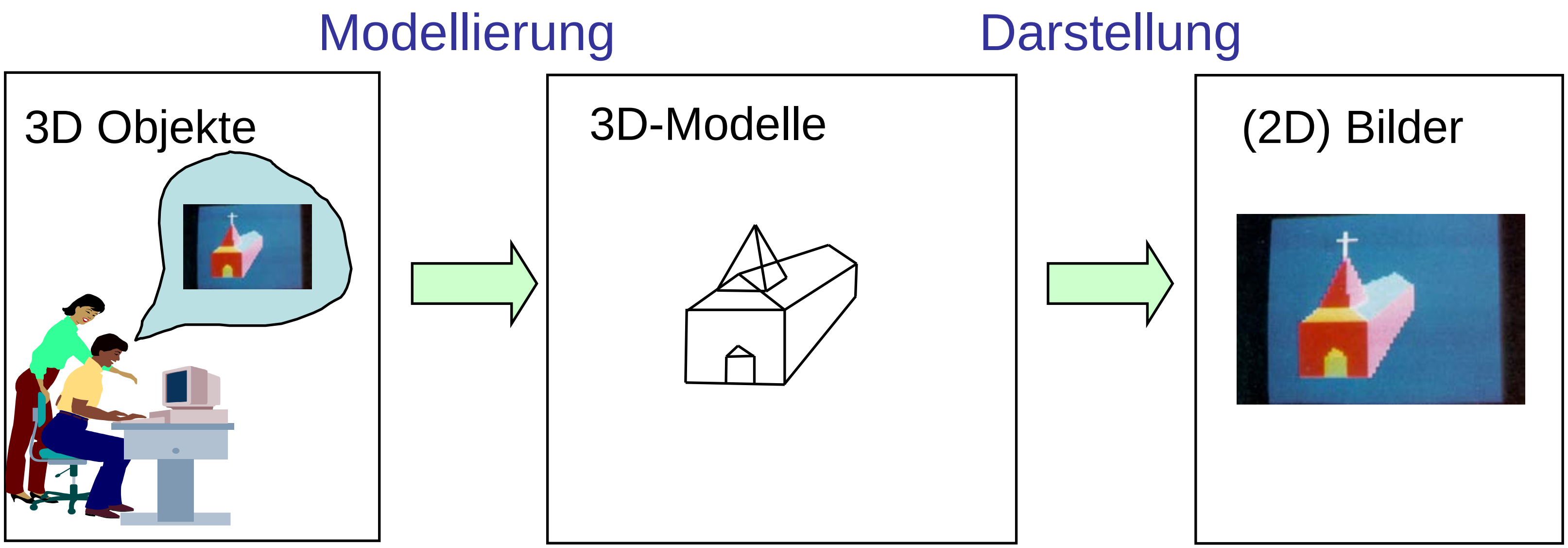
Datum	
26. Okt	Einführung + Visual Computing
02. Nov	Wahrnehmung
09. Nov	Objekterkennung und Bayes
16. Nov	Fourier Theorie
23. Nov	Bilder
30. Nov	Bildverarbeitung
07. Dez	Grafikpipeline & Eingabemodalitäten & VR+AR
14. Dez	Transformationen & 2D/3D Ausgabe
18. Jan	3D-Visualisierung
25. Jan	X3D – 3D in HTML
01. Feb	Informationsvisualisierung
08. Feb	Farbe
15. Feb	User Interfaces + Multimedia Retrieval
07. Mrz	Klausur



Überblick

- **Wiederholung: Grafikpipeline**
- Transformationen und affine Abbildungen
- Skalierung, Scherung, Rotation
- Projektion
- 3D-Interaktion

3D-Grafik-Pipeline - Schematisch



CPU

GPU

Anwendung → Geometrie-verarbeitung → Rasterisierung → Ausgabe

3D-Grafikpipeline

GPU

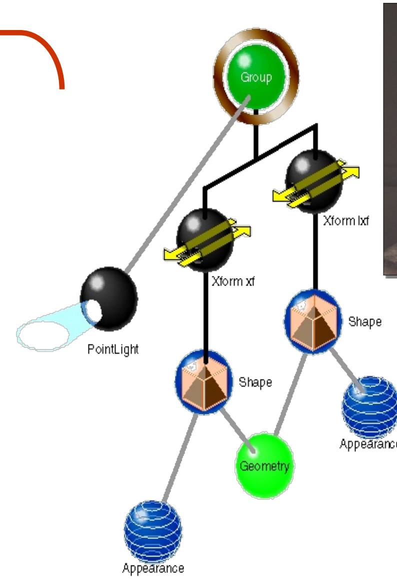
CPU

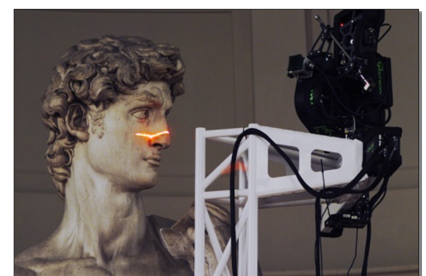
Anwendung → Geometrie-verarbeitung → Rasterisierung → Ausgabe

Eingabe graphischer Daten

Repräsentation von 3D Daten

- Grafische Primitive
- Transformationen
- Räumliche Datenstrukturen



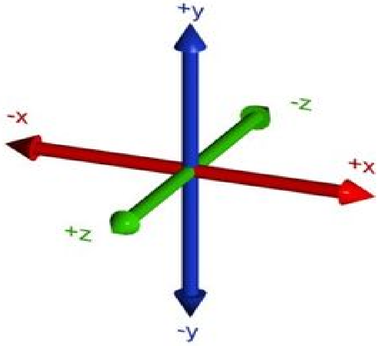


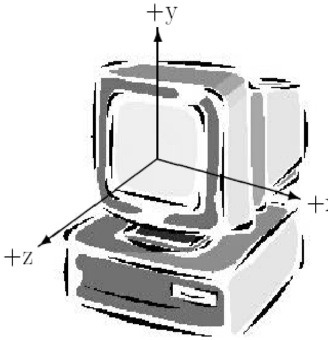
Überblick

- Wiederholung: Grafikpipeline
- **Transformationen und affine Abbildungen**
- Skalierung, Scherung, Rotation
- Projektion
- 3D-Interaktion

Koordinaten aus Sicht der Grafikpipeline

1. *Objektkoordinaten* (object coordinates):
   legen Lage von 3D-Objekten lokal fest
2. *Weltkoordinaten* (eye coordinates):
   beschreiben die gesamte Szene in 3D





3. *Projektionskoordinaten* (clip coordinates): erhält man nach Anwendung der Projektionstransformation (parallel oder perspektivisch)
4. *Normierte Koordinaten* (normalized device coordinates)
5. *Bildschirmkoordinaten* (window coordinates): stellen Szene in Fenster einer gewählten Größe und Position dar

Transformationen in der Grafikpipeline

Überblick und Vergleich mit „Fotografieren“

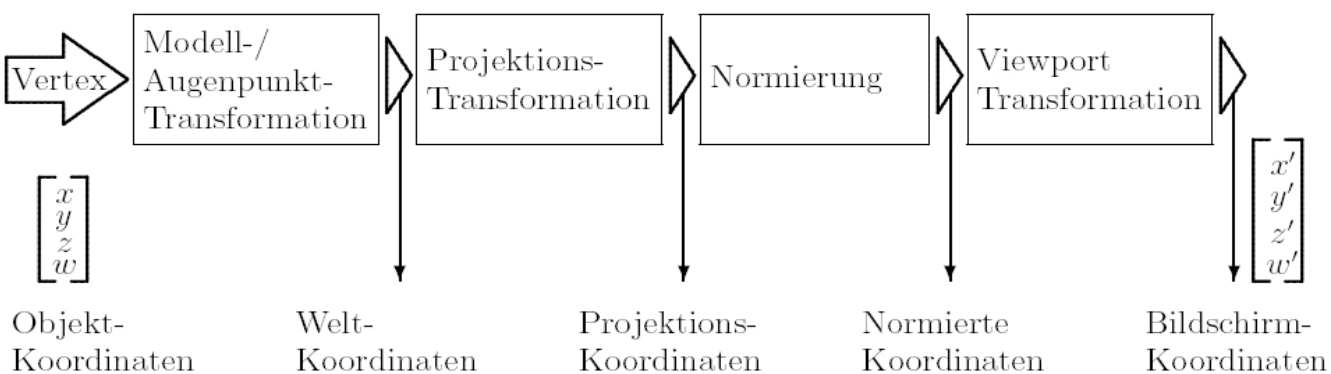
1. *Modelling* Transformations: ordne 3D-Objekte (Modelle) im Raum an und positioniere diese
2. *Viewing* Transformations: wähle Betrachterstandpunkt und positioniere diesen (default: Position: Ursprung & Blickrichtung: negative z-Achse)
3. *Projection* Transformations: projiziere Viewing Volume (sichtbarer Ausschnitt der Szene) in 2D
4. *Viewport* Transformations: wandle in Bildschirmkoordinaten um

Bereite Foto vor: ordne Personen und Objekte an

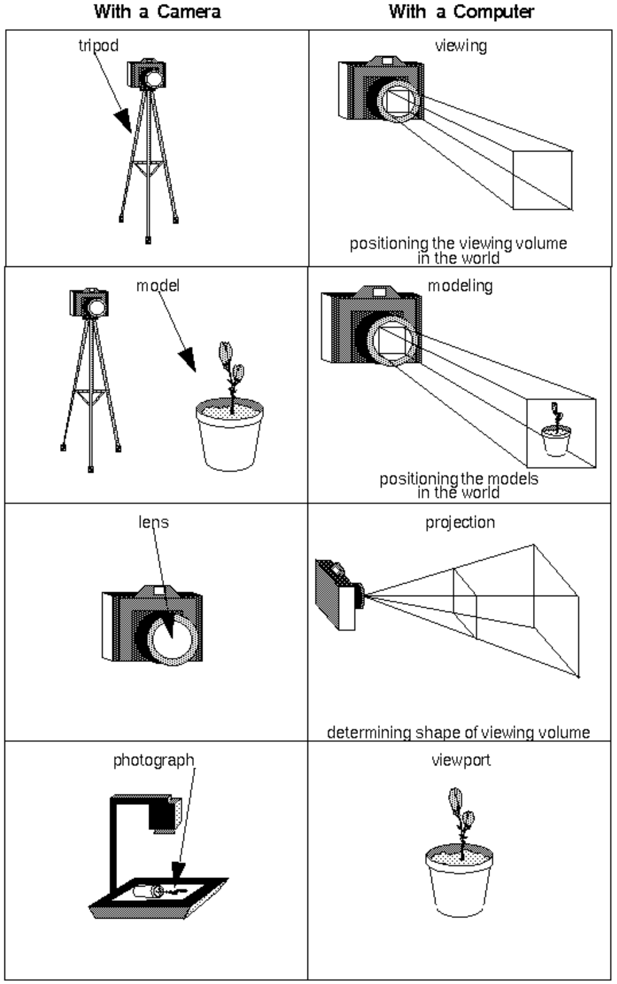
Positioniere die Kamera und justiere diese auf einem Stativ

Zoome, um den gewünschten Bildausschnitt festzulegen

Bestimme das Format in dem das Bild ausgedruckt werden soll



Transformationen in der Grafikpipeline
Vergleich mit „Fotografieren"



Transformationen
Positionierung von Objekten und Primitiven

Absolute Koordinaten?

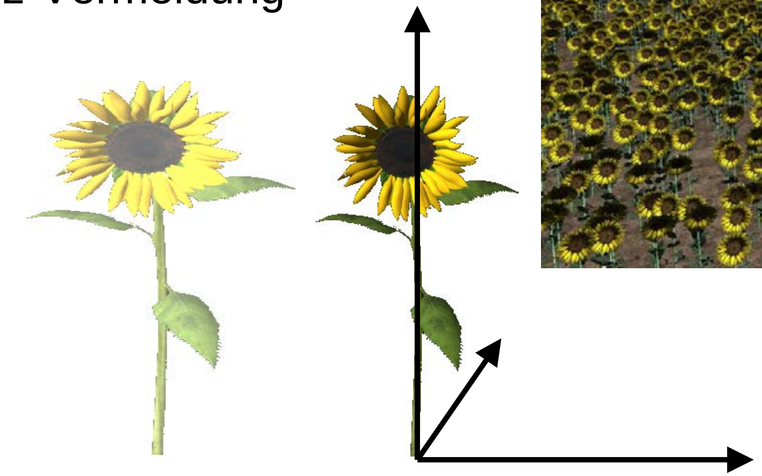
Transformationen
Positionierung von Objekten und Primitiven

Absolute Koordinaten?

- 28.000 Sonnenblumen
- 11 verschiedene Modelle
- Je 35.000 Dreiecke

Transformation von 3D-Objekten

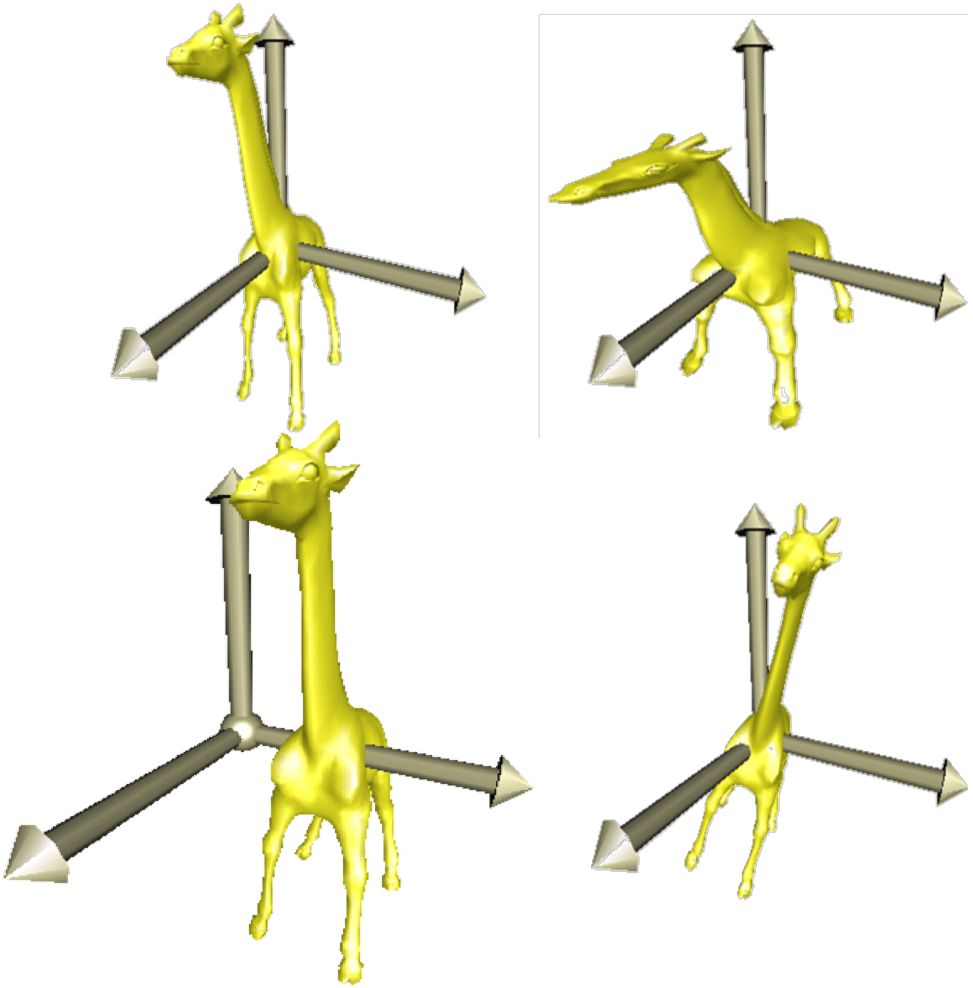
- Flexibilität
- verschiedene Instanzen eines 3D-Objekts
- Redundanz-Vermeidung



Transformationen
Positionierung von Objekten und Primitiven

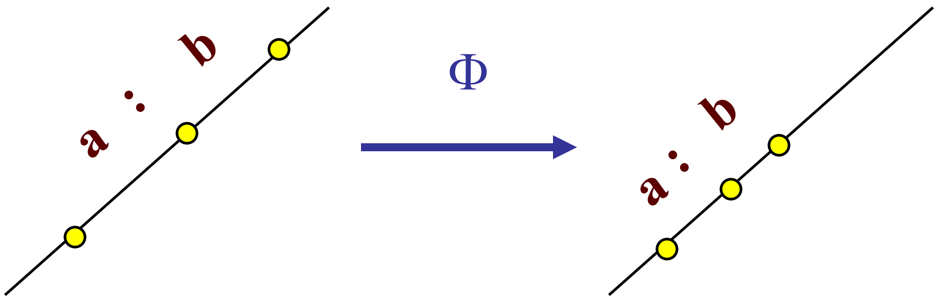
Affine Transformation/Abbildung

- Translation
- Rotation
- Skalierung
- Scherung



Affine Abbildungen
Eigenschaften

1. Bilden Geraden auf Geraden ab.
2. Beschränkte Objekte bleiben beschränkt.
3. Verhältnisse von Längen, Flächen, Volumen bleiben erhalten.
4. Parallele Objekte (Geraden, Ebenen, ...) bleiben parallel



Affine Abbildungen
…als lineare Abbildung

Eine Abbildung Φ: $R^n \rightarrow R^n$ heißt affine Abbildung,

wenn Φ in der Form $$\boxed{\Phi(\mathbf{v})=A(\mathbf{v})+I(\mathbf{b})}$$

darstellbar ist, wobei A, I lineare Abbildungen, I die Identitätsabbildung und $\mathbf{v}, \mathbf{b} \in R^n$ sind.

A heißt lineare Abbildung, wenn gilt:

$$\boxed{A(\lambda\mathbf{u} + \mu\mathbf{v}) = \lambda A(\mathbf{u}) + \mu A(\mathbf{v})}$$

für alle $u, v \in R^n$ und für alle $\lambda, \mu \in R$

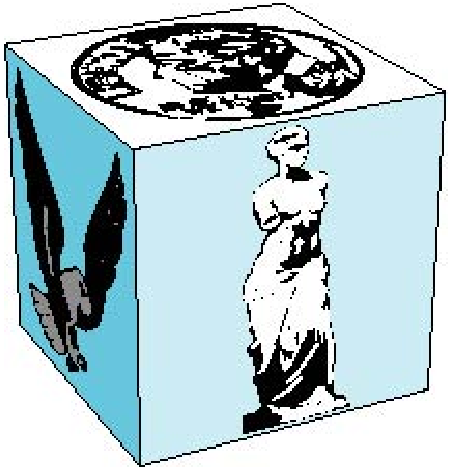
Affine Abbildungen
…als lineare Abbildung

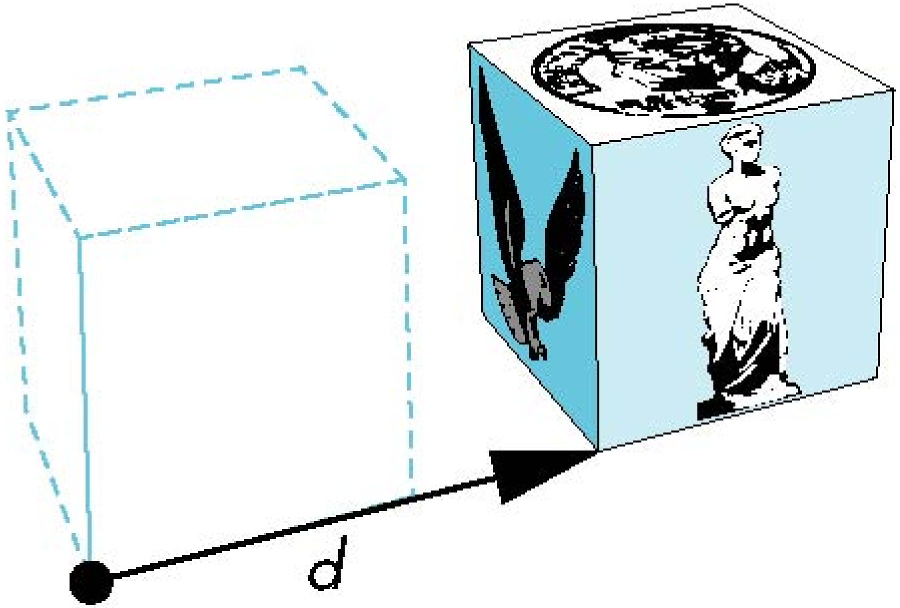
Affine Abbildungen setzen sich aus einer allgemeinen linearen Abbildung (dem multiplikativen Teil) und einer Translation (dem additiven Teil) zusammen.

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \underbrace{\begin{pmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{pmatrix}}_{\text{Lineare Abb.}} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \underbrace{\begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix}}_{\text{Translation}}$$

Die Translation ergibt sich aus der Multiplikation des Vektors (x_0, y_0, z_0) mit der Einheitsmatrix.

Translation





Objekt

Translation: jeder Punkt wird um den gleichen Vektor **d** verschoben

Translation

Transformation mit Translationsanteil $(x_0\ y_0\ z_0)^t$

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix}$$

➔ Repräsentation als 3x3-Matrix mit $a_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$

(Identität)

und Verschiebungsvektor

Kompaktere Darstellung möglich?

Homogene Koordinaten

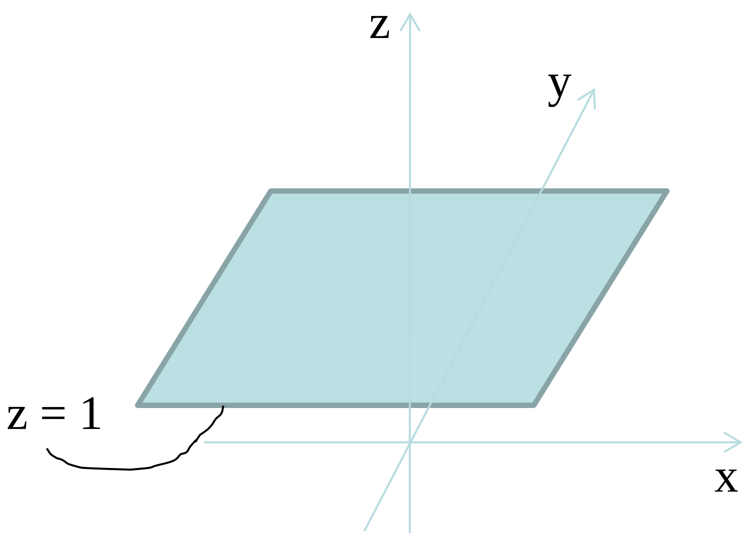
n-dimensionale (inhomogene) Koordinaten

werden zu

$(n+1)$-dimensionalen homogenen Koordinaten

z.B. 2D ➔ 3D:

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$



Homogene Koordinaten

definiere Äquivalenzklasse:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \rightarrow \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \cong \begin{pmatrix} sx \\ sy \\ sz \\ s \end{pmatrix} \qquad \begin{pmatrix} x \\ y \\ z \\ w \end{pmatrix} \rightarrow \begin{pmatrix} x/w \\ y/w \\ z/w \end{pmatrix}$$

Skalierungsfaktor *s* bzw. Gewicht *w* ungleich 0

Translation

Translation als Matrix-Multiplikation in homogenen Koordinaten

- Der lineare Teil einer Translation T ist die Identität
- Bsp. Translation um den Vektor $(x_0\ y_0\ z_0)^t$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & x_0 \\ 0 & 1 & 0 & y_0 \\ 0 & 0 & 1 & z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x + x_0 \\ y + y_0 \\ z + z_0 \\ 1 \end{bmatrix}$$

Homogene Koordinaten sind einfach deutbar als ein erweitertes Rechenschema!

Translation

Translation als Matrix-Multiplikation in homogenen Koordinaten

- Der lineare Teil einer Translation T ist die Identität
- Bsp. Translation um den Vektor $(x_0\ y_0\ z_0)^t$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & x_0 \\ 0 & 1 & 0 & y_0 \\ 0 & 0 & 1 & z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x + x_0 \\ y + y_0 \\ z + z_0 \\ 1 \end{bmatrix}$$

Homogene Koordinaten sind einfach deutbar als ein erweitertes Rechenschema!

Translation

Translation als Matrix-Multiplikation in homogenen Koordinaten

- Der lineare Teil einer Translation T ist die Identität
- Bsp. Translation um den Vektor $(x_0\ y_0\ z_0)^t$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & x_0 \\ 0 & 1 & 0 & y_0 \\ 0 & 0 & 1 & z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x + x_0 \\ y + y_0 \\ z + z_0 \\ 1 \end{bmatrix}$$

Homogene Koordinaten sind einfach deutbar als ein erweitertes Rechenschema!

Translation

TECHNISCHE UNIVERSITÄT DARMSTADT

Translation als Matrix-Multiplikation in homogenen Koordinaten

- Der lineare Teil einer Translation T ist die Identität
- Bsp. Translation um den Vektor $(x_0\ y_0\ z_0)^t$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & x_0 \\ 0 & 1 & 0 & y_0 \\ 0 & 0 & 1 & z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x + x_0 \\ y + y_0 \\ z + z_0 \\ 1 \end{bmatrix}$$

Homogene Koordinaten sind einfach deutbar als ein erweitertes Rechenschema!

MATHEMATICAL AND APPLIED VISUAL COMPUTING

Translation

Translation als Matrix-Multiplikation in homogenen Koordinaten

- Der lineare Teil einer Translation T ist die Identität
- Bsp. Translation um den Vektor $(x_0\; y_0\; z_0)^t$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & x_0 \\ 0 & 1 & 0 & y_0 \\ 0 & 0 & 1 & z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x + x_0 \\ y + y_0 \\ z + z_0 \\ 1 \end{bmatrix}$$

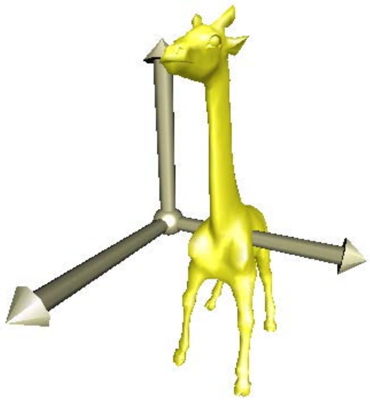
Homogene Koordinaten sind einfach deutbar als ein erweitertes Rechenschema!

Affine Abbildungen
Matrizenschreibweise

Allgemein lässt sich jede 3D affine Abbildung durch eine 4x4-Matrix ausdrücken

- Linke obere 3x3-Submatrix: Rotation, Skalierung etc.
- Rechte Spalte: Translation

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & x_0 \\ a_{21} & a_{22} & a_{23} & y_0 \\ a_{31} & a_{32} & a_{33} & z_0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$



3D affine Abbildungen werden durch homogene (erweiterte) 4x4-Matrizen beschrieben

- Einheitliche Darstellung ➔ einfache Implementierung
- Hintereinanderausführung verschiedener Transformationen ➔ nur Multiplikation der Matrizen

$$p' = A_1 \cdot p$$

$$p'' = A_2 \cdot p'$$

$$\vdots$$

$$p^n = A_n \cdot p^{n-1}$$

➔ $$p^n = A_n \cdot \ldots \cdot A_2 \cdot A_1 \cdot p$$

Überblick

- Wiederholung: Grafikpipeline
- Transformationen und affine Abbildungen
- **Skalierung, Scherung, Rotation**
- Projektion

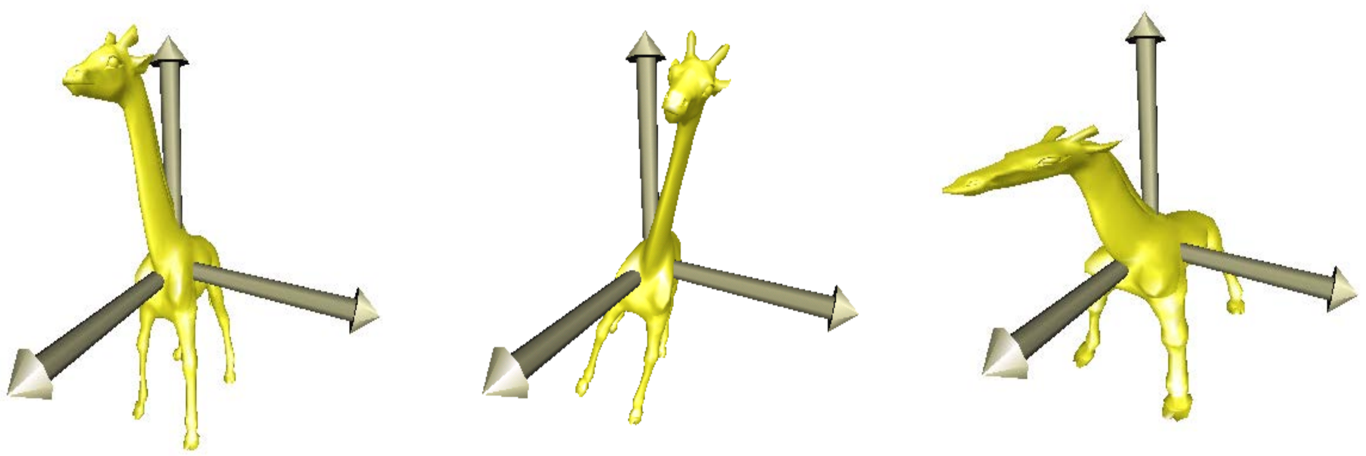
Skalierung, Scherung und Rotation

Die affinen Abbildungen *Skalierung*, *Scherung*, *Rotation* lassen den Ursprung invariant

Sie besitzen keinen Translationsanteil

Es sind genau die linearen Transformationen

3x3-Matritzen wären ausreichend



Skalierung, Scherung und Rotation

Homogene Form

$$\begin{bmatrix} \boxed{\begin{matrix} a_{11} \\ a_{21} \\ a_{31} \end{matrix}} & \begin{matrix} a_{12} \\ a_{22} \\ a_{32} \end{matrix} & \begin{matrix} a_{13} \\ a_{23} \\ a_{33} \end{matrix} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Dabei bestimmen die Bilder der Basisvektoren $\boxed{(1\ 0\ 0)^t}$, $(0\ 1\ 0)^t$, $(0\ 0\ 1)^t$ eine lineare Abbildung

Skalierung, Scherung und Rotation

Homogene Form

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 \\ a_{21} & a_{22} & a_{23} & 0 \\ a_{31} & a_{32} & a_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Dabei bestimmen die Bilder der Basisvektoren $(1\ 0\ 0)^t$, $\boxed{(0\ 1\ 0)^t}$, $(0\ 0\ 1)^t$ eine lineare Abbildung

Skalierung, Scherung und Rotation

Homogene Form

$$\begin{bmatrix} a_{11} & a_{12} & \boxed{a_{13}} & 0 \\ a_{21} & a_{22} & \boxed{a_{23}} & 0 \\ a_{31} & a_{32} & \boxed{a_{33}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Dabei bestimmen die Bilder der Basisvektoren $(1\ 0\ 0)^t$, $(0\ 1\ 0)^t$, $\boxed{(0\ 0\ 1)^t}$ eine lineare Abbildung

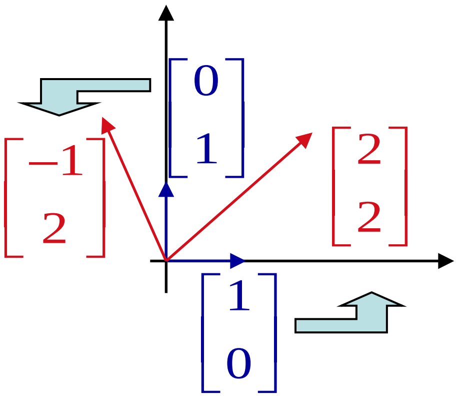
Skalierung, Scherung und Rotation

Multipliziert man die Ortsvektoren der Punkte von rechts, so stehen die Bilder der Basisvektoren der linearen Abbildung A in den Spalten der A beschreibenden Matrix.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \end{bmatrix}$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} a_{13} \\ a_{23} \\ a_{33} \end{bmatrix}$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} a_{12} \\ a_{22} \\ a_{32} \end{bmatrix}$$



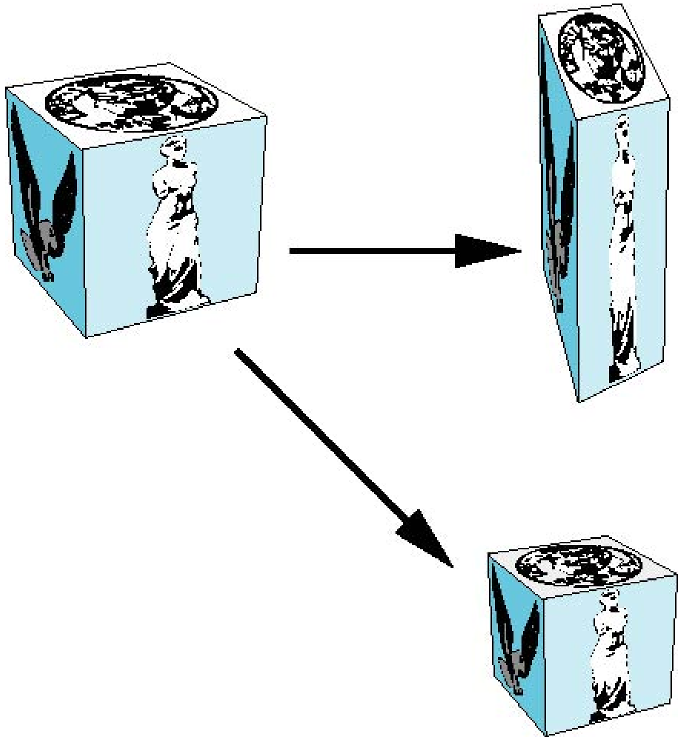
2D Beispiel:
Die lineare Abbildung in 2D wird also durch folgende Matrix beschrieben:

$$\begin{bmatrix} 2 & -1 \\ 2 & 2 \end{bmatrix}$$

Skalierung

Eine Skalierung S ergibt für die affinen Basisvektoren folgende Beziehung:

- $S((1\ 0\ 0)^t) = (s_1\ 0\ 0)^t$
- $S((0\ 1\ 0)^t) = (0\ s_2\ 0)^t$
- $S((0\ 0\ 1)^t) = (0\ 0\ s_3)^t$.



Skalierung wird durch eine Diagonalmatrix beschrieben!

Skalierung kann unterschiedlich sein für die einzelnen Komponenten

Skalierung

Die zugehörige 3 x 3 Matrix ergibt sich daher zu

$$\begin{pmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & s_3 \end{pmatrix}$$

In homogenen Koordinaten

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} s_1 & 0 & 0 & 0 \\ 0 & s_2 & 0 & 0 \\ 0 & 0 & s_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Skalierung

Der Sonderfall $s_1 = s_2 = s_3 = s$ bedeutet die gleiche Skalierung für alle Koordinaten

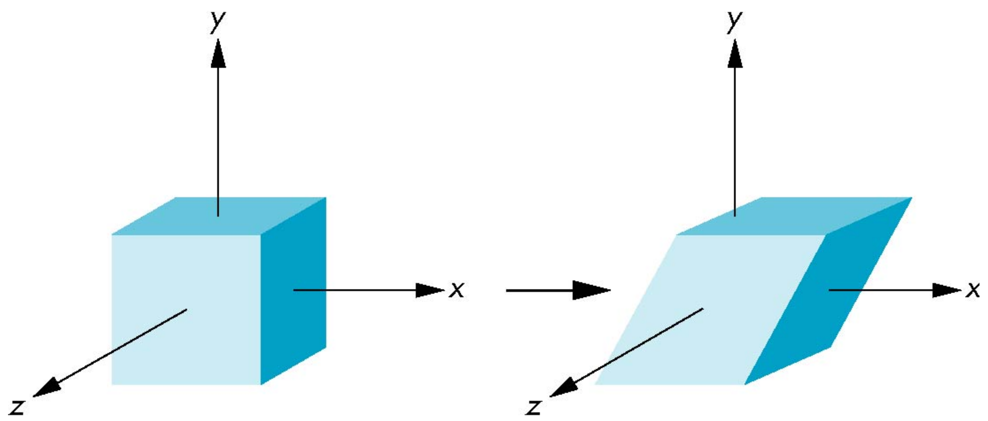
Die zugehörige homogene Matrix hat dann die Form

$$\begin{bmatrix} s & 0 & 0 & 0 \\ 0 & s & 0 & 0 \\ 0 & 0 & s & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cong \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \frac{1}{s} \end{bmatrix}$$

Scherung

Eine Scherung SH ergibt für die affinen Basisvektoren folgende Beziehung

- $SH((1\ 0\ 0)^t) = (1\ s_1\ s_3)^t$
- $SH((0\ 1\ 0)^t) = (s_2\ 1\ s_4)^t$
- $SH((0\ 0\ 1)^t) = (s_5\ s_6\ 1)^t$



Hier: nur $s_2 \neq 0$

Scherung

Die dazugehörige 3 x 3 Matrix wird daher zu

$$\begin{pmatrix} 1 & s_2 & s_5 \\ s_1 & 1 & s_6 \\ s_3 & s_4 & 1 \end{pmatrix}$$

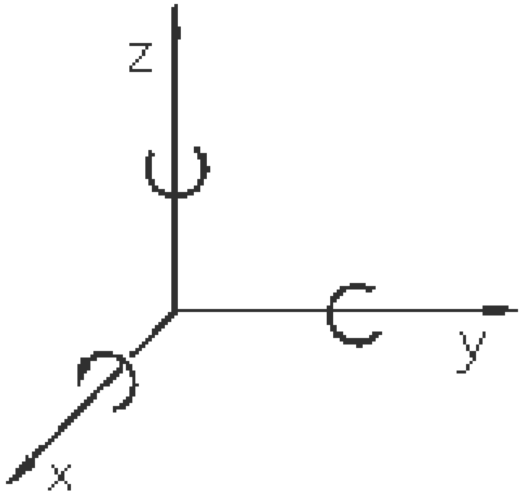
In homogenen Koordinaten folgt

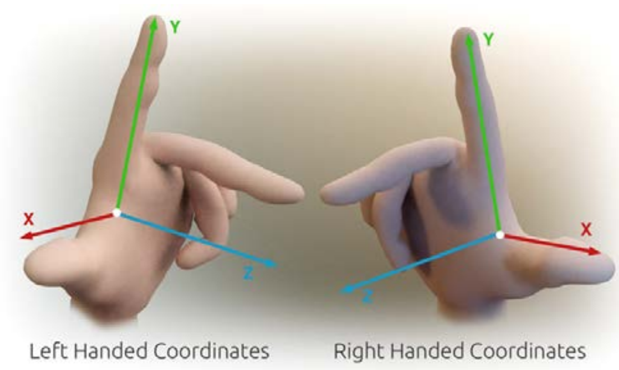
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & s_2 & s_5 & 0 \\ s_1 & 1 & s_6 & 0 \\ s_3 & s_4 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

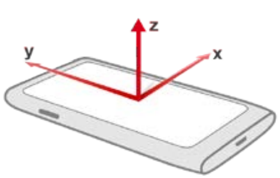
Rotation

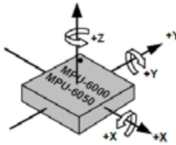
Die Drehwinkel sind in dem **Rechts**system x,y,z immer positiv:

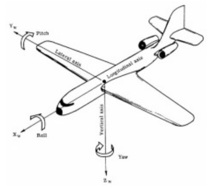
(https://en.wikipedia.org/wiki/Cartesian_coordinate_system)

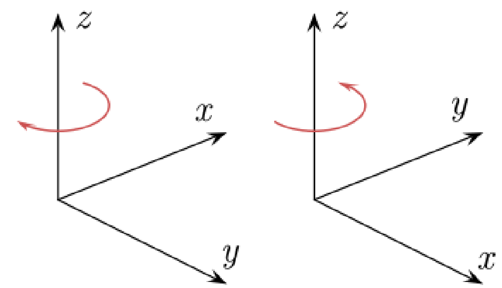








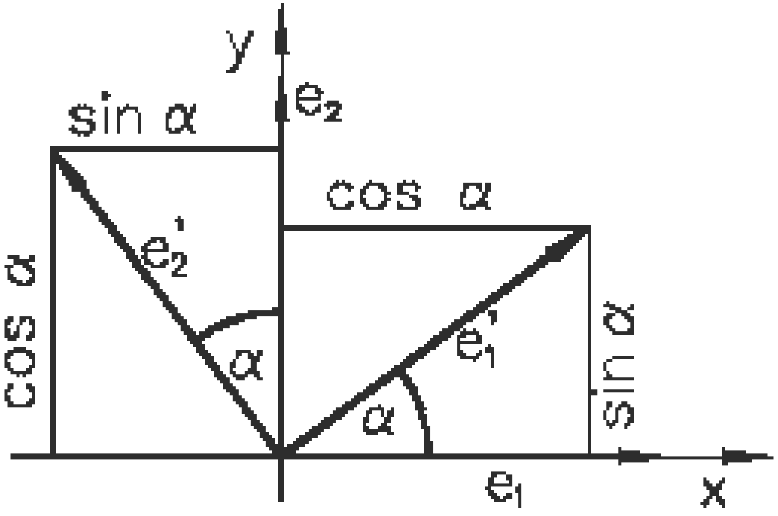




Rotation

Eine Rotation R_α um den Winkel α um die z-Achse in mathematisch positive Richtung ergibt für die Basisvektoren folgende Beziehung:

- $R_\alpha((1\ 0\ 0)^t) = (\cos\alpha, \sin\alpha, 0)$
- $R_\alpha((0\ 1\ 0)^t) = (-\sin\alpha, \cos\alpha, 0)$
- $R_\alpha((0\ 0\ 1)^t) = (0, 0, 1)$



Rotation

Die zugehörige 3 x 3 Matrix ergibt sich daher zu

$$\begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

In homogenen Koordinaten folgt für die Rotation R_α um die z-Achse

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

Rotation

Bei Rotation R_α um die x-bzw. y-Achse ergeben sich analog folgende homogene Darstellungen.

- Drehung mit dem Winkel α um die x-Achse

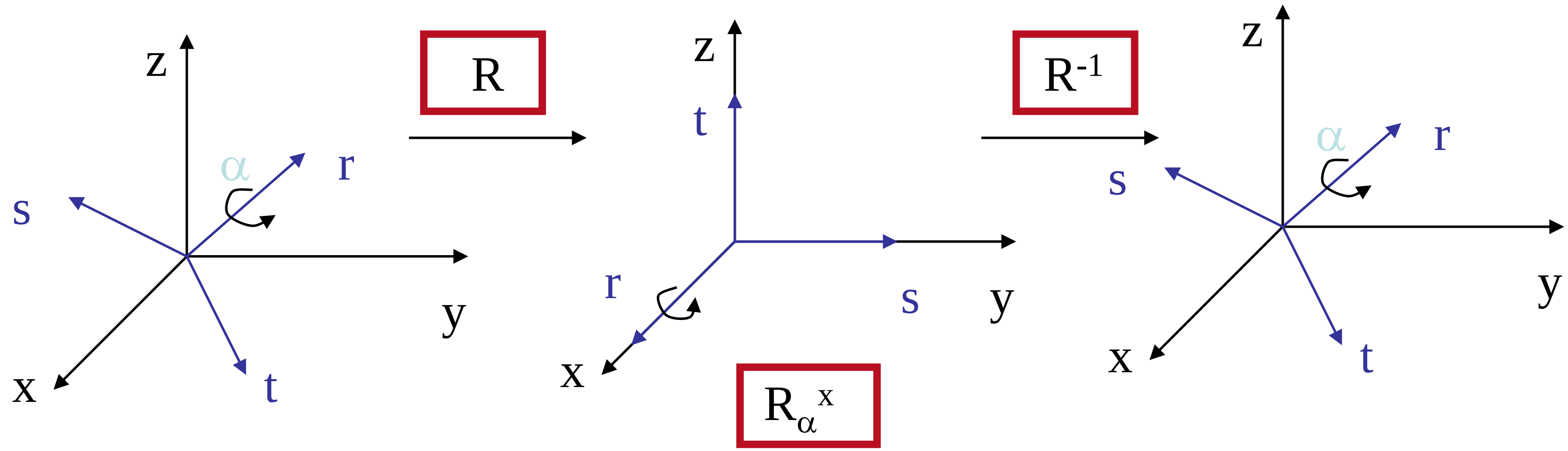
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- Drehung mit dem Winkel α um die y-Achse

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

Rotation um beliebige Achse

Drehung $R_\alpha(x,y,z)$ um beliebige Achse in Richtung des normierten Vektors
$\mathbf{r} = (x\ y\ z)^t$ um den Winkel α



$$R_{(x,y,z)} = R^{-1} R_\alpha{}^x R$$

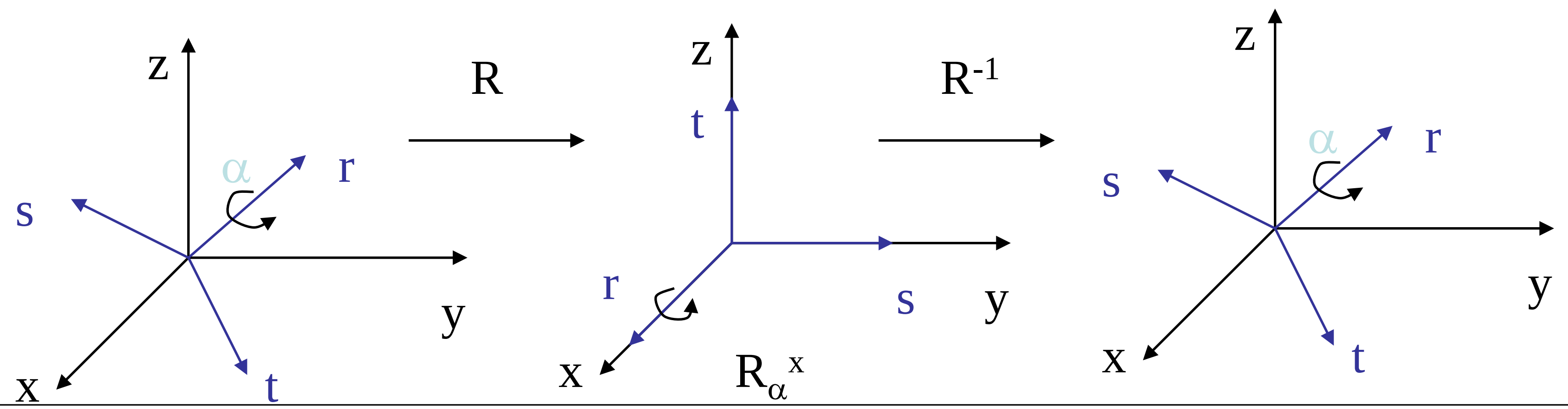
TECHNISCHE UNIVERSITÄT DARMSTADT

Orthonormale Basis $(\mathbf{r},\mathbf{s},\mathbf{t})$ bestimmen

- erster Basisvektor ist $\mathbf{r}$
- zweiter Basisvektor $\mathbf{s}$ soll senkrecht auf $\mathbf{r}$ stehen; Kreuzprodukt:

$$s = \frac{r \times e_x}{\left\| r \times e_x \right\|} \quad \text{oder (falls } r \| e_x \text{)} \quad s = \frac{r \times e_y}{\left\| r \times e_y \right\|}$$

- dritter Basisvektor $\mathbf{t} = \mathbf{r} \times \mathbf{s}$



MATHEMATICAL AND APPLIED VISUAL COMPUTING

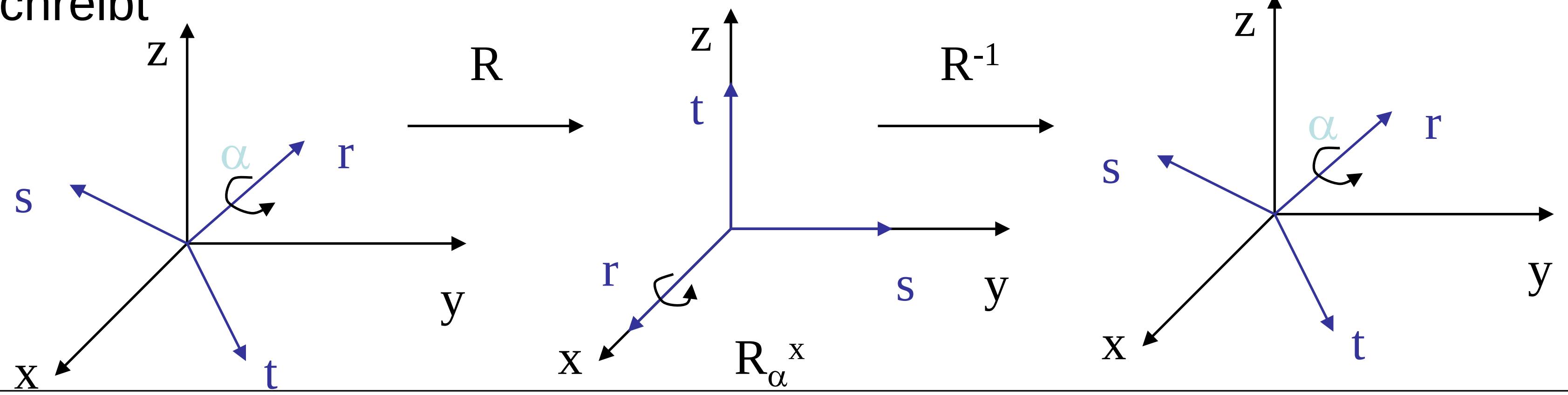
Rotation um beliebige Achse
Berechnung von R

Vektoren ($\mathbf{r},\mathbf{s},\mathbf{t}$) werden in die Spalten der Transformationsmatrix geschrieben (Sehe Folie 32!)

T-Matrix ist orthogonal und transformiert

- $\mathbf{e}_x \to \mathbf{r}$, $\mathbf{e}_y \to \mathbf{s}$, $\mathbf{e}_z \to \mathbf{t}$. (das ist R^{-1})
- Für orthonormierte Matrizen A gilt stets $A^{-1}=A^t$.

Also: R ergibt sich, indem man die Vektoren ($\mathbf{r},\mathbf{s},\mathbf{t}$) in die Zeilen von A schreibt

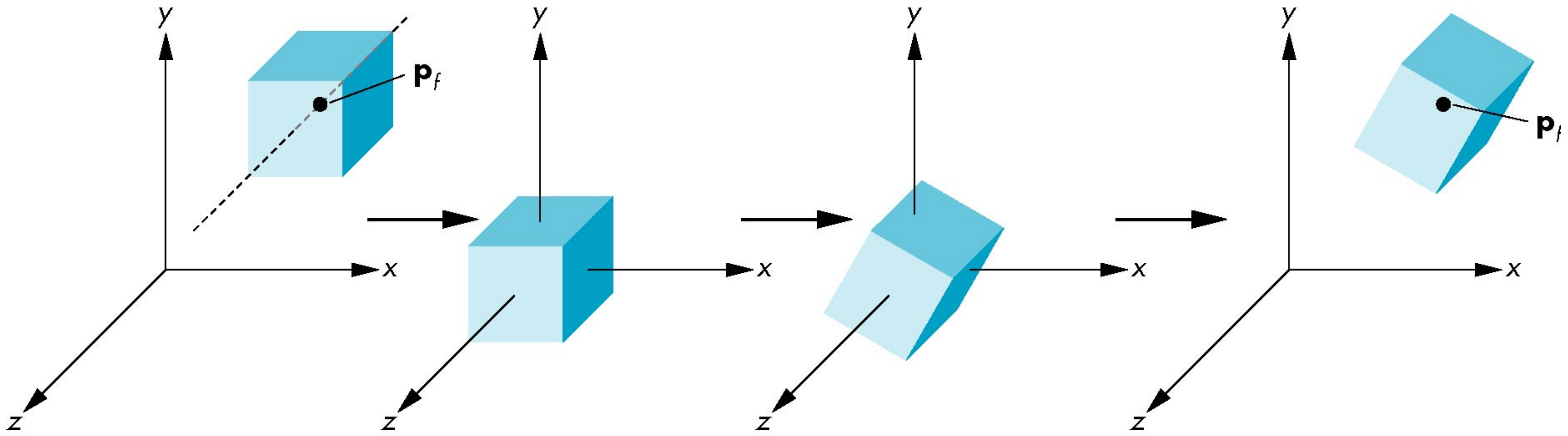


Rotation um beliebige Raumachse

Die diskutierten Rotationen lassen den Ursprung fest!

Rotationsachse durch eine beliebige Achse im Raum:

1. Verschiebung des Rotationszentrums in den Ursprung
2. anschließende Rotation und
3. Zurückverschiebung in das Rotationszentrum



Rotation um beliebige Raumachse

Beispiel

- Rotation in positiver Richtung um eine Achse durch den Punkt (x_0, y_0, z_0) und um den Winkel α
- Die Richtung der Rotationsachse sei die z-Richtung.

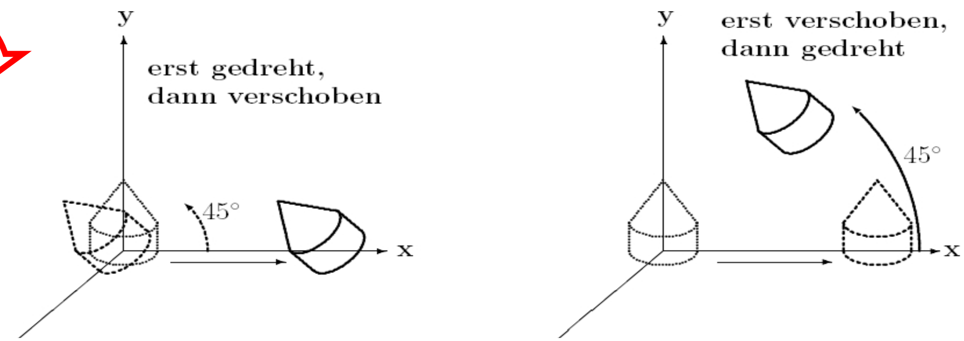
$$p' = \begin{bmatrix} 1 & 0 & 0 & x_0 \\ 0 & 1 & 0 & y_0 \\ 0 & 0 & 1 & z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 & 0 \\ \sin\alpha & \cos\alpha & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & -x_0 \\ 0 & 1 & 0 & -y_0 \\ 0 & 0 & 1 & -z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot p$$

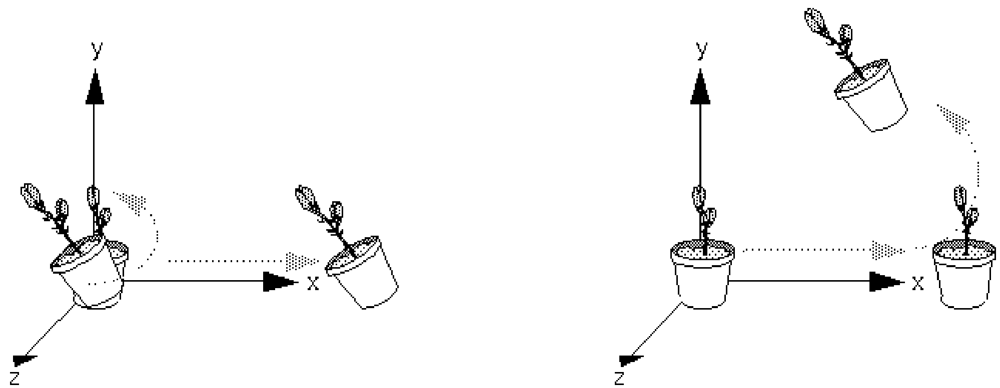
Transformationen - Nicht-Kommutativität

Reihenfolge der Transformationen darf i. A. nicht vertauscht werden

Grund: Matrizenmultiplikation ist nicht kommutativ, AB=BA gilt nicht für alle Matrizen A und B

Beispiel: TR versus RT





Transformationen
Bemerkung zum Aufwand

TECHNISCHE UNIVERSITÄT DARMSTADT

Für viele Eckpunkte (typisch in GDV) ist es günstiger einmalig die gesamte Transformationskette zu berechnen

Berechne einmalig das Produkt $A_n \ldots A_2A_1$ und wende die Eckpunkte darauf an

$$p' = A_1 \cdot p$$
$$p'' = A_2 \cdot p'$$
$$\vdots$$
$$p^n = A_n \cdot p^{n-1}$$

No!

$$p^n = A_n \cdot \ldots \cdot A_2 \cdot A_1 \cdot p$$

MATHEMATICAL AND APPLIED VISUAL COMPUTING

Überblick

- Wiederholung: Grafikpipeline
- Transformationen und affine Abbildungen
- Homogene Koordinaten
- Skalierung, Scherung, Rotation
- **Projektion**
- 3D-Interaktion

3D Grafik-Pipeline

CPU

GPU

Anwendung → Geometrie-verarbeitung → Rasterisierung → Ausgabe

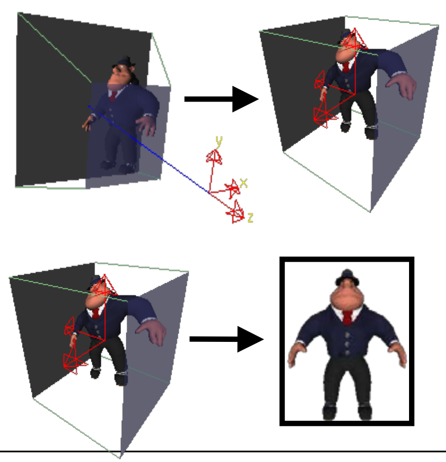
Simulation der Beleuchtung

Perspektivische Transformation

Clipping

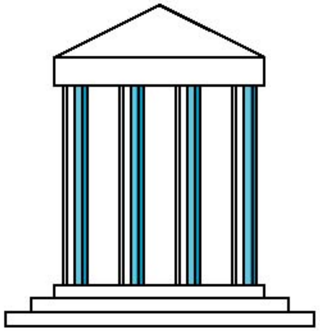
Culling

Projektion

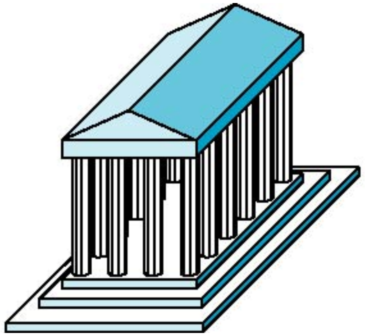


Klassische Projektionen

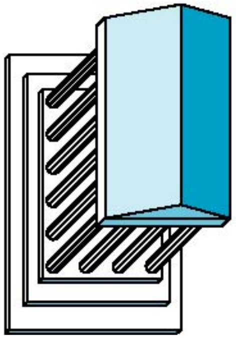
TECHNISCHE UNIVERSITÄT DARMSTADT



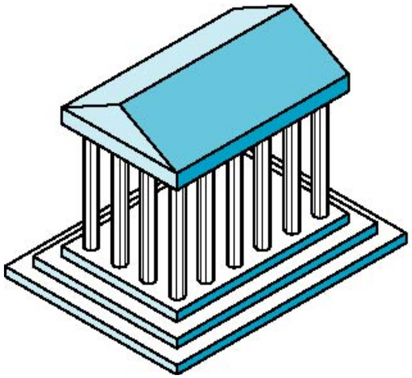
Aufriss (Frontansicht)



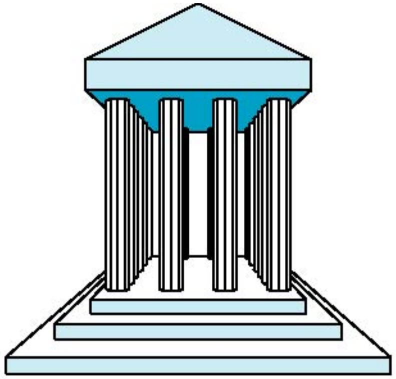
Kabinett-/ Kavallierspersperspektive



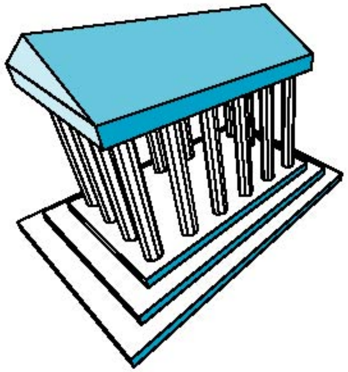
Allgemeine Parallelprojektion



Isometrische Perspektive



Zentralperspektive



Vogelperspektive

Projektive Abbildungen - Eigenschaften

können durch homogene 4×4-Matrizen beschrieben werden

1. Geraden werden auf Geraden abgebildet
2. Schnitte von Geraden bleiben erhalten
3. Flächen werden auf Flächen abgebildet
4. Reihenfolge von Punkten auf projektiven Geraden bleiben erhalten

Winkel werden verändert

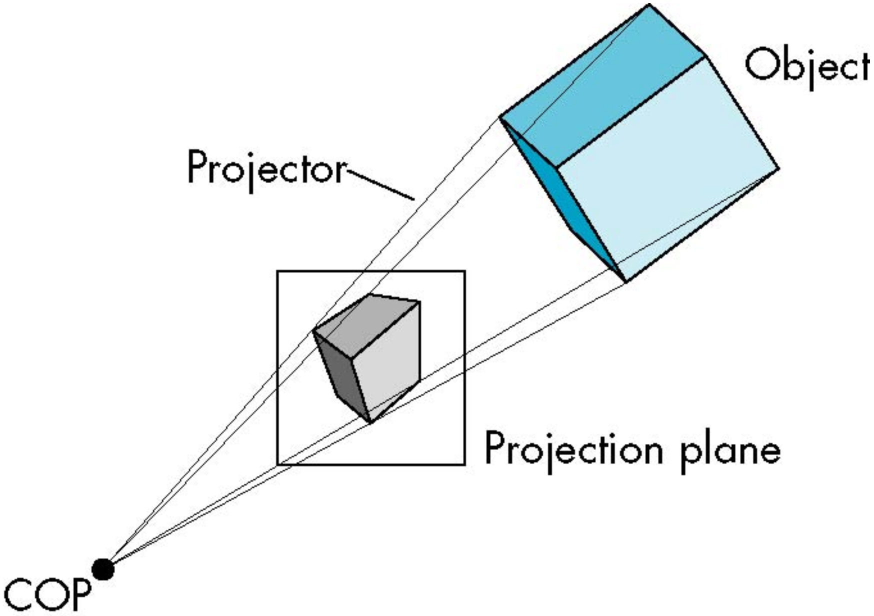
Parallelität geht oft verloren ➔ Parallelen schneiden sich in Fluchtpunkten

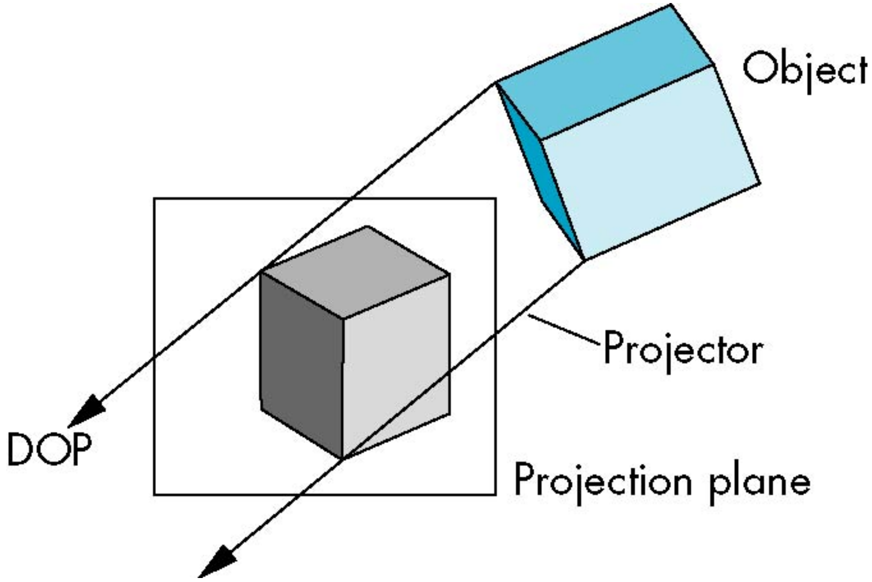
Rechtecke werden auf Vierecke transformiert

Perspektivische und parallele Projektionen

bei perspektivischer Projektion treffen sich die Strahlen im Augpunkt (Projektionszentrum)

bei parallelen Projektionen sind die Projektionsstrahlen parallel.

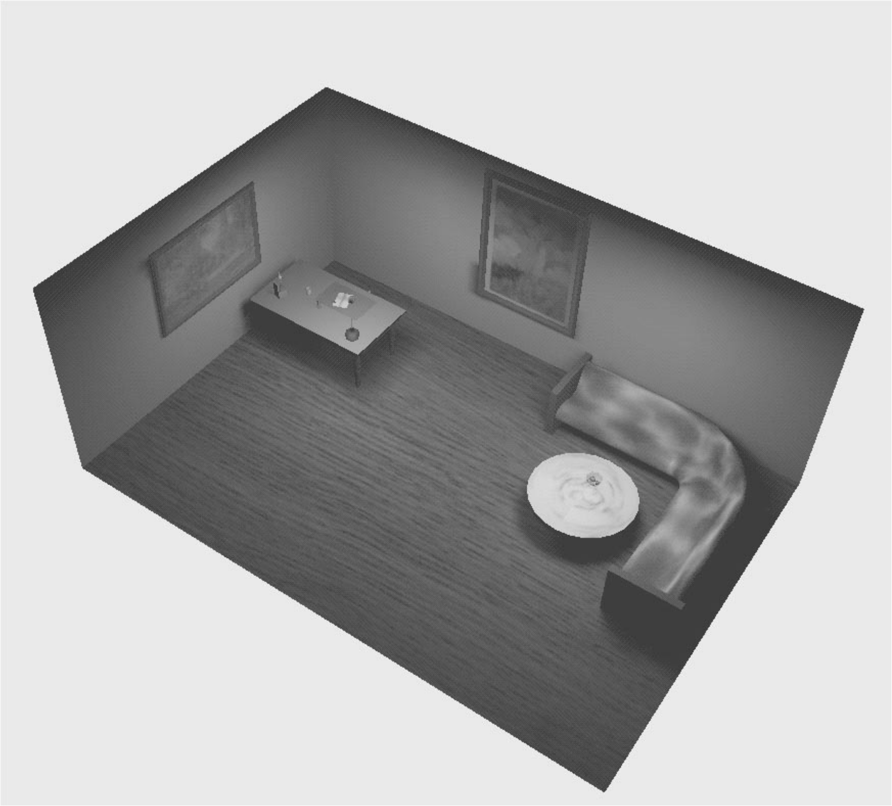




Vergleich: perspektivische und…

perspektivische Projektion(auch: Zentral-Projektion)

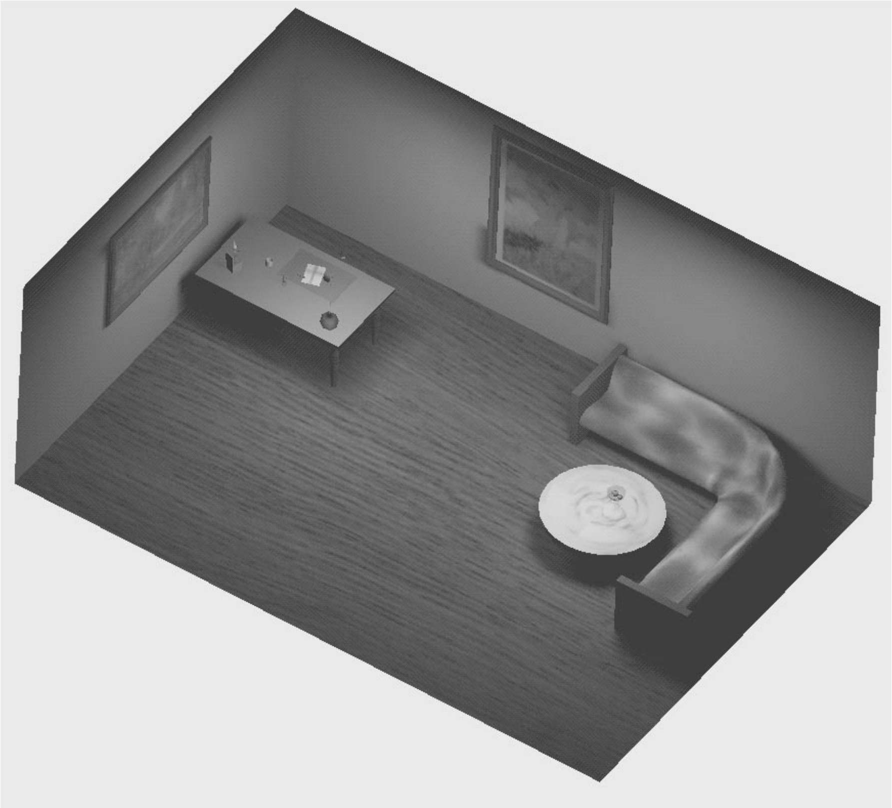
- vergleichbar dem fotografischen System, entspricht natürlicher Wahrnehmung des Menschen
- Abstand zwischen Objekten und Projektionsebene geht ein
- Längenverhältnisse ändern sich
- Winkel ändern sich
- parallele Geraden bleiben nicht parallel



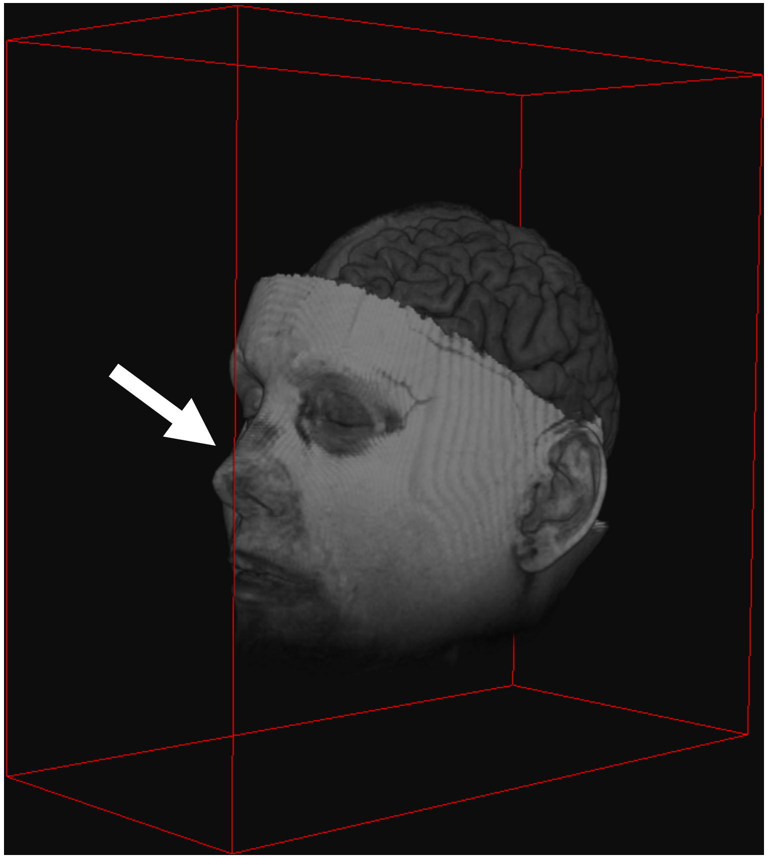
…parallele Projektion

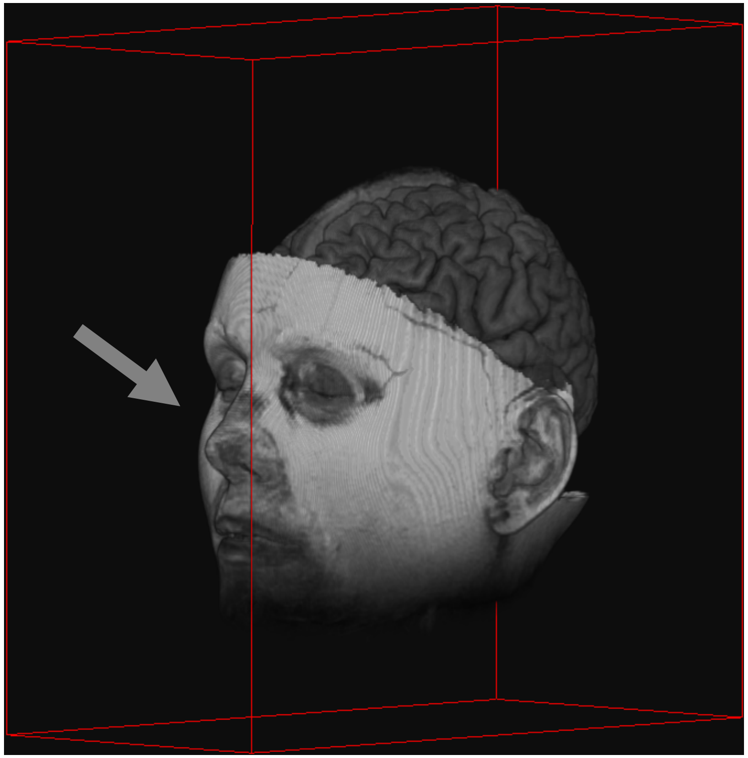
parallele Projektion (auch: orthografische Projektion)

- „weniger Realismus“ in der Darstellung
- Winkel ändern sich i.A. nicht
- parallele Geraden bleiben parallel



Medizin: parallele Projektion bevorzugt





Realismus = unverzerrte Bilder

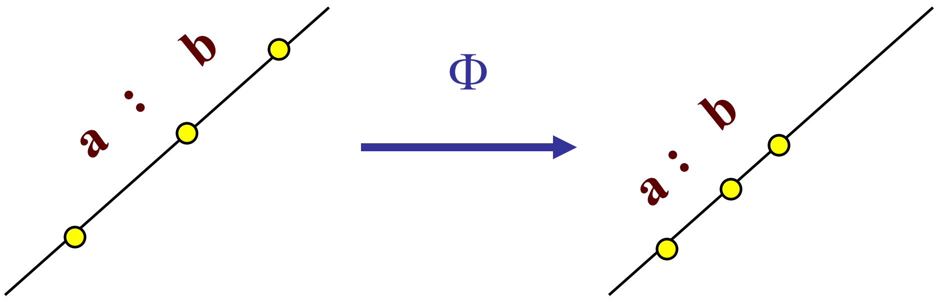
➔ Längen und Abstände sind relevant!

Perspektivische Verzerrung

Erinnerung

Affine Abbildungen – Eigenschaften

1. Bilden Geraden auf Geraden ab.
2. Beschränkte Objekte bleiben beschränkt.
3. Verhältnisse von Längen, Flächen, Volumen bleiben erhalten.
4. Parallele Objekte (Geraden, Ebenen, ...) bleiben parallel

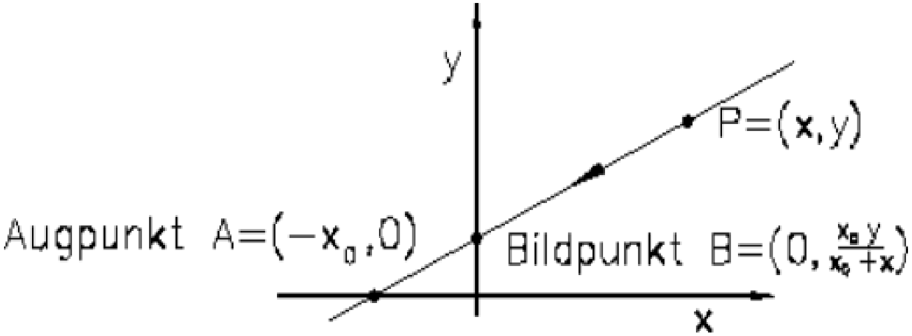


Perspektivische Abbildungen

Perspektivische Abbildungen sind **keine** affinen Abbildungen

- Längenverhältnisse sind nicht invariant
- Parallele Objekte bleiben nicht parallel

Vom Blickpunkt (Augpunkt, Beobachtungspunkt) weit entfernte Objekte werden kleiner dargestellt als nahe am Blickpunkt befindliche Objekte.



Sind der zu projizierende Punkt P=(x, y) und der Augpunkt A=($-x_0$, 0) gegeben, so gilt nach dem Strahlensatz für den Bildpunkt B=(0, y_0)

$$\frac{y_0}{y} = \frac{x_0}{x + x_0}$$

Perspektivische Abbildungen

Allgemein lautet mit $y_0 = y \cdot \frac{x_0}{x_0 + x}$

die Abbildung wie folgt: $\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} 0 \\ \frac{y \cdot x_0}{x_0 + x} \end{pmatrix}$

Daraus ergibt sich die homogene 3 × 3-Matrix (2D-Geometrie!)

$$\begin{bmatrix} 0 \\ \frac{x_0}{x + x_0} \cdot y \\ 1 \end{bmatrix} \cong \begin{bmatrix} 0 \\ x_0 \cdot y \\ x + x_0 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & x_0 & 0 \\ 1 & 0 & x_0 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \cong \frac{1}{x_0} \cdot \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ \frac{1}{x_0} & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Kanonisches Sichtvolumen

Die perspektivische Projektion wird in zwei Abbildungen zerlegt:

1. die perspektivische Transformation und
2. eine anschließende Parallelprojektion

Nach der perspektivischen *Transformation* ist das Sichtvolumen ein Würfel!

Durch Rotation (dh. der Augpunkt A=(-x_0, 0)) erreicht man, dass der Würfel achsenparallel wird.

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ \frac{1}{x_0} & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \frac{1}{x_0} & 0 & 1 \end{bmatrix}$$

Perspektivische Projektion | **Parallel-projektion** | **Perspektivische Transformation**

Kanonisches Sichtvolumen

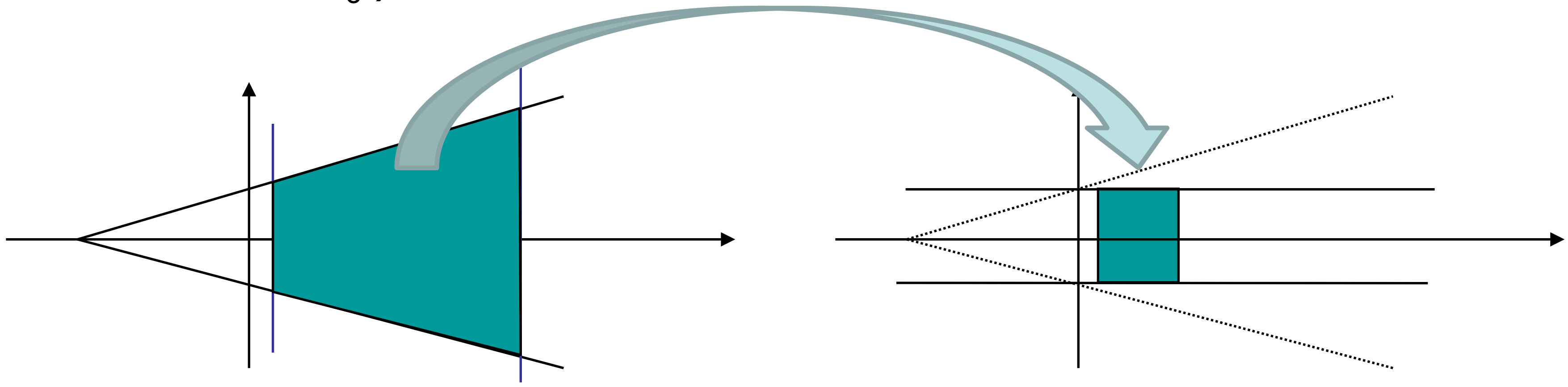
Parallelprojektion (Kamera im Unendlichen):

Sichtvolumen = Einheitswürfel

Perspektive: Sichtvolumen = Pyramide

Nach der perspektivischen *Transformation* ist das Sichtvolumen ein Würfel!

(NB: Gewicht $x+x_0$!)



Allgemeine perspektivische Transformation

Allgemeine perspektivische Transformation zu einem Fluchtpunkt in (x_0, y_0, z_0):

$$T_p \cdot \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \frac{1}{x_0} & \frac{1}{y_0} & \frac{1}{z_0} & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ \frac{x}{x_0} + \frac{y}{y_0} + \frac{z}{z_0} + w \end{bmatrix}$$

Die Richtungen von Parallelen zu den Koordinatenachsen werden auf die Achsen-Fluchtpunkte $[x_0, 0, 0, 0]^t$, $[0, y_0, 0, 0]^t$, $[0, 0, z_0, 0]^t$ abgebildet.

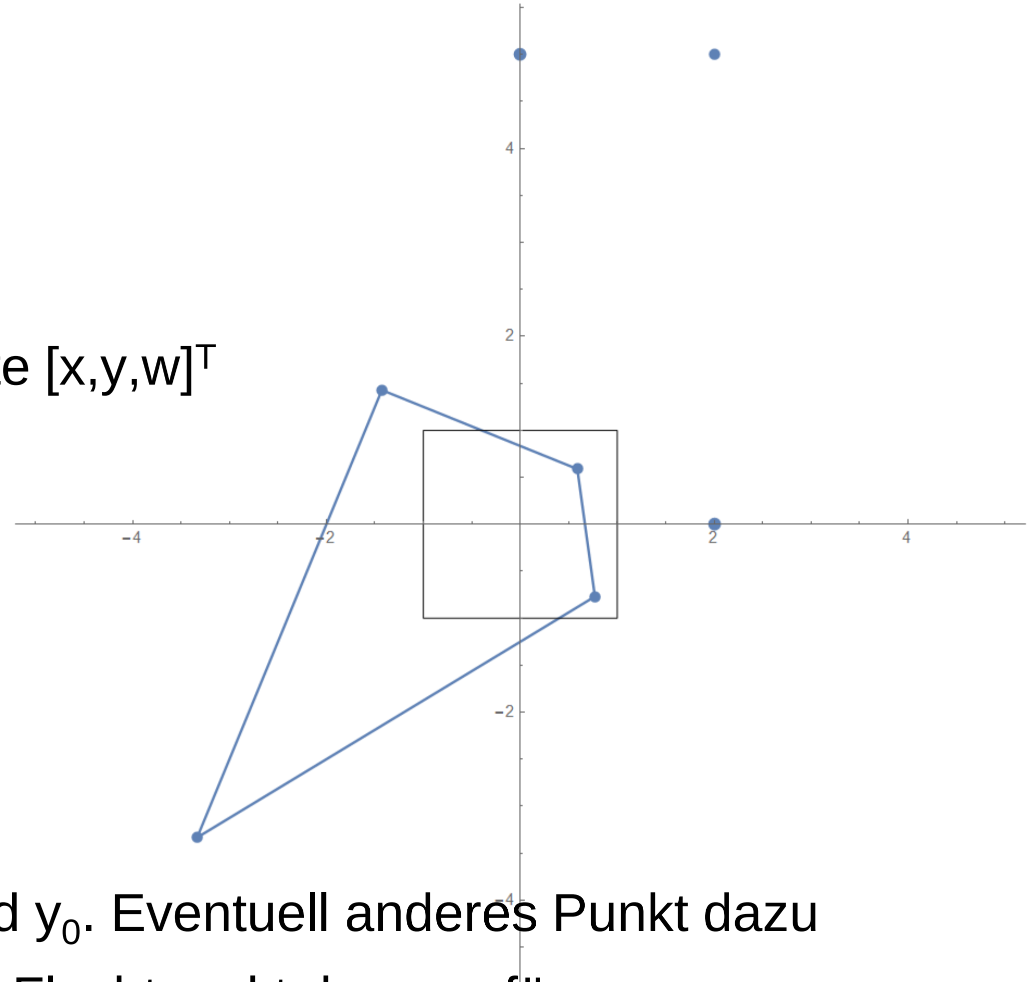
2D Beispiel

$x_0=2,\ y_0=5\ \ \text{->}\ \ M=\begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ \frac{1}{2} & \frac{1}{5} & 1\end{pmatrix}$

Quadrat in {□1,□1}

M □Eckpunkte -> neue Eckpunkte $[x,y,w]^T$

$\begin{Bmatrix}1\\1\\\frac{17}{10}\end{Bmatrix}\quad \begin{Bmatrix}-1\\1\\\frac{7}{10}\end{Bmatrix}\quad \begin{Bmatrix}1\\-1\\\frac{13}{10}\end{Bmatrix}\quad \begin{Bmatrix}-1\\-1\\\frac{3}{10}\end{Bmatrix}$

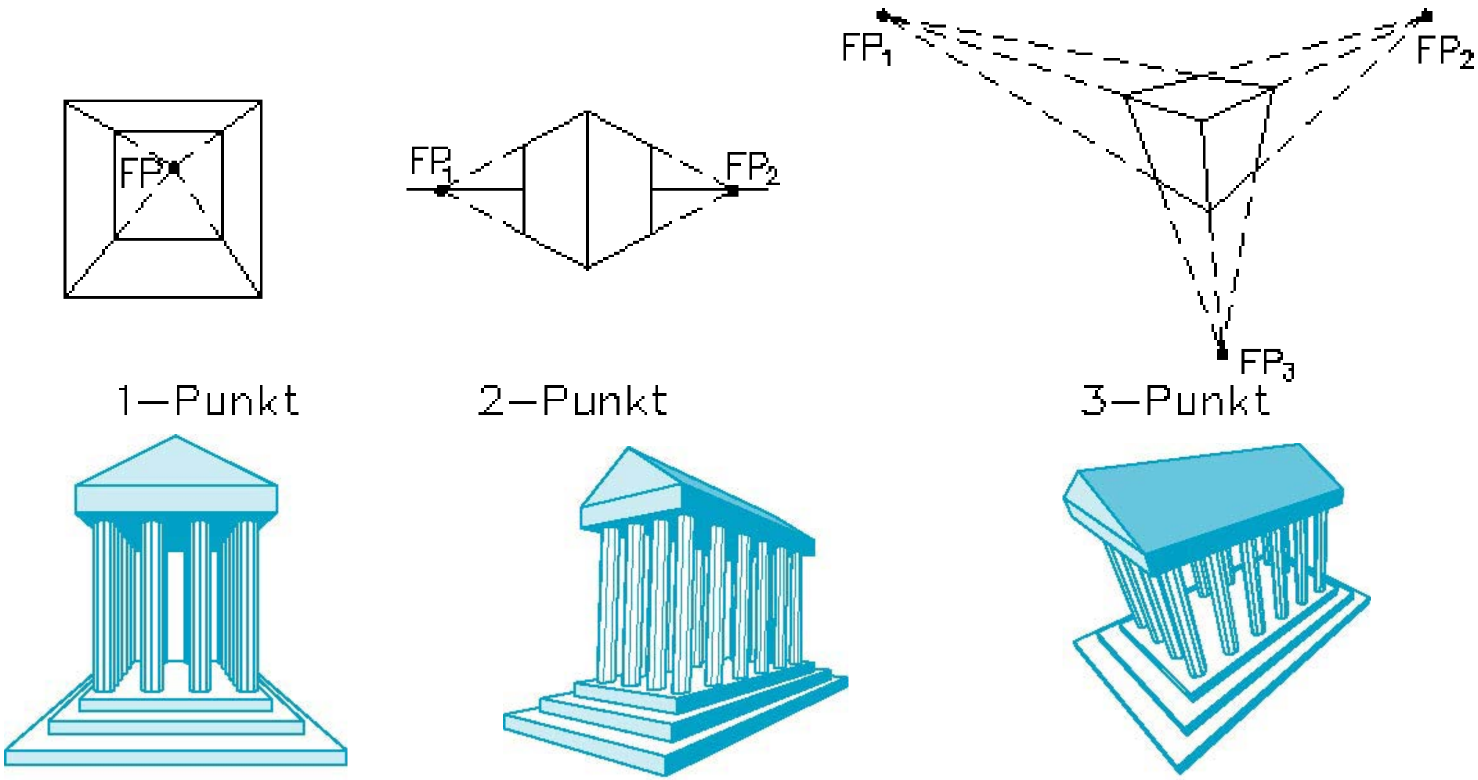


Bringe W->1

-> Sichtfläche geändert für x_0 und y_0. Eventuell anderes Punkt dazu nehmen, zB x_1=-2, y_1=-5 um 2. Fluchtpunkt dazu zu fügen.

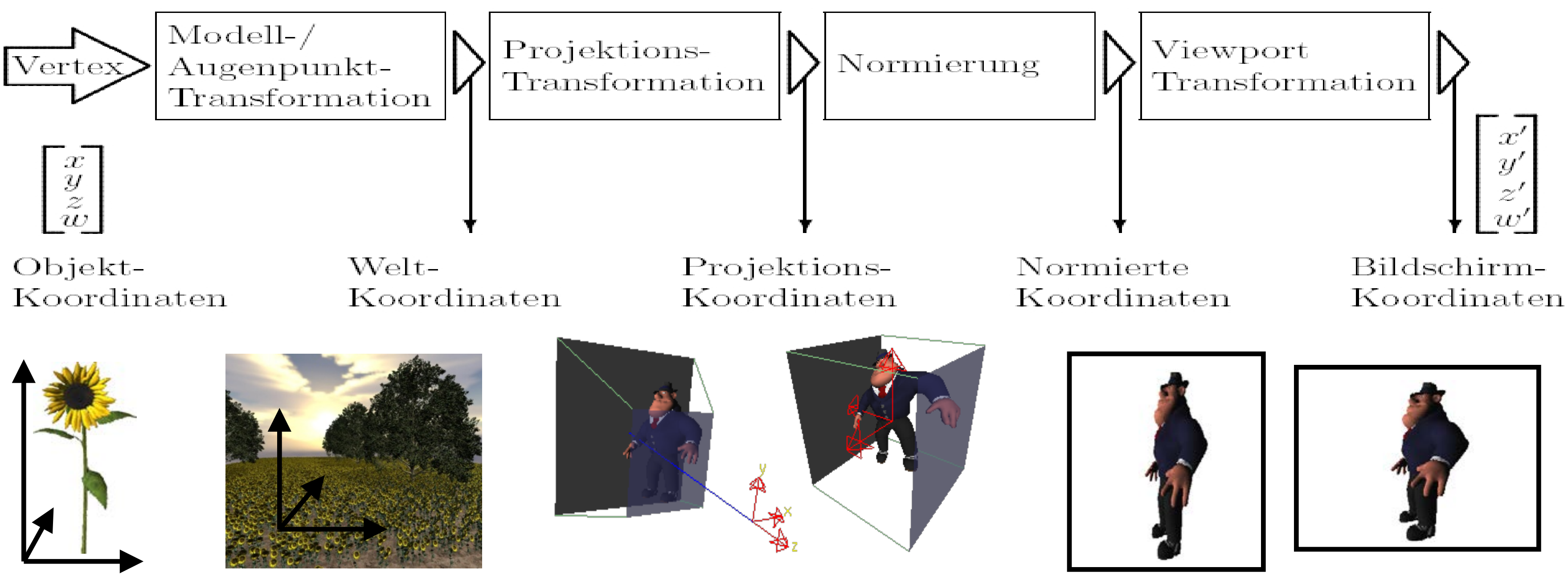
Ein-, Zwei- und Dreipunktperspektive

Manchmal braucht man mehrere Fluchtpunkte, entscheide dann welche Sichtfläche von welcher Punkt gesehen wird.



Transformationen in der Grafik-Pipeline

1. *Modelling* Transformations: ordne 3D-Objekte (Modelle) im Raum an und positioniere diese
2. *Viewing* Transformations: wähle Betrachterstandpunkt und positioniere diesen (default: Position: Ursprung & Blickrichtung: negative z-Achse)
3. *Projection* Transformations: projiziere Viewing Volume in 2D
4. *Viewport* Transformations: wandle in Bildschirmkoordinaten um



Zum Spielen

Zwei Sachen zu Transformationen:

http://apike.ca/prog_svg_transform_demo.html

Hier kann man die Transformationsmatrix direkt einstellen wie man möchte.

Ein kleines Windows Programm offline:

http://www.songho.ca/opengl/gl_transform.html

(etwas scrollen oder Direktlink

http://www.songho.ca/opengl/files/matrixModelView.zip)

Zeigt zwar eigentlich die openGL Transformationen, aber man sieht auch die entsprechende Matrix.

Überblick

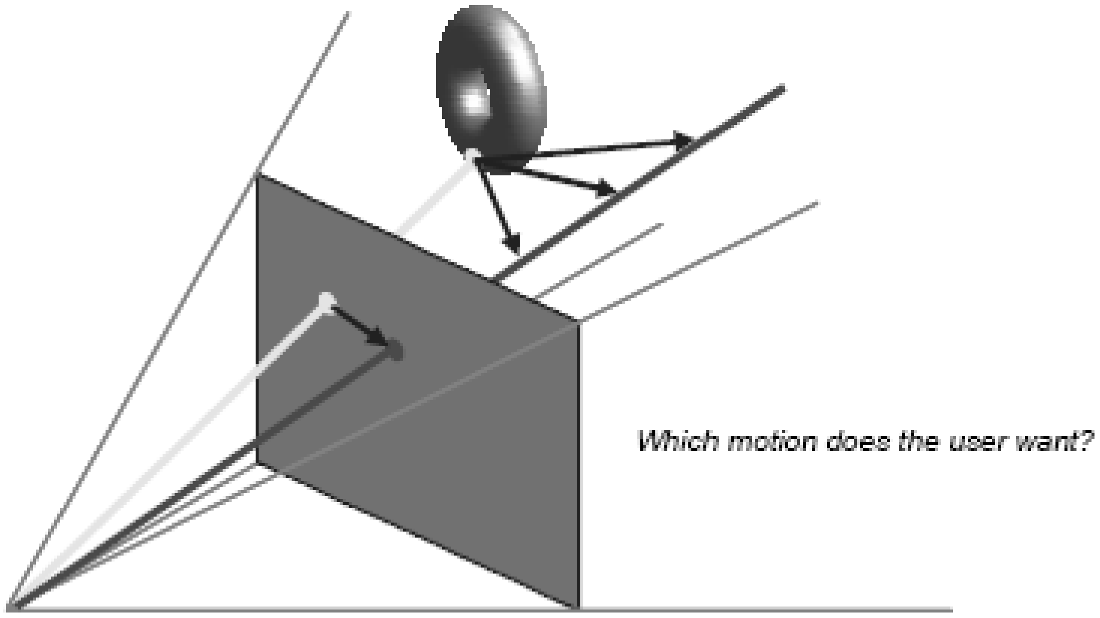
- Wiederholung: Grafikpipeline
- Transformationen und affine Abbildungen
- Homogene Koordinaten
- Skalierung, Scherung, Rotation
- Projektion
- **3D-Interaktion**

3D-Interaktion mit 2D-Eingabegeräten

Problem:

Welche Art der Bewegung möchte der Benutzer ausgeführt haben?

- Mehrdeutigkeit



3D-Interaktion mit 2D-Eingabegeräten

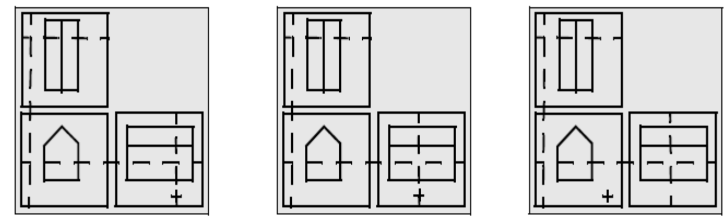
Ansätze:

Desktop

Multi-Window (Mehrfachauswahl)

Direktes 2D-Maus-Mapping

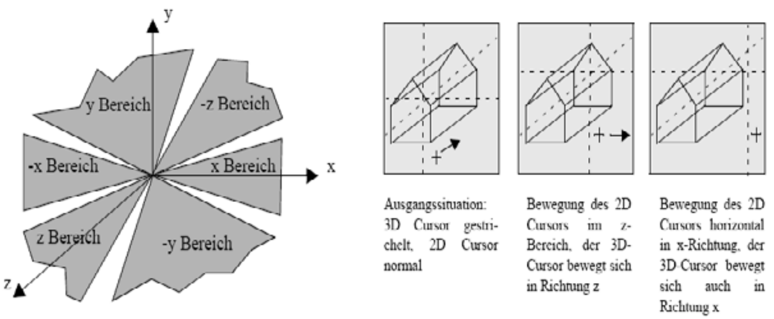
Manipulatoren



Um den gestrichelten 3D Cursor zu bewegen, wird er mit dem 2D-Mauszeiger gepickt

Bewegungen des Mauszeigers führen nun zur Translation des 3D Cursors in der Ebene.

Nach Loslassen des Mausknopfs kann der 2D Cursor wieder frei bewegt werden, ohne daß der 3D Cursor folgt.

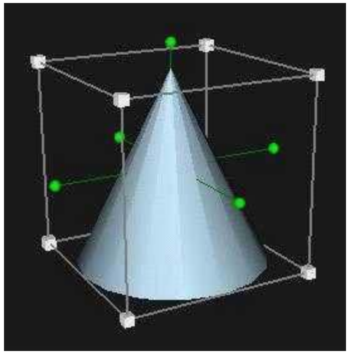


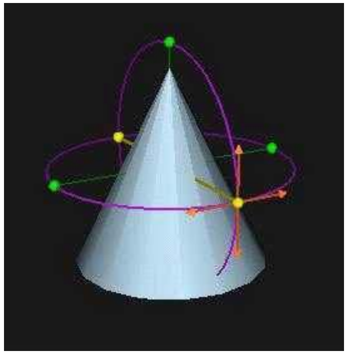
Ausgangssituation: 3D Cursor gestri-chelt, 2D Cursor normal

Bewegung des 2D Cursors im z-Bereich, der 3D-Cursor bewegt sich in Richtung z

Bewegung des 2D Cursors horizontal in x-Richtung, der 3D Cursor bewegt sich auch in Richtung x

Abbildung 3: 3D-Cursorsteuerung mit 2D-Maus nach Nielson und Olson [107]





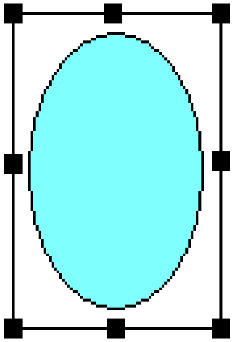
Manipulatoren

Häufig im Zweidimensionalen verwendet

- Kästen in Grafikprogrammen, mit denen skaliert, rotiert und verschoben werden kann
- Drag-and-Drop-Operationen

Immer häufiger auch im Dreidimensionalen

- Manipulatoren für Transformationen
- Navigation der Kamera

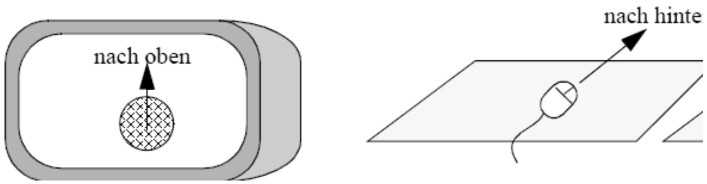


Manipulatoren

TECHNISCHE UNIVERSITÄT DARMSTADT

In 2D einfach zu verwenden (und zu implementieren)

- 1:1 Abbilding zwischen der Mauszeigerposition und dem „Knauf“ im 2D-Raum (räumliche Domäne)
- Schnitttests sind leicht zu implementieren
- Interpretation der Bewegung ist einfach
- Keine Probleme mit der perspektivischen Abbildung



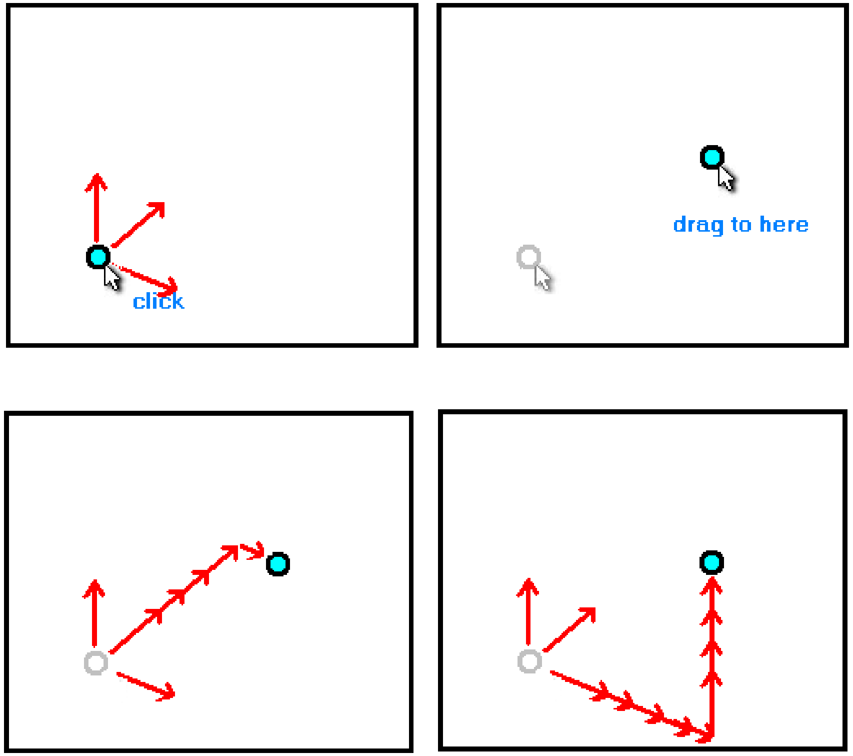
MATHEMATICAL AND APPLIED VISUAL COMPUTING

Manipulatoren

Was verändert sich in 3D?

Warum wird es schwieriger?

- Eine virtuelle 3D-Szene wird 2D angezeigt
- Mehrdeutigkeiten: Unendlich viele Möglichkeiten, die Cursorposition auf eine gerade Linie im 3D-Raum abzubilden
- Noch schwieriger wird es, wenn der 2D-Cursor bewegt wird



3D-Interaktion mit 3D-Widgets

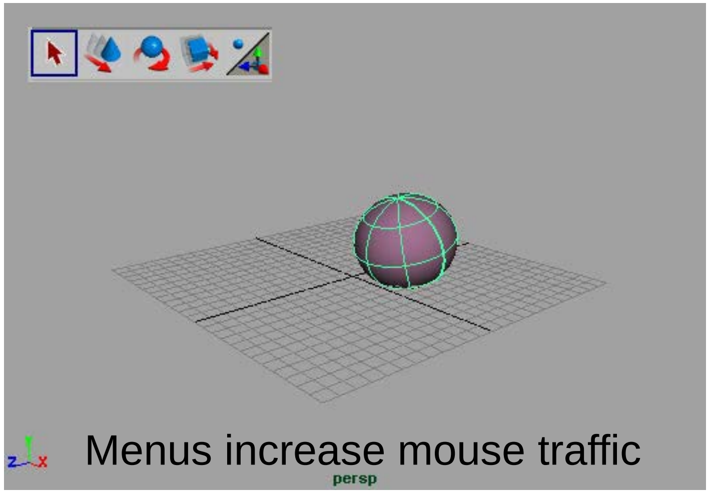
Was ist ein Manipulator?

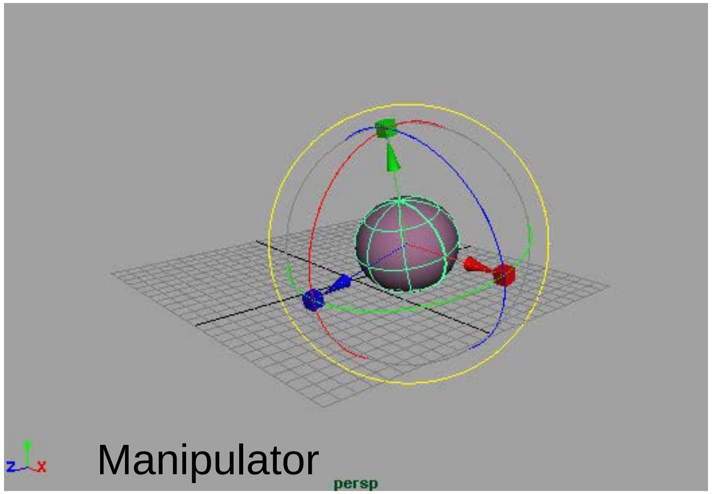
Eine visuelle grafische Repräsentation einer Operation oder der Status eines Objekts, der zusammen mit dem Objekt selbst angezeigt wird.

Der Status bzw. Die Operation kann durch Klicken und Bewegen (Dragging) der grafischen Elements (Handle) des Manipulators kontrolliert werden

➔Der Zeiger bleibt innerhalb der Szene

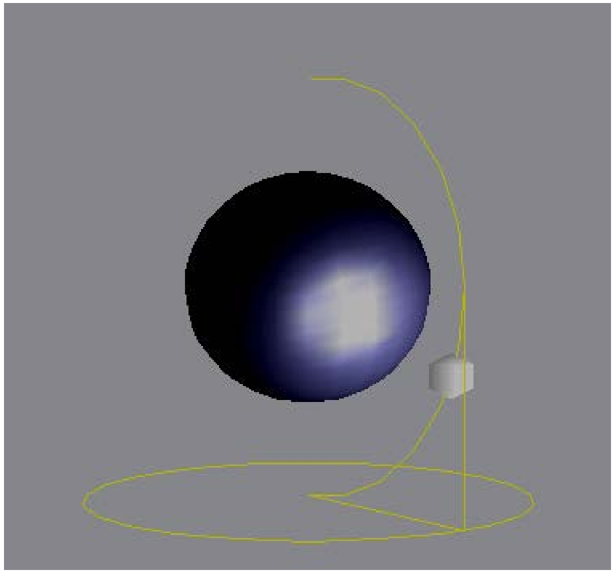
➔Reduziert Mausbewegungen



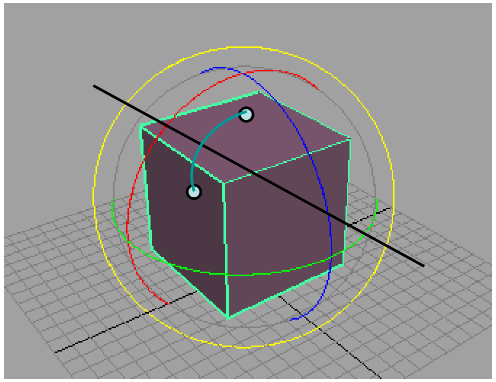


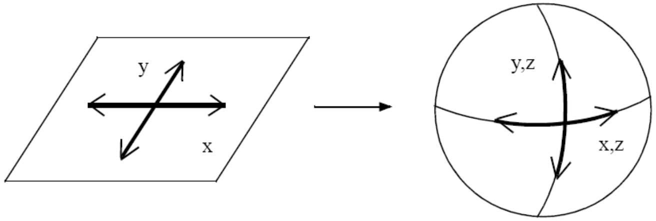
3D-Widgets-Beispiel

Rotation um Achsen



Freie Rotation





TECHNISCHE UNIVERSITÄT DARMSTADT

Vielen Dank für die Aufmerksamkeit