# Grundlagen der Spieleprogrammierung

## Teil I: 3D-Graphik

## Kapitel 2: Die Mathematik

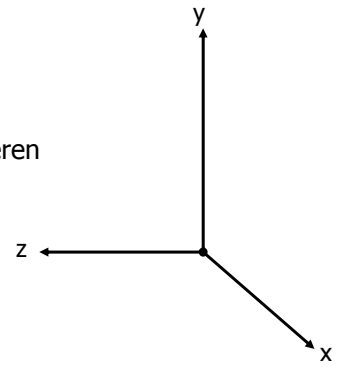
Peter Sturm
Universität Trier

## Outline

1. Übersicht und Motivation
2. Mathematische Grundlagen
3. Das Ideal: Photorealistisch (Raytracing, Radiosity)
4. Die Realität: DirectX und OpenGL (Übersicht)
5. Schritt 1: Drahtgitter
6. Schritt 2: Texturen
7. Schritt 4: Licht, Filter, etc.
8. Schritt 5: Fortgeschrittene Techniken (Vertex-, Pixel-Shader, …)
9. 3D-Hardware
10. 3D-Engines im Überblick, Cg von nvidia
11. Spielekonsolen
12. Zusammenfassung und Ausblick

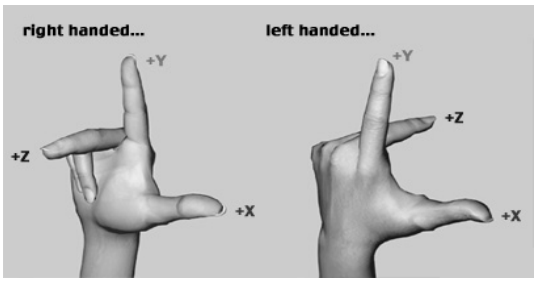
## Motivation

- 3D-Graphik $\Rightarrow$ Mathematik im 3D-Raum
- Abstraktionen
  - Koordinatensysteme
  - Punkt (Vektor)
  - Gerade, Fläche, Körper
  - Affine Transformationen
    - Verschieben, Skalieren, Drehen, Scheren
- Leistungsaspekte
  - Gleitkommazahlen
  - Viele viele Gleitkommaoperationen
  - Hoher Optimierungsdruck




## Koordinationsysteme

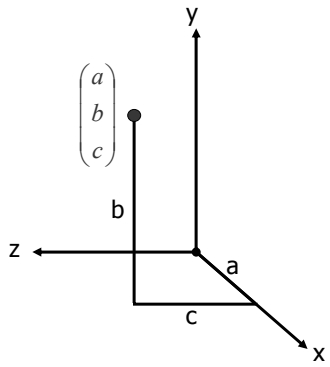
- Bezugssysteme
  - Ursprung und orthogonale Richtungsvektoren
  - Objektposition relativ zum Ursprung
- Absolutes Koordinatensystem
- Relatives Koordinatensystem
  - Objektspezifischer Ursprung
- Linkshändige und rechtshändige Modelle




http://www.tripledtools.com/ramjac/

## Punkte

- Punkte sind relativ zu einem Koordinatensystem
  - Abstand vom Ursprung
- Punkt: 3-Tupel
- Operationen auf Punkte
  - Verschieben (Translation)
  - Skalierung
  - Rotation




## Affine Transformationen

- Translation
  - Verschiebung bzgl. den Grundachsen
  - Mathematisch: Addition
  - V' = V+D
- Skalierung
  - Abstand zum Ursprung kleiner oder größer
  - Mathematisch: Multiplikation
  - V' = S · V
- Rotation
  - Drehung (meist um eine Grundachse)
  - Mathematisch: Winkelfunktionen
  - V' = R · V
  - Beispiel: Drehung im Z-Achse:

## Wünschenswert

- Einheitliche Datenstruktur für alle affinen Transformationen
  - Matrizenoperationen
- Homogenes Koordinatensystem
  - 4-dimensionale Koordinaten
  - Punkt = V(x', y', z', w) mit a' = a/w und w≠0
  - Gängig ist w=1
- Transformationsmatrizen: 4x4

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

## Translation

- Einfache Verschiebungsmatrix

$$\begin{pmatrix} 1 & 0 & 0 & tx \\ 0 & 1 & 0 & ty \\ 0 & 0 & 1 & tz \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Operation

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & tx \\ 0 & 1 & 0 & ty \\ 0 & 0 & 1 & tz \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

## Skalierung

- Skalierungsmatrix

$$\begin{pmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Uniform: sx=sy=sz

- Operation

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

## Rotation

- Nur über jeweils eine Grundachse
  - Alles andere geht auch, ist aber Brain Twister ☺
- Grundmatrizen:

$$Rx = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) & 0 \\ 0 & \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$Rz = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) & 0 & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$Ry = \begin{pmatrix} \cos(\alpha) & 0 & \sin(\alpha) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\alpha) & 0 & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

## Matrizen zusammenfassen

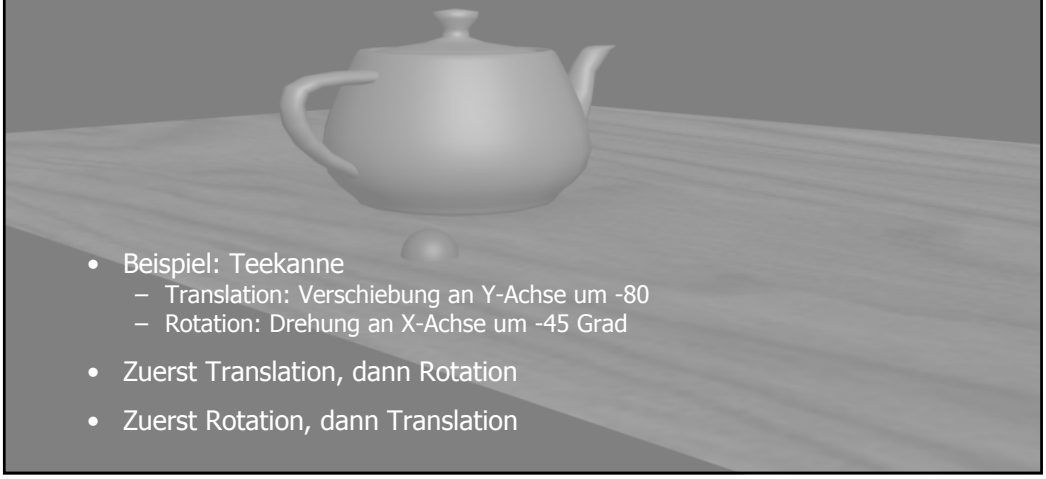
- Rotation und dann Verschiebung
  - Beispiel
    - Drehung um 45 Grad bzgl. X-Achse
    - Verschiebung um -80 bzgl. Y-Achse
- Achtung: Erste Operation = Letzte Matrix
  - M2·M1: Erste Transformation ist M1

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -80 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.707 & -0.707 & 0 \\ 0 & 0.707 & 0.707 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.707 & -0.707 & -80 \\ 0 & 0.707 & 0.707 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

## Konstruktion eines lokalen Systems

- Matrizenoperationen bzgl. Koordinatensystem (Welt)
- Aufbau des lokalen Systems:
  1. Verschieben des "Zentrums" (Pivot) zum Ursprung
  2. Gewünschte affine Transformation ausführen
  3. Verschiebung zurück an den Ausgangspunkt
- Resultat: V2·Taff·V1

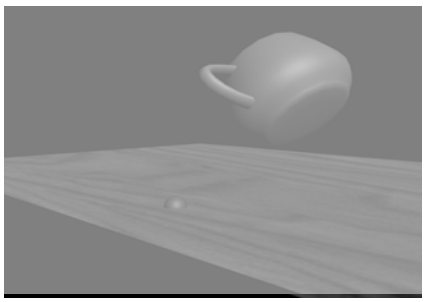
## Matrixoperationen nicht kommutativ

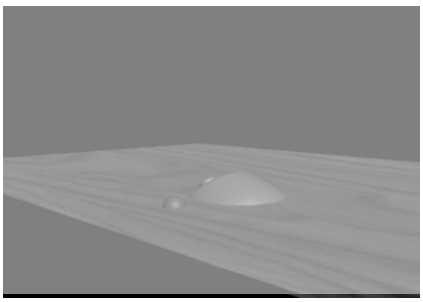


- Beispiel: Teekanne
  - Translation: Verschiebung an Y-Achse um -80
  - Rotation: Drehung an X-Achse um -45 Grad
- Zuerst Translation, dann Rotation
- Zuerst Rotation, dann Translation

## Live

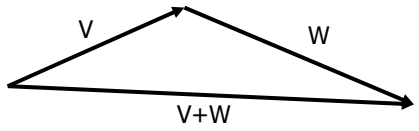
- T, dann R
- R, dann T





## Weitere sinnvolle Funktionen

- Vektoraddition
  - Kräftparallelogramm



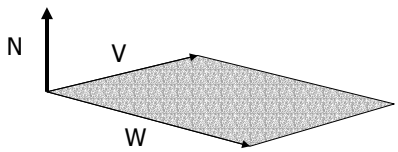

- Länge eines Vektors

$$|V| = \sqrt{x^2 + y^2 + z^2}$$

  - Normalisierter Vektor: $U = \frac{V}{|V|}$

## Normalvektor

- Normale zu einer Oberfläche wird häufig benötigt




- Kreuzprodukt: N = V x W

$$N = \begin{pmatrix} vy \cdot wz - vz \cdot wy \\ vz \cdot wx - vx \cdot wz \\ vx \cdot wy - vy \cdot wx \end{pmatrix}$$

- Bei Kurven muß erst die Tangentialebene bestimmt werden

## Skalarprodukt

- Hilfsmittel zur Winkelbestimmung zwischen Vektoren

$$x = V \cdot W = vx \cdot wx + vy \cdot wy + vz \cdot wz$$

  - Eigenschaft
    - x>0 wenn $\alpha$<90 Grad
    - x=0 wenn $\alpha$=90 Grad
    - x<0 wenn $\alpha$>90 Grad
- Winkelbestimmung zwischen V und W

$$\cos(\alpha) = \frac{V \cdot W}{|V| \cdot |W|}$$

## Implementierungsaspekte

- Hierarchische Definition lokaler Koordinatensysteme
- Ein Weltkoordinatensystem
- Performanz
  - Wann werden kombinierte Transformationsmatrizen berechnet?
  - Speicherbedarf?