

§1 Hardwaregrundlagen

§2 Transformationen und Projektionen

...

2.4 Projektionen

2.5 Perspektivische Projektionen

2.6 Parallele Projektionen

2.7 Umsetzung der Zentralprojektion

2.8 Weitere Projektionen

2.9 Koordinatensysteme, Frts.

2.10 Window to Viewport

2.11 Clipping

§3 Repräsentation und Modellierung von Objekten

§4 Rasterung

§5 Visibilität und Verdeckung

§6 Rendering

§7 Abbildungsverfahren (Texturen, etc.)

§8 Freiformmodellierung

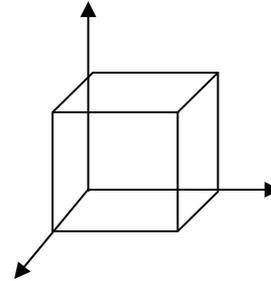
Anhang: Graphiksprachen und Graphikstandards

Anhang: Einführung in OpenGL

Weitere Themen: Netze, Fraktale, Animation, ...

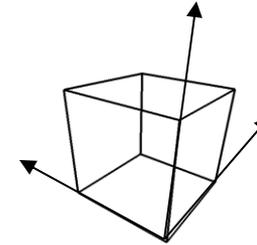
Koordinatensysteme

- Weltkoordinaten (3D) \mathbb{R}^3



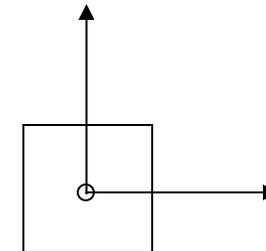
Affine Transformationen

- Beobachterkoordinaten (3D) \mathbb{R}^3



Projektionen

- Normalisierte Koordinaten (3D) $[-1,1] \times [-1,1] \times [-1,1]$

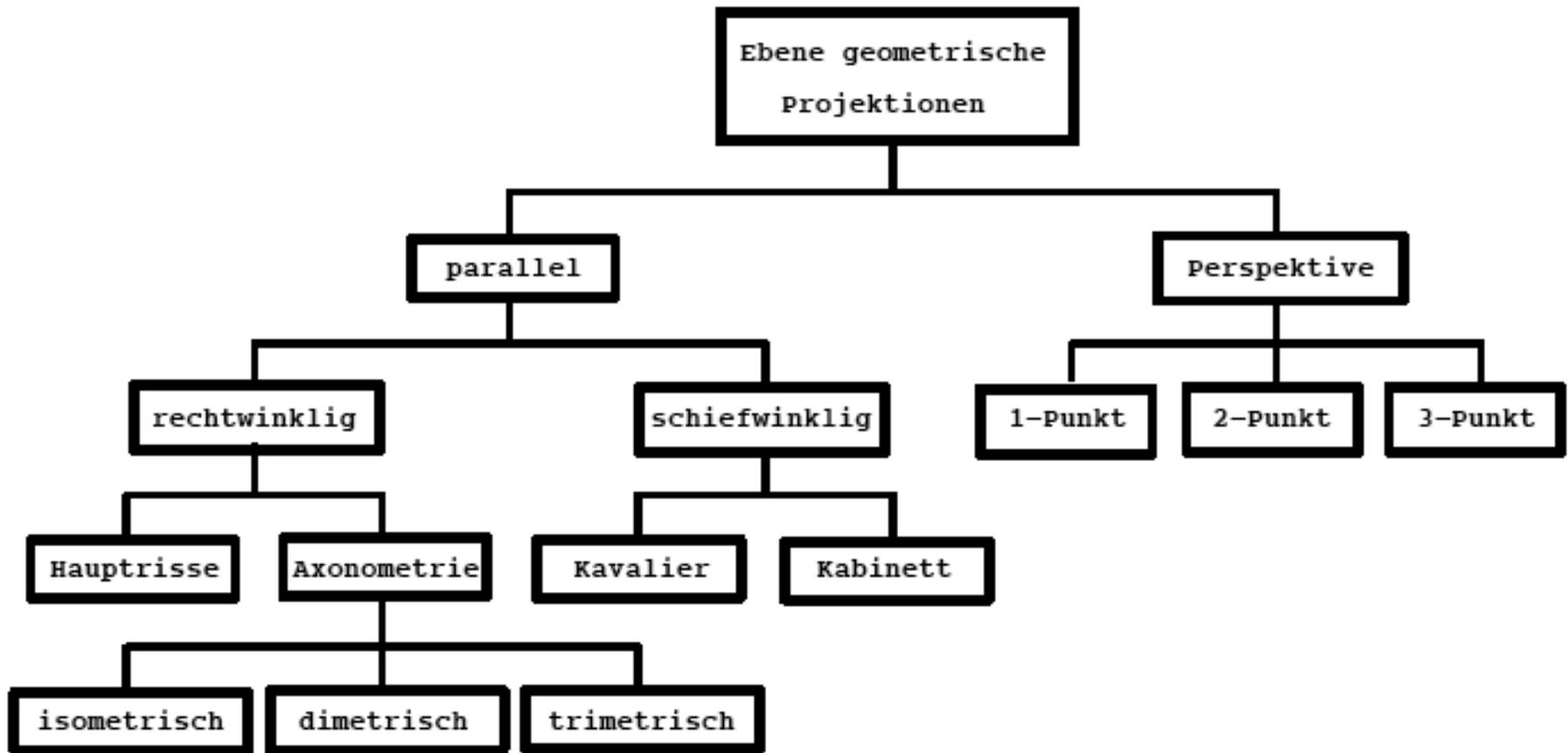


- Abbildung aus einem Raum der **Dimension n** in einen Raum der **Dimension m** , mit **$m < n$** .
- Bildschirm ist zweidimensionales Ausgabemedium:
dreidimensionale Objekte werden in zweidimensionalen Ansichten dargestellt
- Hierzu wird ein **Raumpunkt entlang eines Projektionsstrahls** (Projector) auf eine **vorgegebene Projektionsebene** (Projection Plane) abgebildet.
- Projektionsstrahl durch **Projektionszentrum und Raumpunkt** festgelegt.
- **Schnittpunkt** des Projektionsstrahls mit der Projektionsebene bestimmt **projizierten Raumpunkt**.

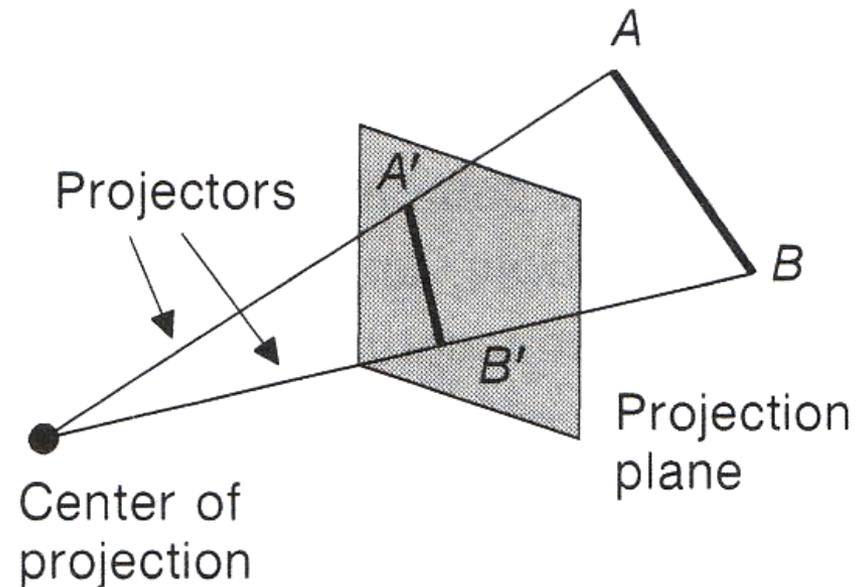
- **Perspektivische Projektion** (Zentralprojektion) und **Parallelprojektion** sind geometrisch planare Projektionen.
- Projektionszentrum der Parallelprojektion liegt in einem **unendlich fernen Punkt**.
- Im Rahmen der projektiven Geometrie stellt die Parallelprojektion somit einen **Spezialfall der Zentralprojektion** dar.

Dies lässt sich z.B. bei der praktischen Umsetzung der Projektionen als **Matrixschreibweisen** gewinnbringend anwenden!

Klassifikation der gängigen Projektionsarten



- Alle Projektionsstrahlen laufen durch das Projektionszentrum
- Projektionszentrum fällt mit dem **Auge** des Beobachters zusammen
- Das Verfahren erzeugt eine **optische Tiefenwirkung**
- Geht in seinen Anfängen bis in die **Malerei der Antike** zurück.



2.5 Perspektivische Projektion

§2 Transformationen und Projektionen



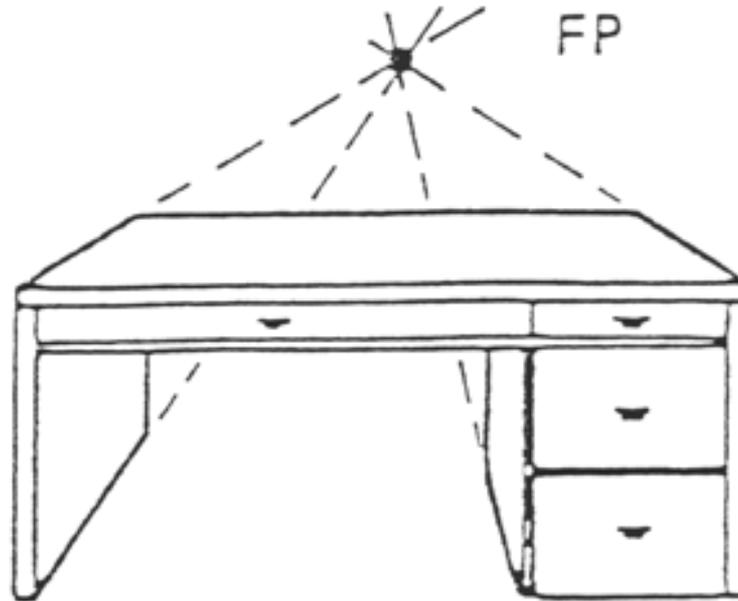
Fluchtpunkt

Raffael: Schule von Athen

Eigenschaften

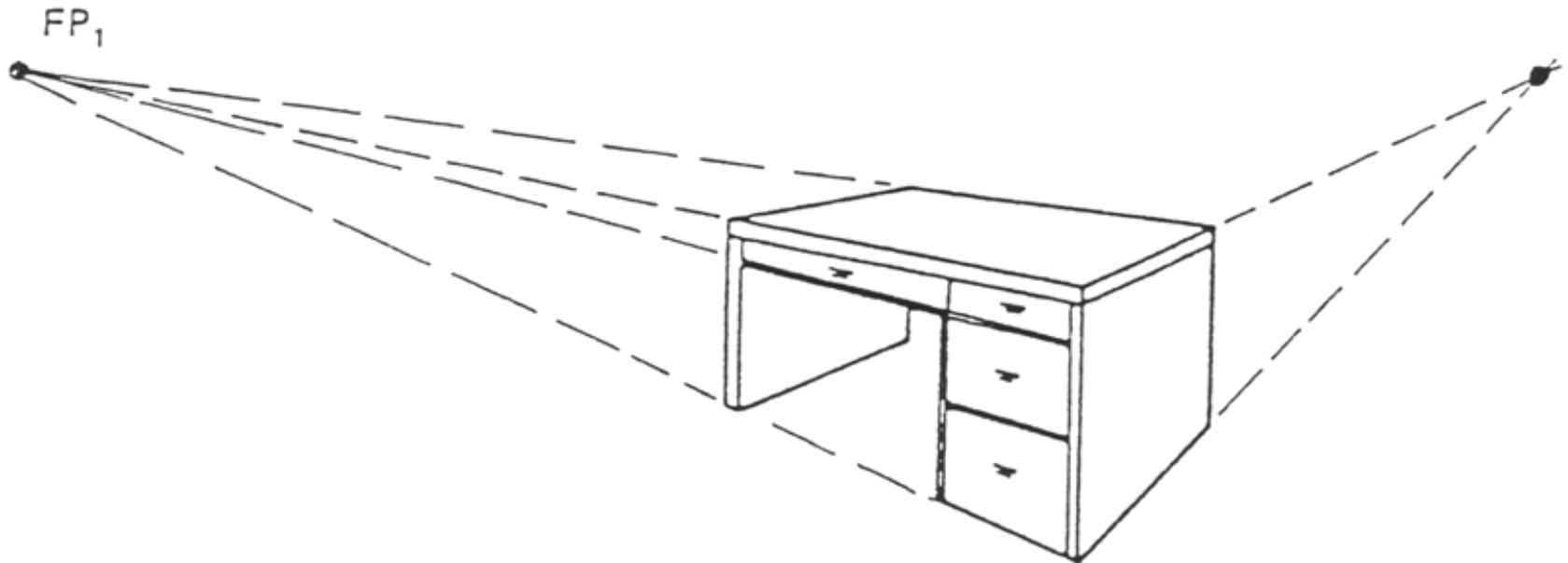
- Je zwei parallele Geraden, die nicht parallel zur Projektionsebene sind, treffen sich in einem Punkt, dem Fluchtpunkt.
- Es gibt **unendlich viele Fluchtpunkte**, je einen pro Richtung nicht parallel zur Projektionsebene.
- Hervorgehoben werden die **Fluchtpunkte der Hauptachsen**
 - Geraden, die parallel zur x-Achse verlaufen, treffen sich im x-Fluchtpunkt (für die anderen Hauptachsen wird dies ähnlich definiert).
- Perspektivische Projektionen werden nach der **Anzahl der Hauptachsen**, die von der Projektionsebene geschnitten werden, klassifiziert
 - 1-Punkt-, 2-Punkt- und 3-Punkt-Perspektiven.

Beispiel



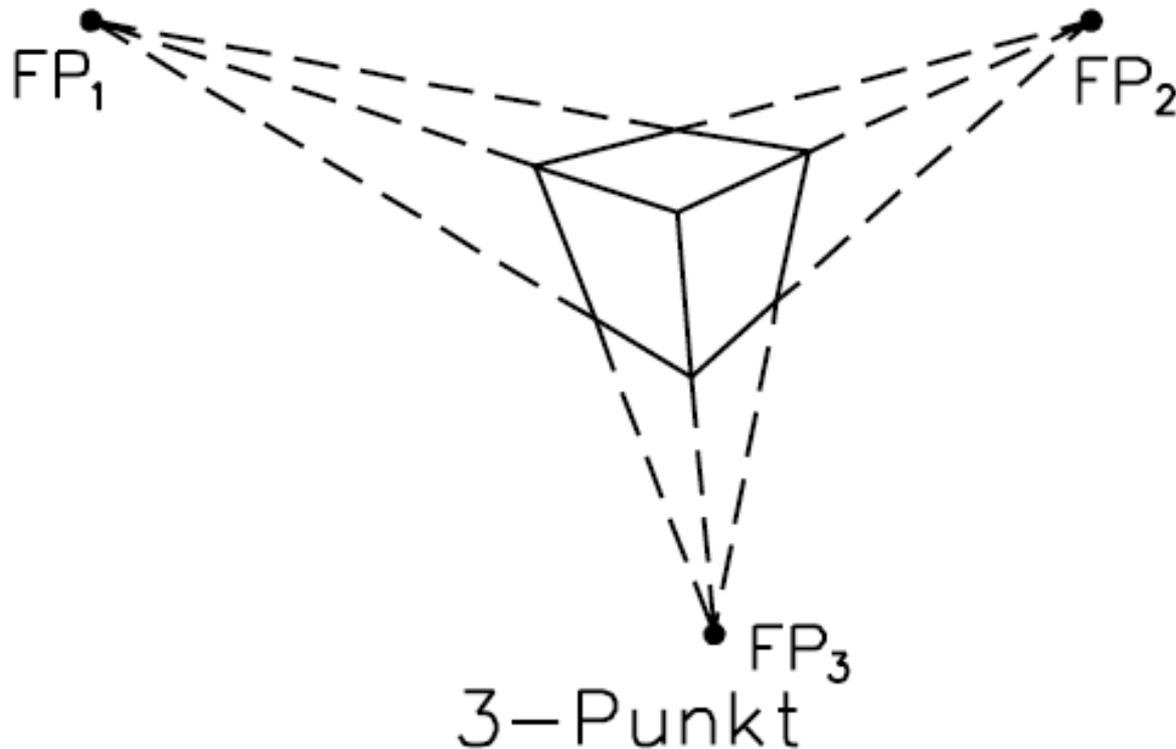
1-Punkt-Perspektive

Beispiel

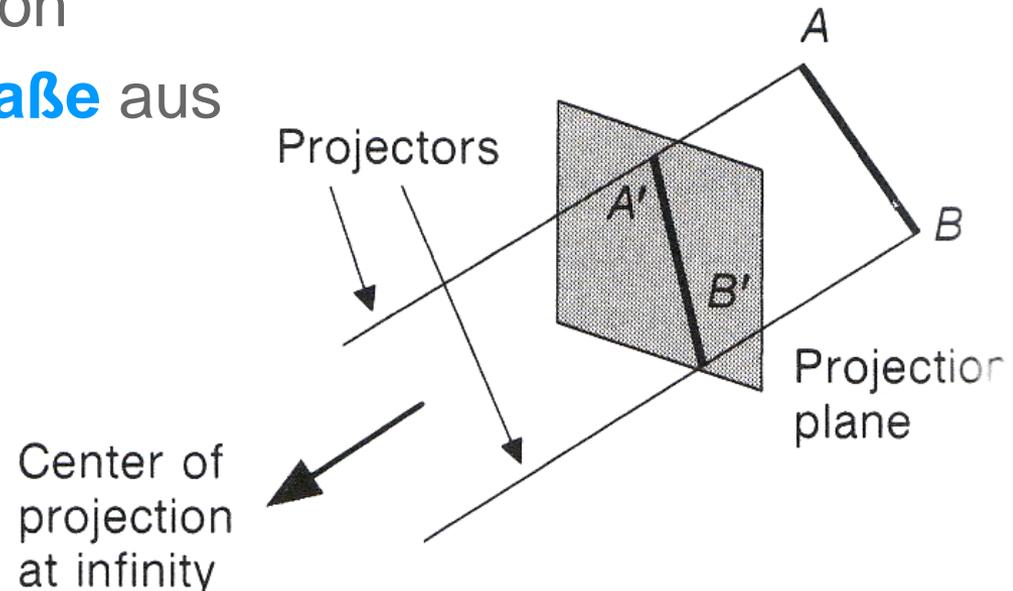


2-Punkt-Perspektive

Beispiel



- Bei der Parallelprojektion ist das **Projektionszentrum im Unendlichen**
- Alle Projektionsstrahlen **verlaufen parallel** in einer Richtung.
- Die Parallelprojektion ist **weniger realistisch** als die perspektivische Projektion
- Aber leichter, **exakte Maße** aus dem projizierten Bild zu bestimmen.

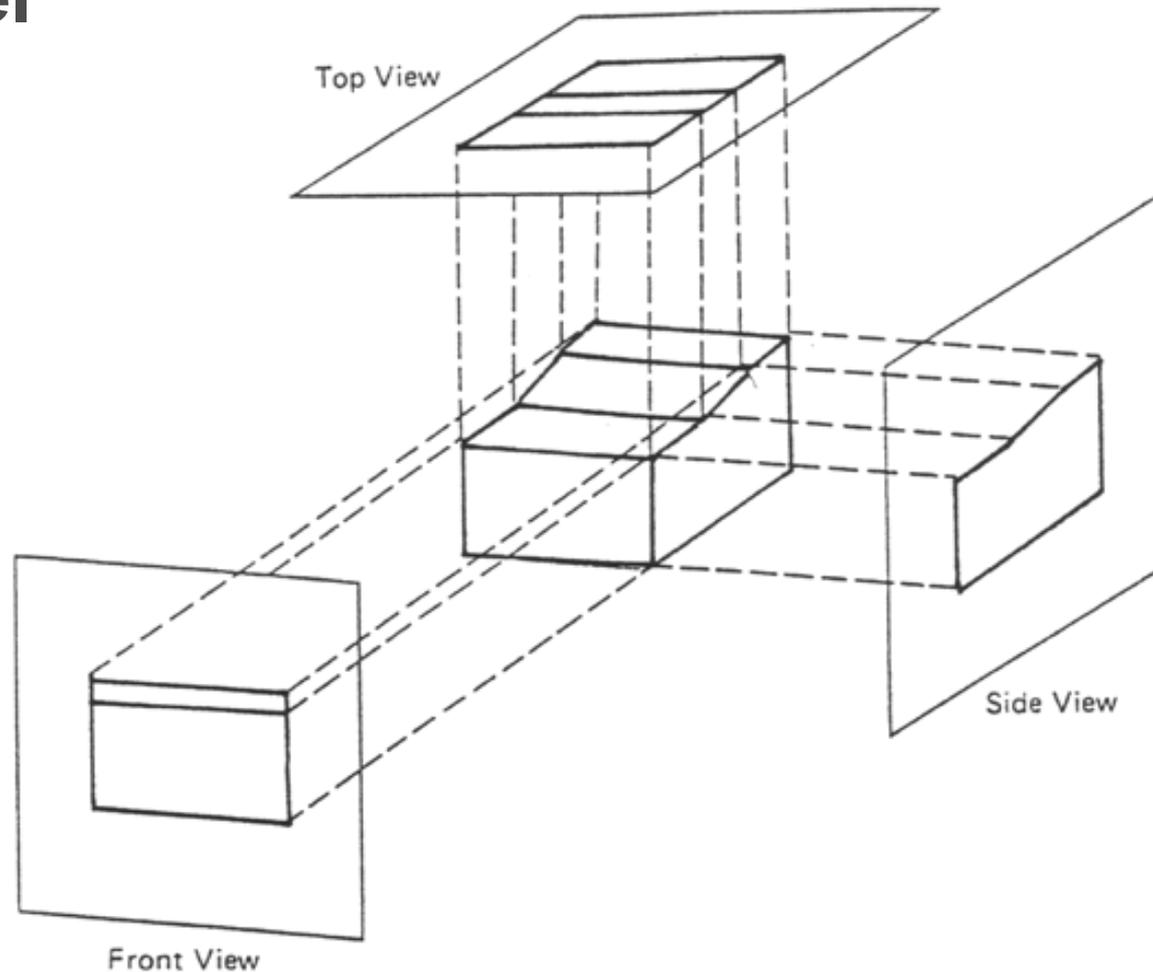


- Die Projektionsstrahlen können bei Parallelprojektionen **gegen die Projektionsebene schief** (\Rightarrow schiefe Projektionen) oder **senkrecht** (\Rightarrow orthographische Projektionen) stehen.

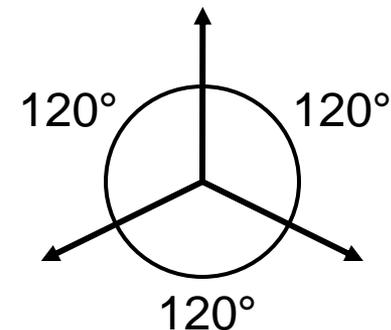
Orthographische Projektion

- Projektionsrichtung fällt mit der **Ebenennormalen** zusammen.
- Man unterscheidet **Haupttrisse und Axonometrie**.
- Bei den so genannten Haupttrissen schneidet die **Projektionsebene nur eine Hauptachse**.
 - Grundriss (Top View), Aufriss (Front View), Kreuzriss (Side View)
 - Die **Normale** der Projektionsebene ist also **parallel** zu einer der **Hauptachsen**.

Beispiel

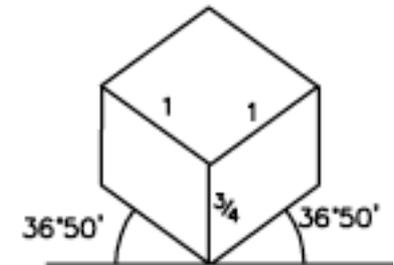
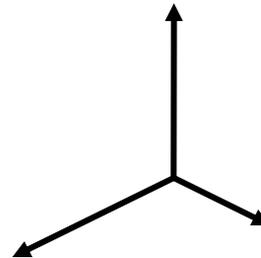


- Bei der **Axonometrie** ist die Projektionsebene nicht orthogonal zu einer der Koordinatenachsen.
- **Parallele Linien** werden auf parallele Linien abgebildet.
- Winkel bleiben allerdings **nicht erhalten**.
- Abstände können **längs der Hauptachsen gemessen** werden, allerdings iA. in jeweils einem anderen Maßstab.
- Im häufigsten Fall der **isometrischen Axonometrie** bildet die Projektionsebene mit allen Hauptachsen den **gleichen Winkel**.
 - Gleichmäßige Verkürzung aller Koordinatenachsen.
 - Es gibt nur **acht mögliche** isometrische Projektionen



Dimetrischen Projektion

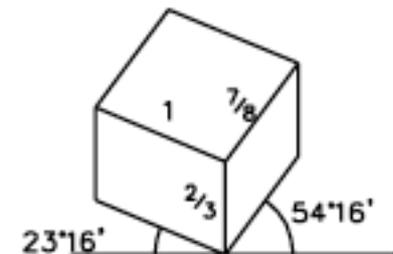
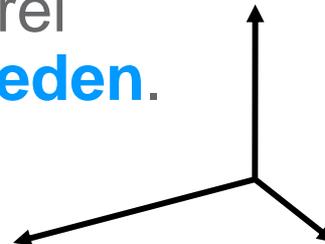
- Projektionsebene mit zwei Hauptachsen den **gleichen Winkel**
- **Skalierung** ist in zwei
- Achsenrichtungen **gleich**



Trimetrischen Projektion

Projektionsebene mit jeder Achse einen **anderen Winkel**

- Skalierungen sind in den drei Achsenrichtungen **verschieden**.

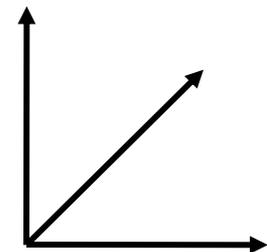


Schiefe Parallelprojektionen

- **Projektionsrichtung unterscheidet** sich von der Projektionsebenennormalen
- Am gebräuchlichsten sind die so genannte **Kavalier-** und die sog. **Kabinettprojektion**.

Kavalierprojektion

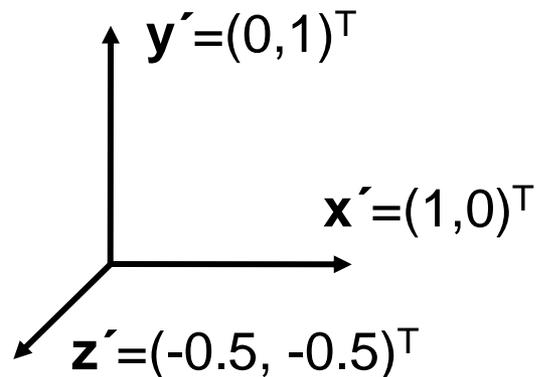
- Der Winkel zwischen Projektionsrichtung und Bildebene **beträgt 45°**
- Hier bleibt die **Länge der Projektion einer Linie**, die senkrecht zur Bildebene steht, unverändert.
- Es gibt **unendlich viele Kavalierprojektionen**, eine für jede Richtung in der Bildebene.



Kabinettprojektion

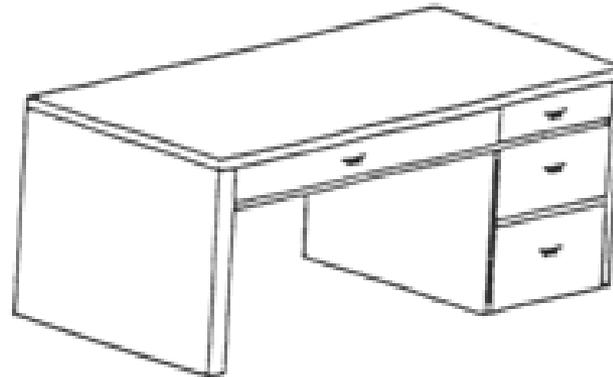
- Länge der Projektion einer zur Projektionsebene (zB. z) senkrechten Linie **soll die Hälfte ihrer Originallänge** werden.
- Der **Winkel zwischen der Projektionsrichtung** und der Bildebene beträgt somit $\arctan(2) = 63.4^\circ$.

Beispiel:

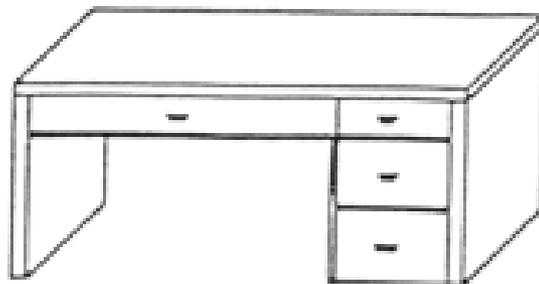


projizierte Einheitsvektoren

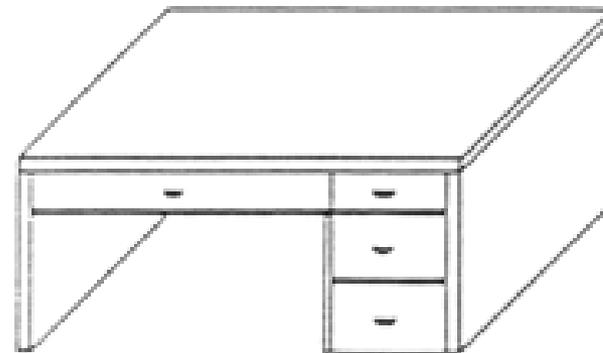
Beispiel



isometrisch: 1:1:1



Kabinettpjektion



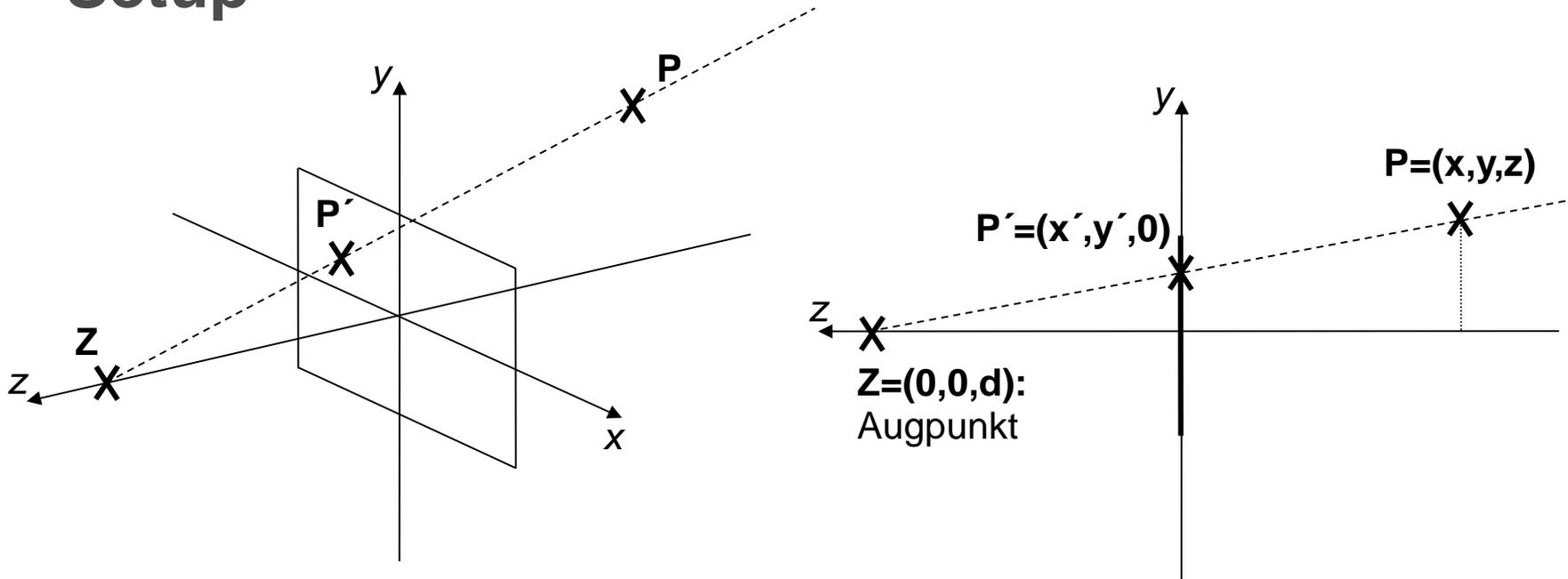
Kavalierprojektion

- Die praktische Umsetzung der perspektivischen Projektion erfolgt je nach Anwendung in **unterschiedlichsten Konfigurationen**
- Diese können mittels **geeigneter Transformationen** des Koordinatensystems erreicht werden.

Exemplarisch wählen wir folgendes Setup:

- Projektionszentrum Z und der Augpunkt **fallen zusammen**
- Beide liegen auf der **positiven z-Achse** mit Abstand $d > 0$ zum Ursprung, also $Z = (0, 0, d)$
- **Blickrichtung** ist die negative z-Achse
- **Bildebene** liegt in der (x,y)-Ebene

Setup



- Aus dem **Strahlensatz** folgt: $\frac{y'}{d} = \frac{y}{d-z}$ und $\frac{x'}{d} = \frac{x}{d-z}$

- Aus dem **Strahlensatz** folgt:

$$\frac{x'}{d} = \frac{x}{d-z} \Rightarrow x' = \frac{x \cdot d}{d-z} \quad \text{und} \quad \frac{y'}{d} = \frac{y}{d-z} \Rightarrow y' = \frac{y \cdot d}{d-z}$$

- Folglich gilt:

$$[x, y, z, 1] \mapsto \begin{bmatrix} \frac{x \cdot d}{d-z} & \frac{y \cdot d}{d-z} & 0 & 1 \end{bmatrix} = [x \cdot d \quad y \cdot d \quad 0 \quad d-z]$$

$$[x, y, z, 1] \mapsto \begin{bmatrix} \frac{x \cdot d}{d-z} & \frac{y \cdot d}{d-z} & 0 & 1 \end{bmatrix} = [x \cdot d \quad y \cdot d \quad 0 \quad d-z]$$

- Die Zentralprojektion wird in diesem Setup somit durch folgende Matrix M beschrieben:

$$[x, y, z, 1] \mapsto [x' \quad y' \quad z' \quad 1] = [x, y, z, 1] \begin{bmatrix} -d & 0 & 0 & 0 \\ 0 & -d & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -d \end{bmatrix}$$

$$= [x, y, z, 1] \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{d} \\ 0 & 0 & 0 & 1 \end{bmatrix} = [x, y, z, 1] M$$

Zerlegung der perspektivischen Projektion

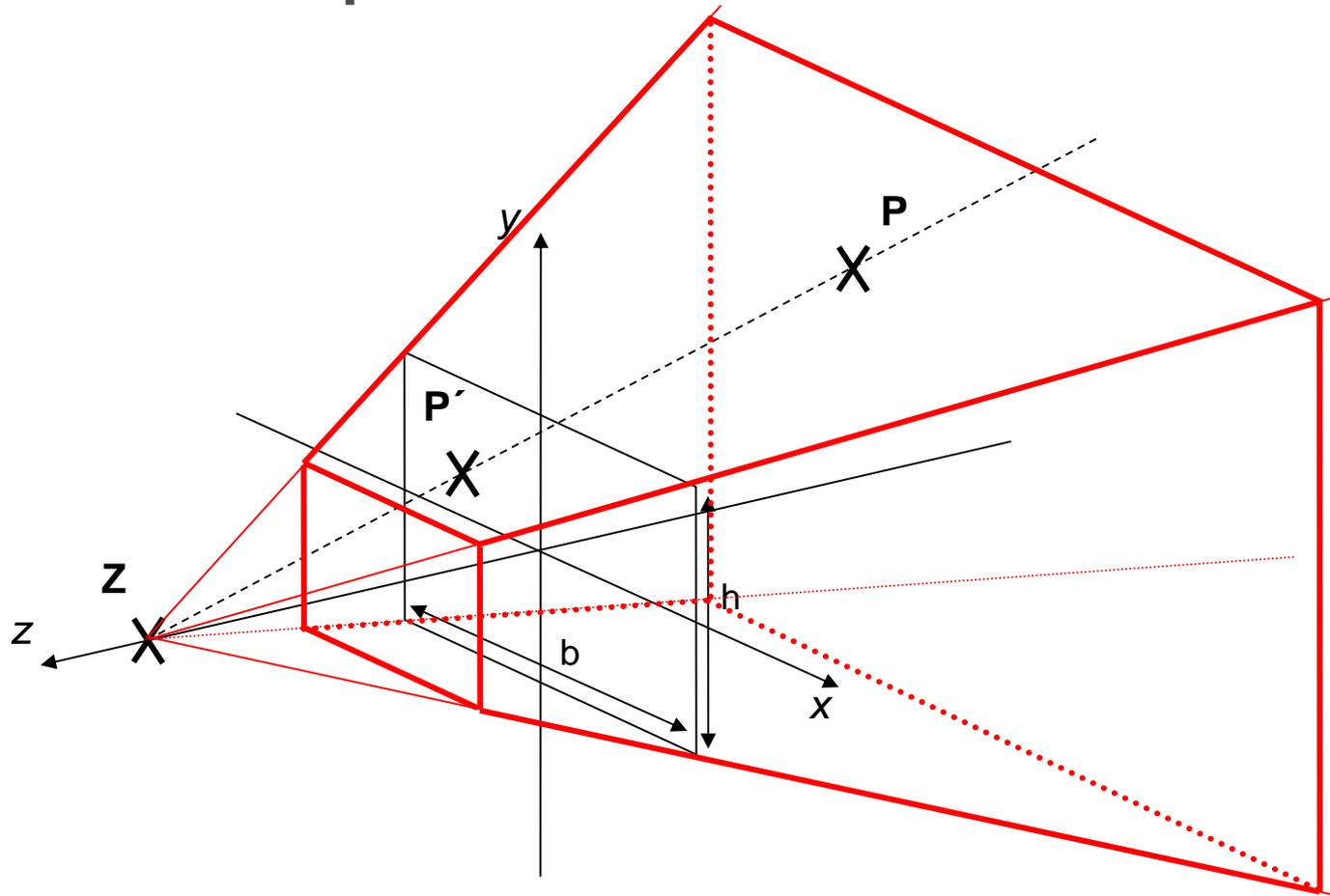
- Perspektivische Transformation M_T ($\mathbb{R}^3 \Rightarrow \mathbb{R}^3$)
- Parallele Projektion M_p auf Ebene $z=0$ ($\mathbb{R}^3 \Rightarrow \mathbb{R}^2$)

$$M = M_T \cdot M_p \Rightarrow \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{d} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -\frac{1}{d} \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

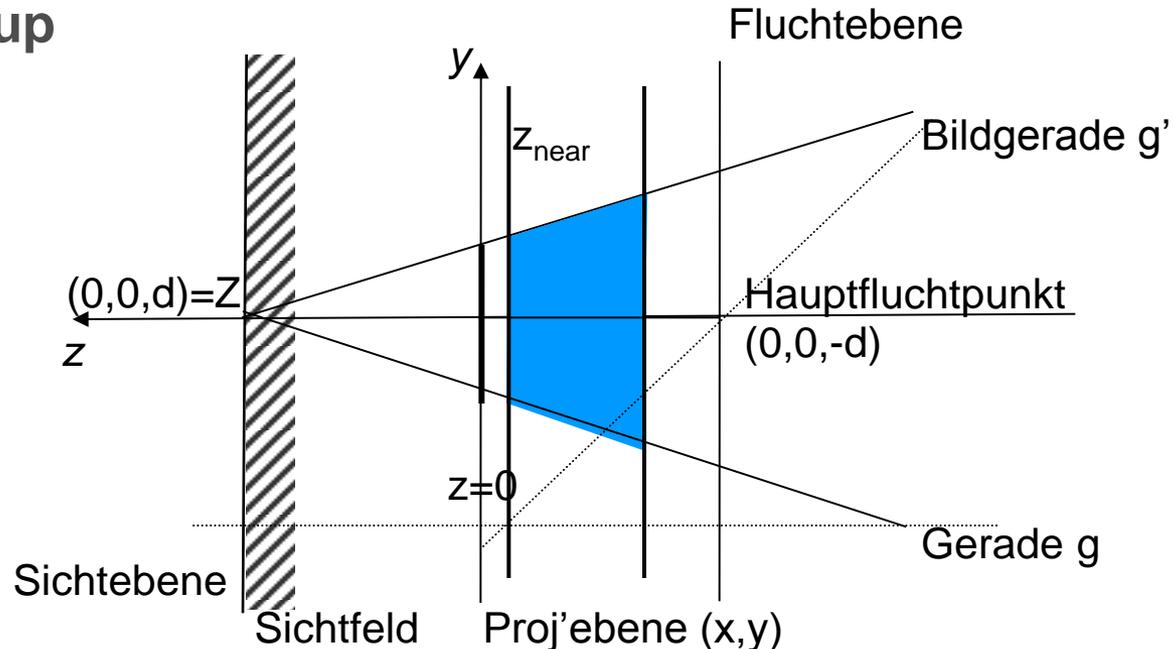
Erweitertes Setup

- In der Bildebene wird ein **Sichtfenster** (View Window) spezifiziert (Breite b , Höhe h , Verhältnis Breite zu Höhe/AR);
 - das Sichtfenster ist **symmetrisch** um den Ursprung angeordnet
- Die Projektoren durch die **Ecken der Bildebene** definieren das so genannte Sichtvolumen (Viewing-Frustum)
- Zusätzlich **begrenzen zwei zur Bildebene parallele Ebenen** (Nahclipebene mit z_{nah} und Fernclipebene mit z_{fern}) das Sichtvolumen in z -Richtung
- Das Sichtvolumen **begrenzt den Teil des Raums**, der dargestellt werden soll (\Rightarrow **Clipping**)

Erweitertes Setup

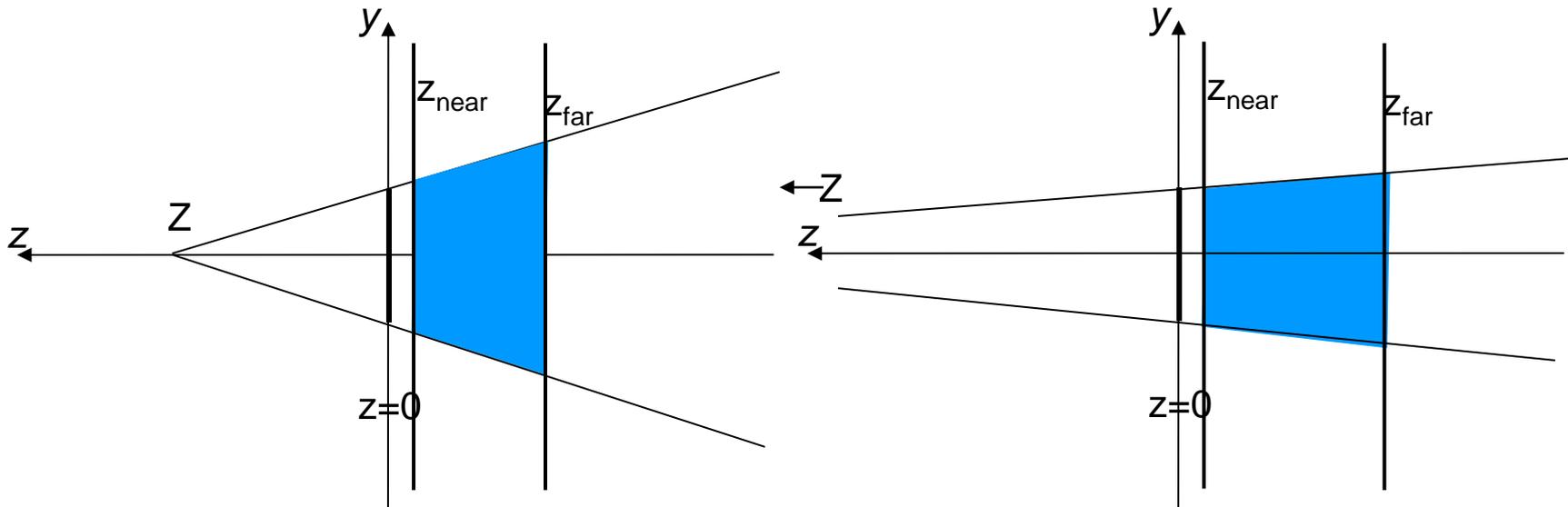


Erweitertes Setup



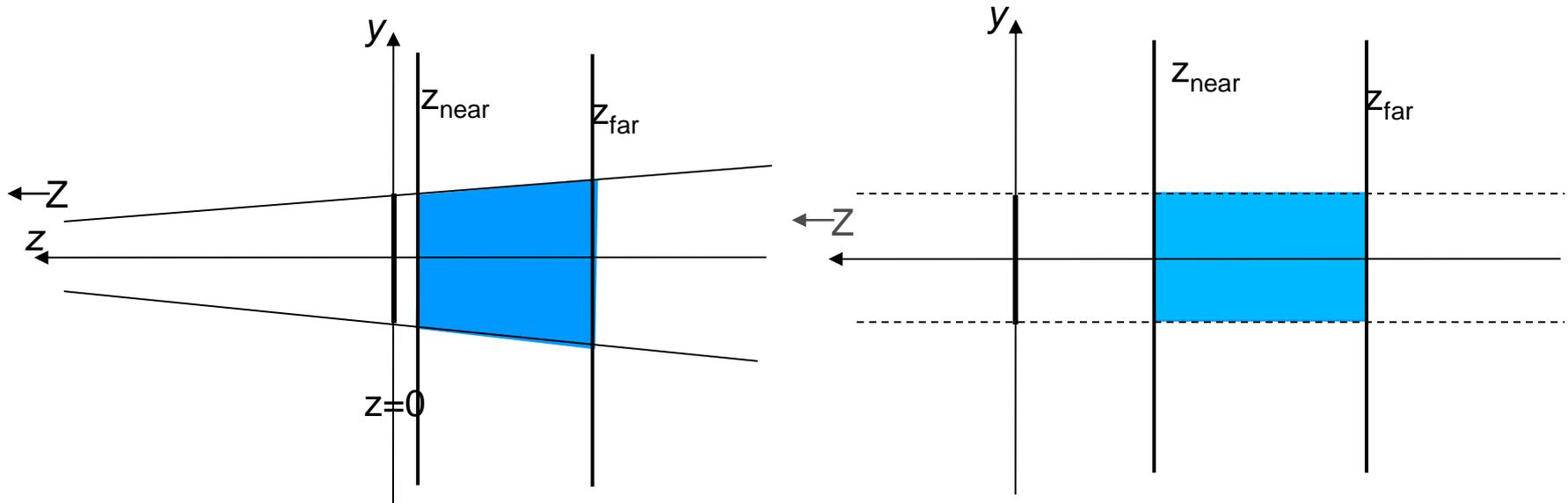
- Sichtebene: Wandert **ins Unendliche** (alle Punkte): $w = 0$
- Sichtfeld ist ein **Halbraum** (ohne Sichtebene)
- Projektionsebene (x,y) ist **Fixebene**
- Alle Punkte $(x,y,z,0)$ werden **auf Fluchtebene** abgebildet
- Geraden g **parallel zur z-Achse** werden auf **Projektionsebene und den Hauptfluchtpunkt** (1-Punkt-Perspektive) zur Bildgeraden g' abgebildet

Perspektivische Transformation M_T des Sichtvolumens



- Affine Gerade durch Z und $[x', y', 0, 1]$ wird auf Gerade parallel zu z -Achse abgebildet
- Sichtfenster in $z=0$ bleibt **gleich groß**
- Translation in den Ursprung (NDC)
 - Kann extra Transformation sein
 - Kann in M_T integriert sein

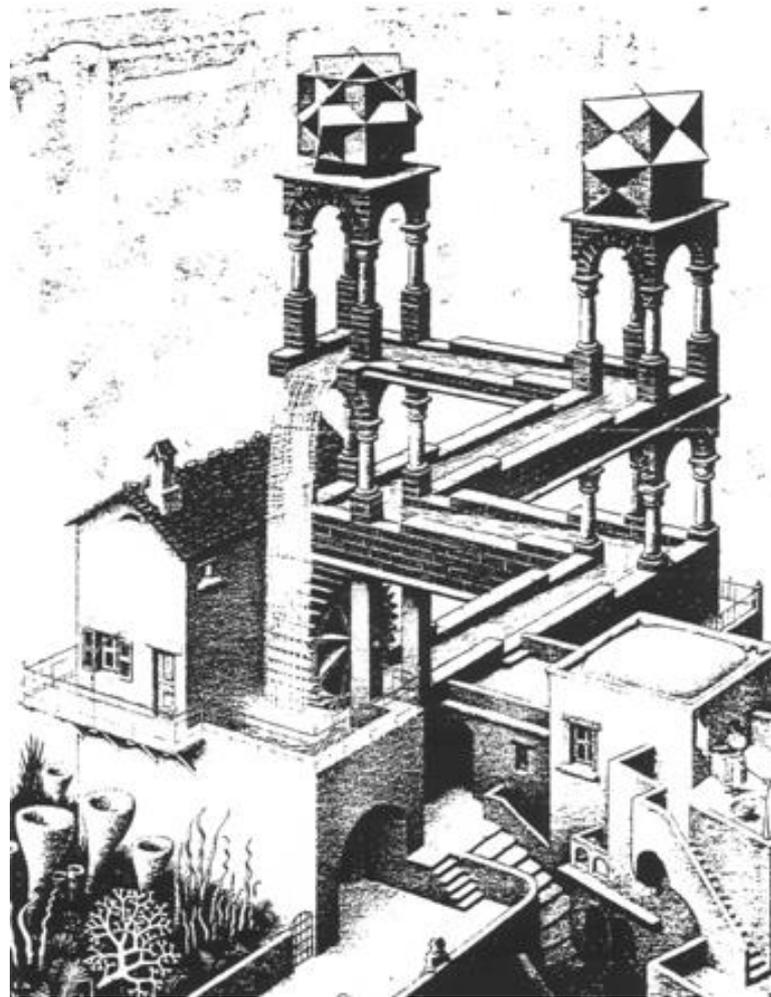
Perspektivische Transformation M_T des Sichtvolumens



- Affine Gerade durch Z und $[x', y', 0, 1]$ wird auf Gerade parallel zu z -Achse abgebildet
- Sichtfenster in $z=0$ bleibt **gleich groß**
- Translation in den Ursprung (NDC)
 - Kann extra Transformation sein
 - Kann in M_T integriert sein

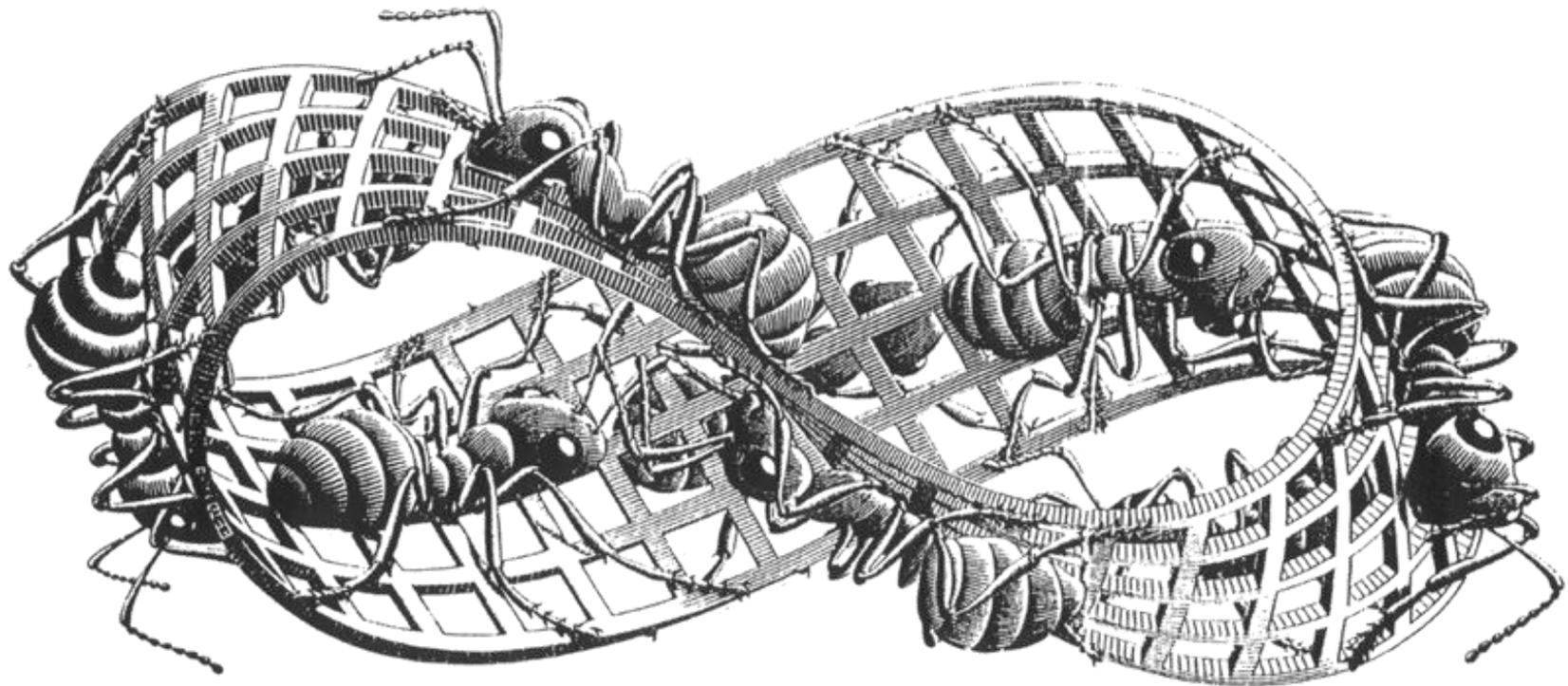


2.8 Projektionen?





2.8 Projektionen?



Motivation

- Objektdefinition, **Szenenmodellierung**, Sichtdefinition, Animation, Rendering, Ausgabe, ...
- ⇒ Einsatz verschiedenster Koordinatensysteme
- ⇒ Hard- und softwaregestützte Umrechnung / Transformation zwischen Koordinatensystemen (idR. bis auf Translationen Basistransformationen im \mathbb{R}^3)

Objektkoordinatensystem OC

(local / object / Modeling Coordinate System)

- **Eigenes** Koordinatensystem für jedes Objekt sinnvoll
- Erleichtert Modellierung und Animation
- Oft über **geometrische Eigenschaften** des Objektes (zB. Symmetrien oder ausgezeichnete Richtungen)

oder durch

- **Erwünschte Aktionen** des Objektes (zB. Flug entlang Kurve mit gleichzeitiger Drehung)

festgelegt

- Auch für Lichtquellen möglich

Weltkoordinatensystem WC (World / Global / Scene Coordinate System)

- Hier „lebt“ die gesamte Szene
- Objekte (Objektkoordinatensysteme) werden mittels Transformationen plaziert
- Zum Rendern: Umrechnung lokaler Objektkoordinaten in globale Objektkoordinaten
(bei Animationen d. h. animiertes Objekt: dynamisch in jedem Frame!)
- Hier ebenso: Beleuchtung der Szene

Beobachterkoordinatensystem BC (Eye / Camera Coordinate System)

- Auch Sichtkoordinatensystem
- Konzept: virtuelle Kamera mit entsprechenden Parametern
- Im Weltsystem (mittels Transformationen) verankert
- So soll es der Beobachter sehen
⇒ hier wird der Bildausschnitt (Sichtfenster) definiert!
- Zum Rendering: idR. Umrechnung globaler Koordinaten **in Sichtkoordinaten** („Szene vor die Kamera drehen“)
(bei Animationen d. h. animierter Kamera: dynamisch in jedem Frame!)
- Virtuelle Kamera bestimmt auch Art der Projektion

Normalisiertes Koordinatensystem NDC (Normalized Device Coordinate System)

- Nach perspektivischer/paralleler Transformation
- Sichtvolumen ist **Einheitswürfel** $[-1, 1] \times [-1, 1] \times [-1, 1]$
(in manchen Darstellungen auch $[0, 1] \times [0, 1] \times [0, 1]$)
- Erleichtert (Vergangenheit) **Hardware**-Operationen
- aber auch viele Softwaretransformationen (**Clipping**)

Bemerkung

- Vorsicht vor Verwirrungen
- Für Transformationen existieren **mehrere äquivalente** Sichtweisen!

„Transformation der Punkte“ - A

- Koordinatensystem bleibt **fest**
- **Punkt ändert Ort** und damit Koordinaten im gleichen System
- Beteiligte Koordinatensysteme: 1

Bemerkung (Fortsetzung)

„Transformation des Koordinatensystems“ - B

- Koordinatensystem wird der **inversen Transformation** der Punkttransformation unterworfen
⇒ **Koordinatensystem bewegt** sich!
- **Punkt bleibt fest**, ändert allerdings Koordinaten, da in neuem System beschrieben
- Beteiligte Koordinatensysteme: eigentlich nur 1 (aber: vorher, nachher)
⇒ „Transformation zwischen Koordinatensystemen“

hervorragend geeignet
für Modellierung und Animation!

Bemerkung (Fortsetzung)

„Transformation mittels Koordinatensystem“ - C

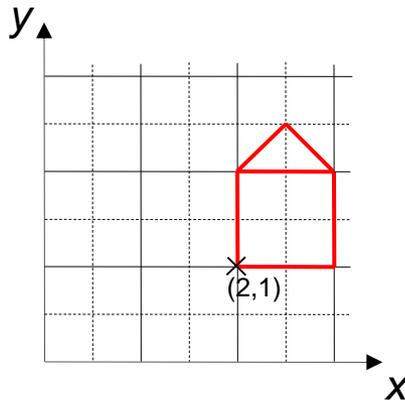
- Ein dem Punkt **lokales Koordinatensystem** wird mittels der Punkttransformation im globalen Koordinatensystem bewegt
⇒ (lokales) **Koordinatensystem bewegt** sich!
- Anschließend wird Punkt (mit „alten“ Koordinaten) in **bewegtes lokales Koordinatensystem eingetragen**
⇒ Punkt ändert Ort
- beteiligte Koordinatensysteme: 2

Verständnisübungen zu 2D-Transformationen

(hintereinander ausgeführte und zusammengesetzte Transformationen)

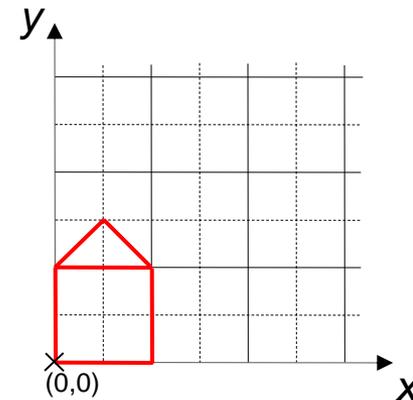
I) Führen Sie die Transformation $R(45^\circ)T(3,1)$ am Haus aus mittels:

- Sichtweise A,
- Matrixrechnung,
- Sichtweise C.



II) Führen Sie die Transformation $R(120^\circ)T(2,2)R(30^\circ)T(2,1)$ am Haus aus mittels:

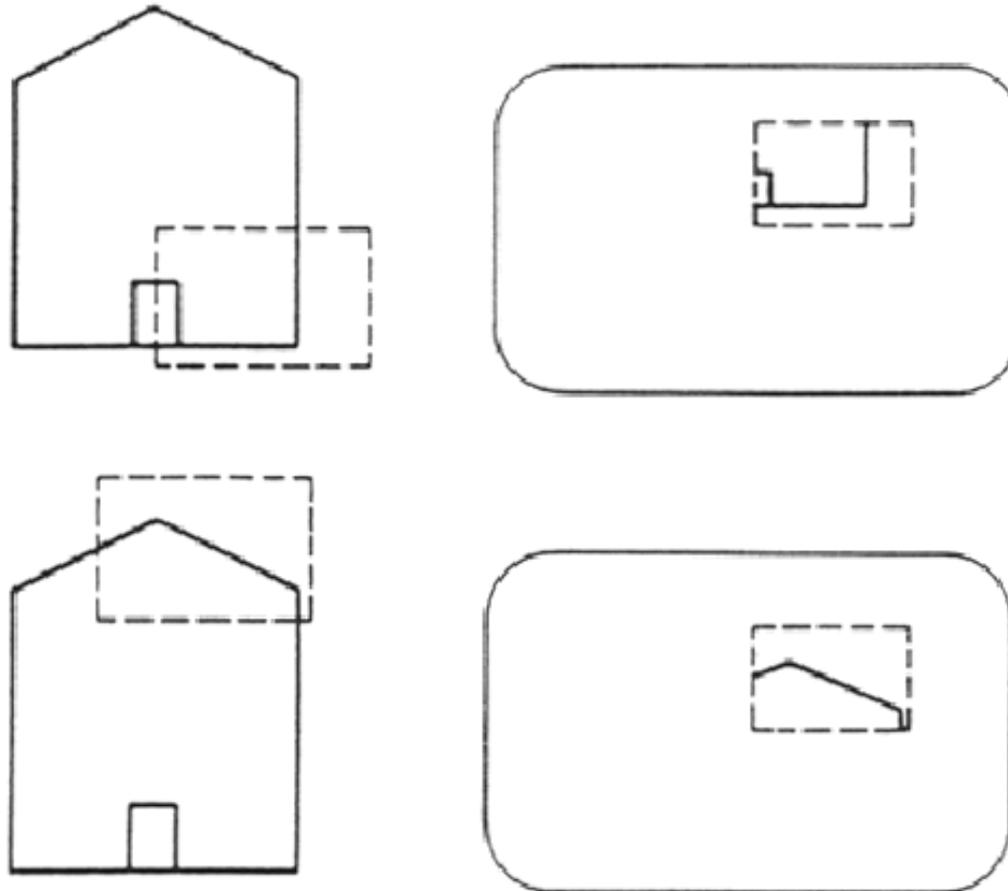
- Sichtweise A,
- Sichtweise C.



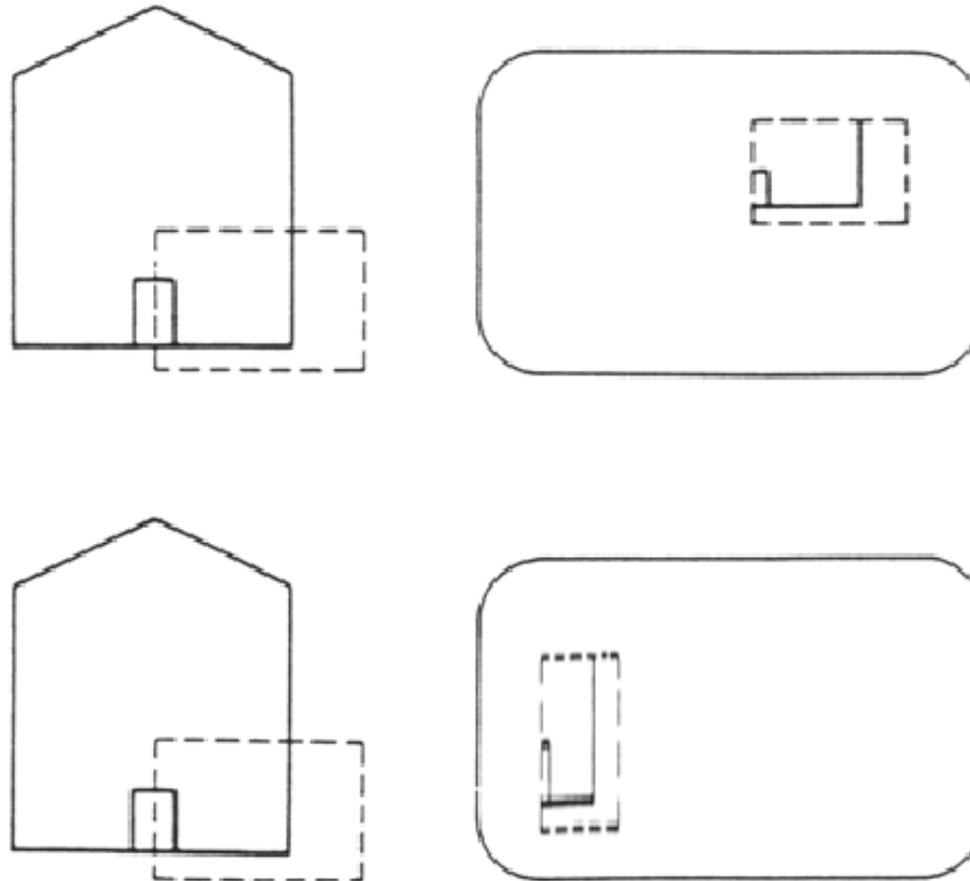
III) Ermitteln Sie allgemein die zusammengesetzten Matrizen für die Transformationen $R(\varphi)T(a,b)$ und $T(a,b)R(\varphi)$.

Begriffe

- Window:
 - Auch „View Window“ (\Rightarrow erweitertes Setup Zentralprojektion)
 - Definiert Sichtfenster in der Bildebene
 - Definiert, welcher Teilbereich der Szene abgebildet werden soll
- Viewport:
 - Definiert Bildschirmbereich in dem der **Inhalt** eines Windows dargestellt werden soll
- Bem.: IdR. sind sowohl Window als auch Viewport an den Koordinatenachsen ausgerichtete rechteckige Gebiete.
- Die **Window-Viewport-Transformation** (Windowing Operation) setzt sich aus **elementaren Translationen und Skalierungen** zusammen.

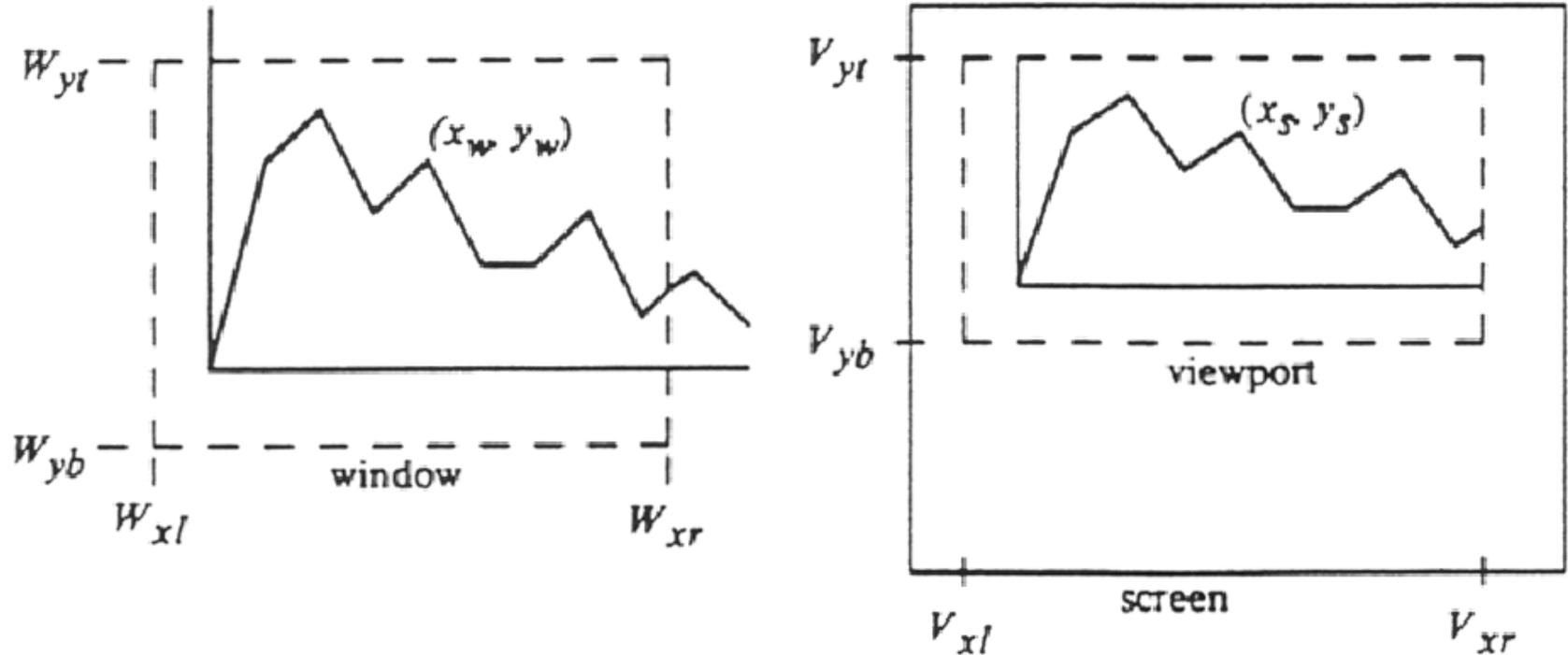


Verschiedene Fenster, dieselben Viewports



Dieselben Fenster, verschiedene Viewports

2.10 Window-Viewport



- x_w, y_w Punktkoordinaten im Window
- x_s, y_s Punktkoordinaten auf dem Bildschirm
- $W_{xl}, W_{xr}, W_{yb}, W_{yt}$ Koordinaten des Windows
- $V_{xl}, V_{xr}, V_{yb}, V_{yt}$ Koordinaten des Viewports im Bildschirmkoordinatensystem

Transformation in 3 Schritten

1) **Translation** in den
Koordinatenursprung

$$x = x'_W - W_{xl}$$

$$y = y'_W - W_{yb}$$

2) **Skalierung** auf
gewünschte Größe

$$x' = \frac{V_{xr} - V_{xl}}{W_{xr} - W_{xl}} x$$

$$y' = \frac{V_{yt} - V_{yb}}{W_{yt} - W_{yb}} y$$

3) **Translation** an gewünschte
Stelle

$$x_S = x' + V_{xl}$$

$$y_S = y' + V_{yb}$$

Transformation in 3 Schritten (Fortsetzung)

Zusammenfassung zu

$$x_S = ax_W + b$$

$$y_S = cy_W + d$$

$$\text{mit } a := \frac{V_{xr} - V_{xl}}{W_{xr} - W_{xl}}, b := V_{xl} - aW_{xl}$$

$$\text{und } c := \frac{V_{yt} - V_{yb}}{W_{yt} - W_{yb}}, d := V_{yb} - cW_{yb}$$

⇒ Punkttransformation durch 2 Multiplikationen und
zwei Additionen