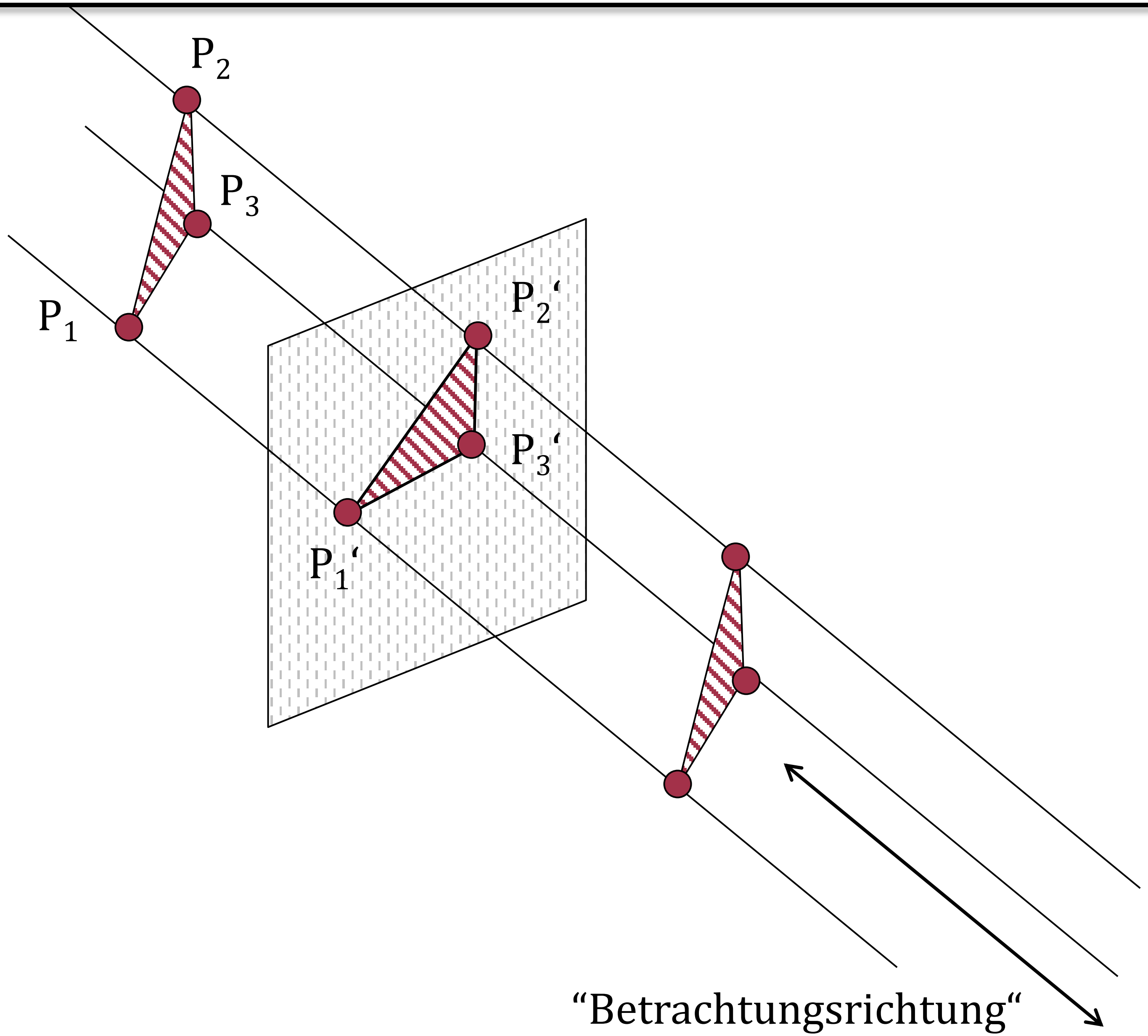
Computergrafik

Universität Osnabrück, Henning Wenke, 2012-05-29

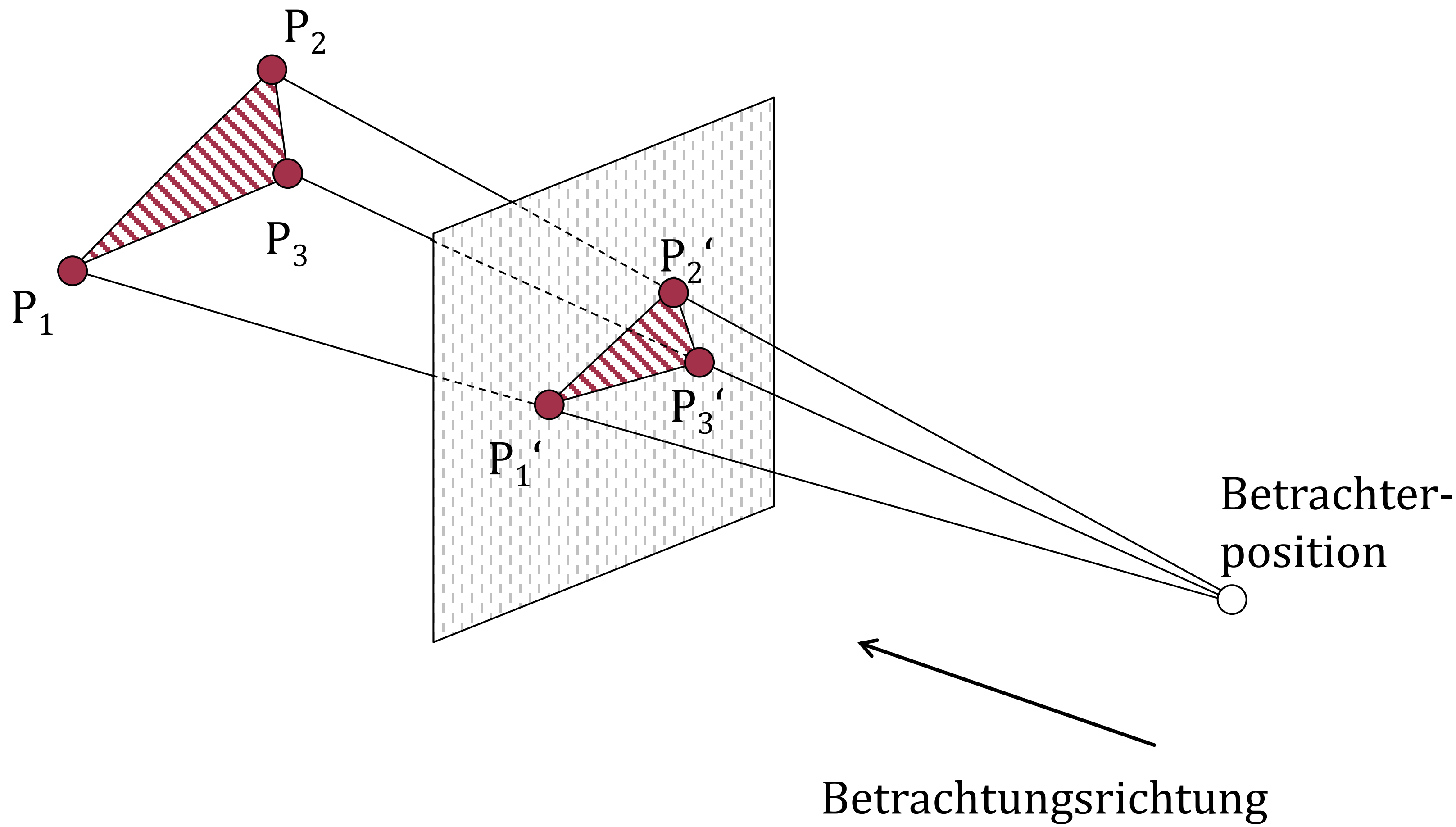
Noch Kapitel VII:

Projection Transformation

Parallele Abbildung auf Bildebene

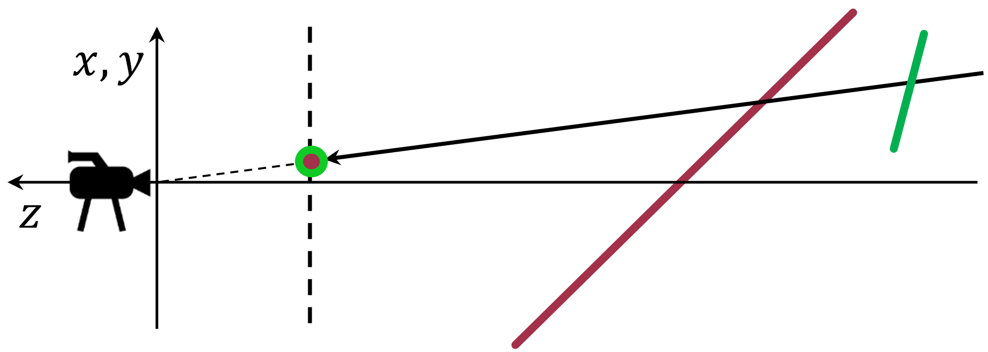


Perspektivische Abbildung auf Bildebene

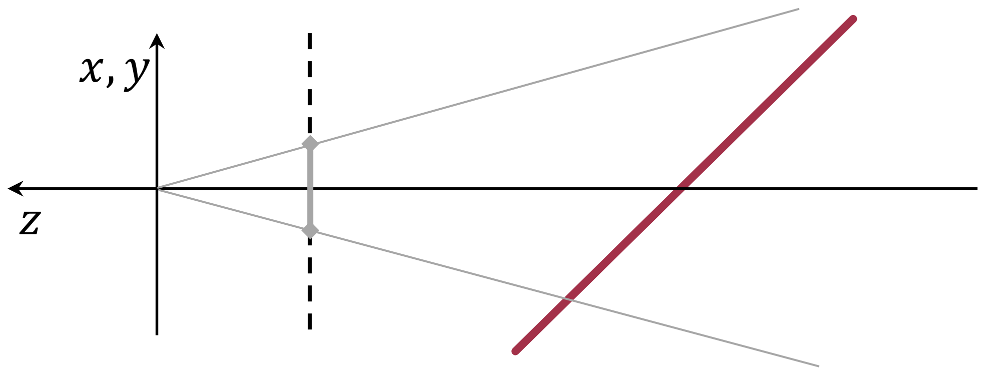


Einschränkungen I

1. Punkte eines Sichtstrahls w. a. identischen Punkt d. Bildebene abgebildet

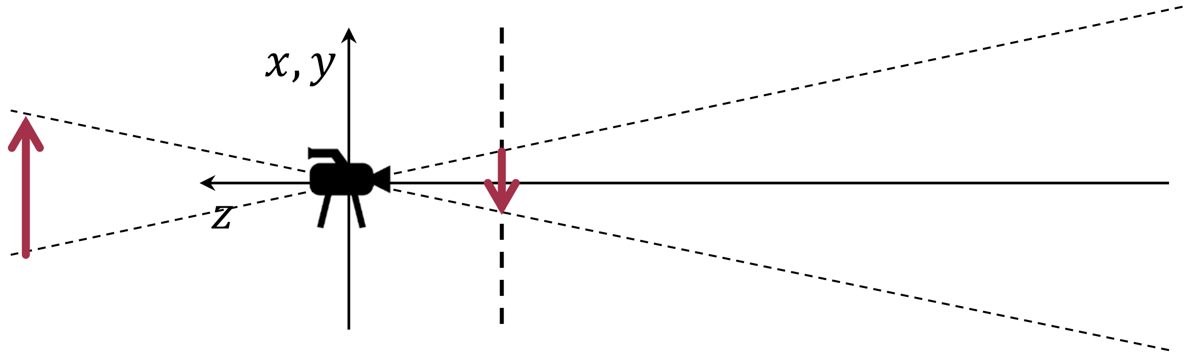


2. Ausgabemedien haben endliche Ausdehnung



Einschränkungen II

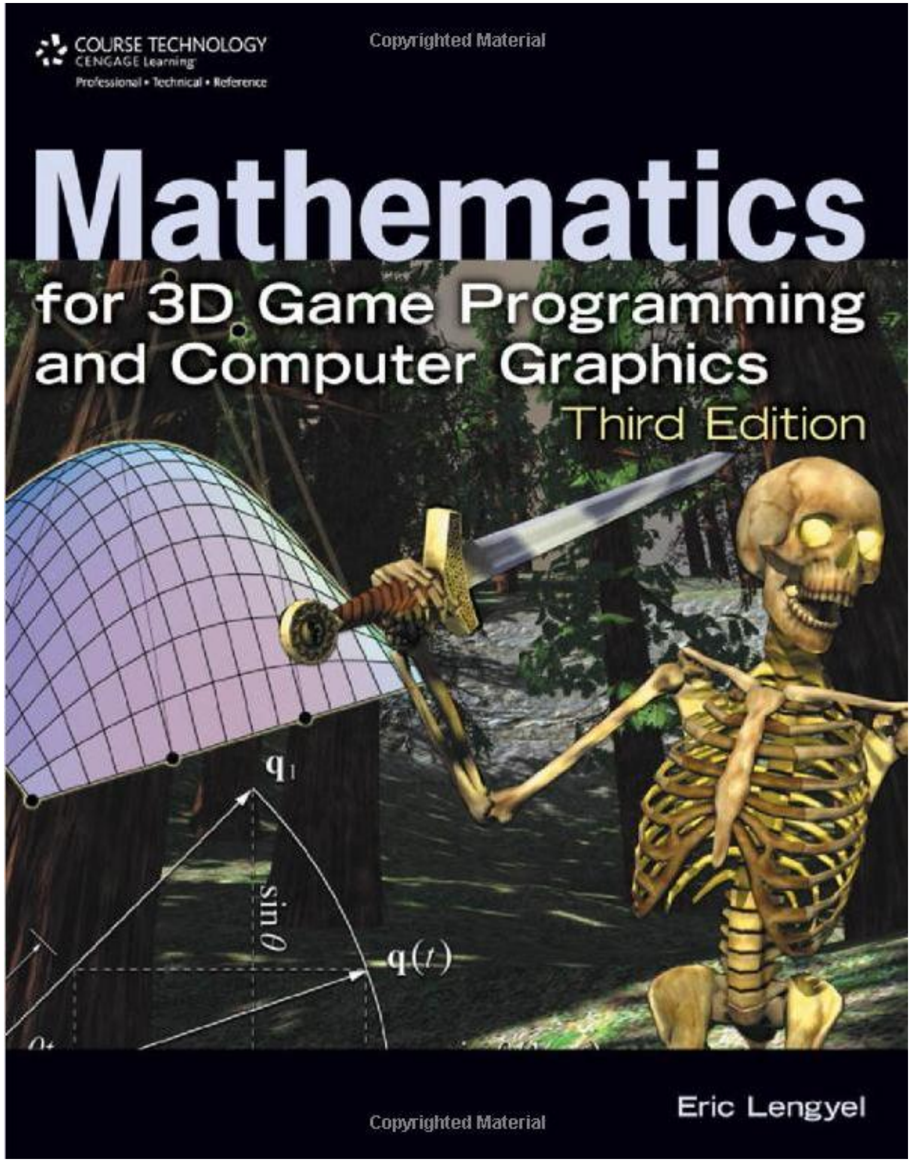
- 3. Bei perspektivischer Projektion: Hinter dem Betrachter sollte nichts sichtbar sein



Anforderungen an Projektion

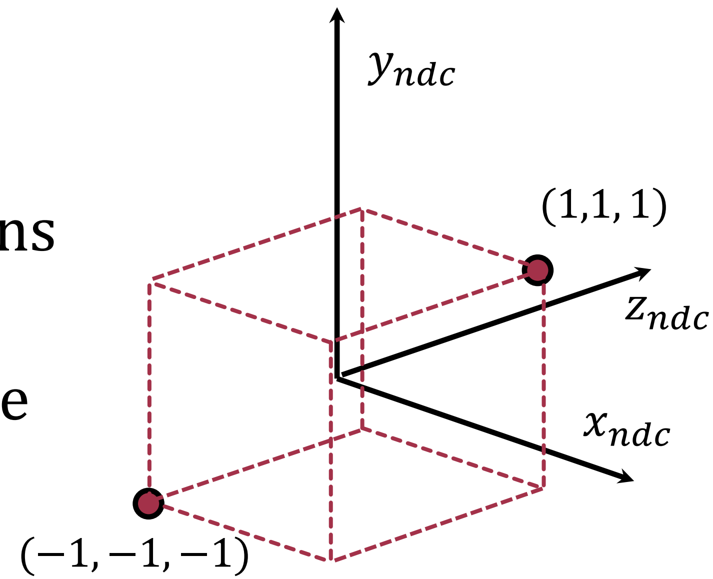
- Berücksichtige jeweilige Charakteristika in x und y
  - Parallelprojektion: $x' = x$ und $y' = y$
  - Perspektivische Projektion: $x' = d \cdot \frac{x}{z}$ und $y' = d \cdot \frac{y}{z}$
- Definition eines sichtbaren Intervalls in x und y
- Erhalten der Tiefeninformation
- Definition des sichtbaren Intervalls in z
  - Nicht immer zwingend
  - Wir grenzen aber immer ein

Zusätzliche Literatur



Canonical View Volume

- Canonical View Volume in OpenGL Würfel mit Kantenlänge 2 um Ursprung
  - $x \in [-1, 1]$
  - $y \in [-1, 1]$
  - $z \in [-1, 1]$
- Nur Objekte innerhalb dieses Volumens sichtbar
- Definiert durch feste Teile der Pipeline
- „Normalized Device Coordinates“
- Linkshändiges KS
- Unsere Aufgabe: Transformation des sichtbaren Teils der Szene in das CVV
- Abbildung auf xy-Ebene: Später

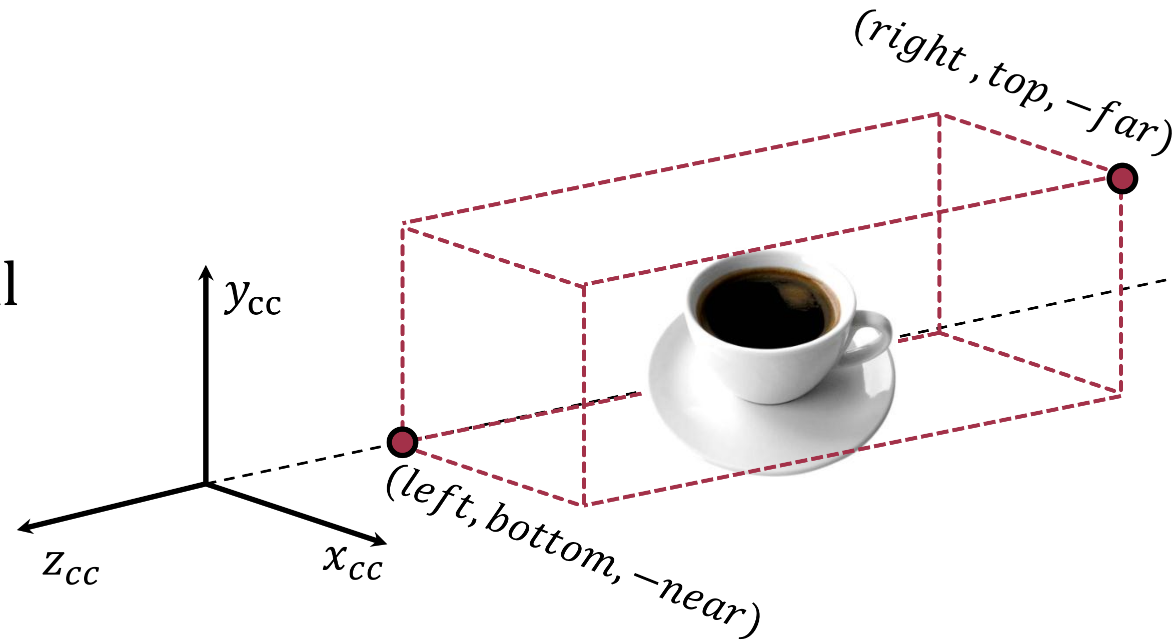


VII.3

Parallel bzw. Orthographic Projection

Projection Normalization

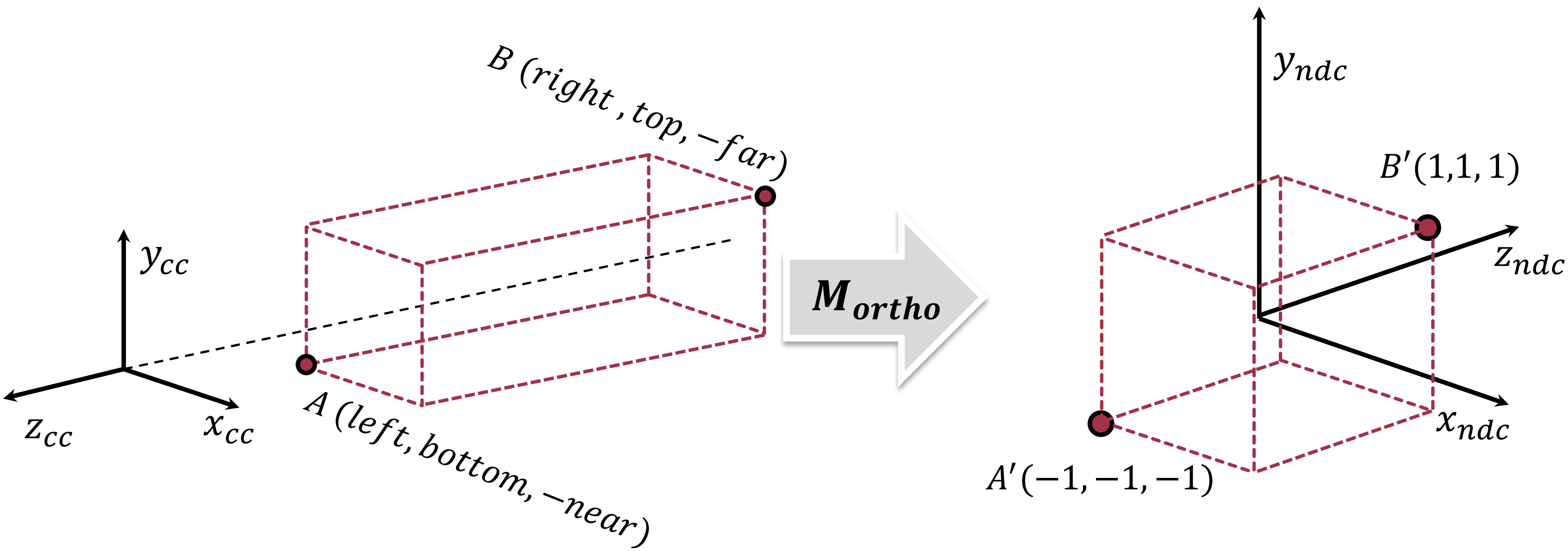
- Definiere sichtbaren Bereich
  - $left \leq x \leq right$
  - $bottom \leq y \leq top$
  - $-far \leq z \leq -near$
  - Kurz: l, r, b, t, n, f
- Überführe in Canonical View Volume
  - Verschiebe in Ursprung $\boldsymbol{T}(-\frac{r+l}{2}, -\frac{t+b}{2}, \frac{n+f}{2})$
  - Normalisiere Volumen $\boldsymbol{S}(\frac{2}{r-l}, \frac{2}{t-b}, -\frac{2}{f-n})$
- Was bei Wahl r, l, t, b zu beachten?
  - $(r-l)/(t-b)$ sollte Seitenverhältnis des Ausgabegeräts entsprechen



Auswirkungen

$$\boldsymbol{M}_{ortho}(\mathrm{r,l,b,t,n,f}) \coloneqq \boldsymbol{S}\left(\frac{2}{r-l}, \frac{2}{t-b}, -\frac{2}{f-n}\right) \cdot \boldsymbol{T}\left(-\frac{r+l}{2}, -\frac{t+b}{2}, \frac{n+f}{2}\right)$$

- Da $n < f$ wird bei Skalierung die z-Achse invertiert
- Aus rechtshändigem wird linkshändiges KS



Matrix

$$\boldsymbol{M}_{\boldsymbol{ortho}}(\mathrm{r,l,b,t,n,f}) \coloneqq \boldsymbol{S}\left(\frac{2}{r-l},\frac{2}{t-b},-\frac{2}{f-n}\right)\cdot \boldsymbol{T}(-\frac{r+l}{2},-\frac{t+b}{2},\frac{n+f}{2})$$

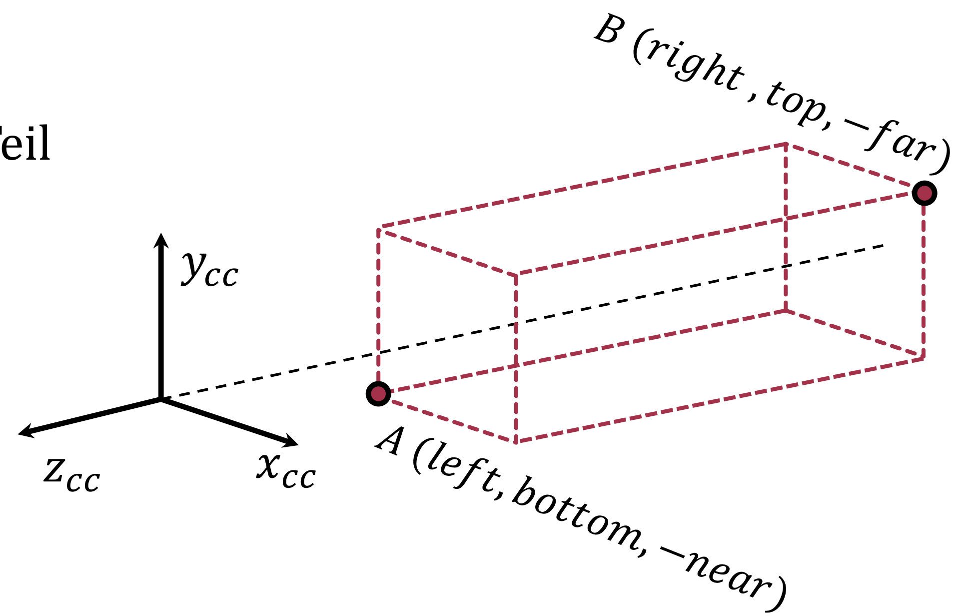
$$\boldsymbol{M}_{\boldsymbol{ortho}}(\mathrm{r,l,b,t,n,f}) \coloneqq \begin{pmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & -\frac{2}{f-n} & -\frac{n+f}{f-n} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Beispiel & Ausblick

- Gegeben:
  - Right, left, top, botton, near, far
  - Vertex mit Koordinate $\boldsymbol{p_{CC}}$ in Camera Coordinates
- Aufgaben
  1. Transformiere in Normalized Device Coordinates
  2. Stelle dann Sichtbarkeit fest
  3. Transformiere in xy-Ebene
- Lösung
  1. Berechne $\boldsymbol{p_{ndc}} = \boldsymbol{M_{ortho}}(r, l, b, t, n, f) \cdot \boldsymbol{p_{cc}}$
  2. Falls $x, y, z \in [-1, 1]$ ist der Vertex potentiell sichtbar
  3. Berechne: $\boldsymbol{P_o} \cdot \boldsymbol{p_{ndc}}$
- Sichtbarkeit vor Transformation?
  - Falls $x \in [l, r],\ y \in [b, t]$ und $z \in [-n, -f]$ gilt

Fragen: Kamera & Parallelprojektion

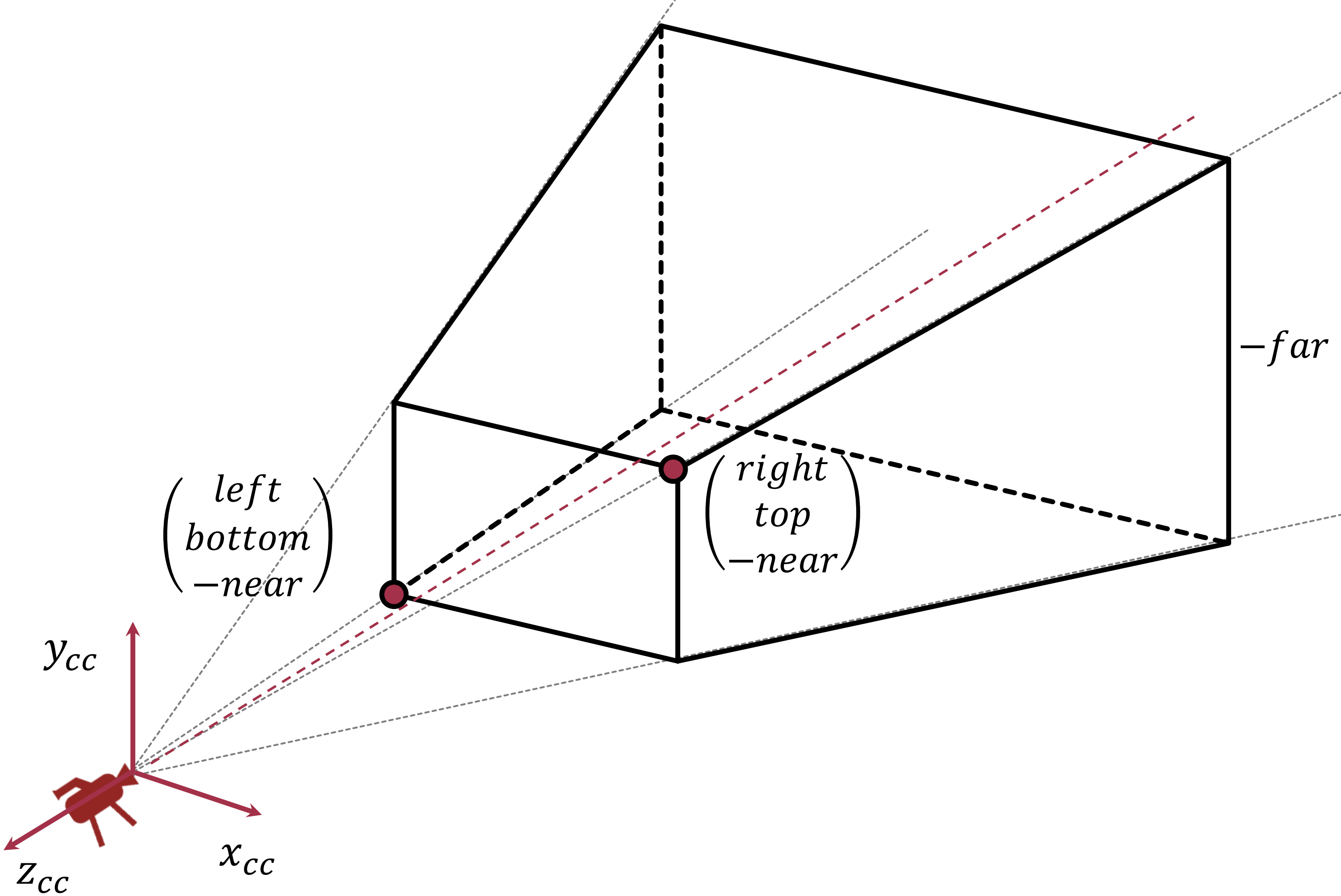
- Gibt es eine Kameraposition?
  - Ja, Ursprung in Camera Coordinates vor der Projektion
  - Danach bedeutungslos
- Auswirkung des Kameraabstands zur Szene?
  - Legt über near und far sichtbaren Teil fest
- Wie Szene verkleinern?
  - right-left bzw. top-bottom größer wählen
- Auswirkungen der Achsen?
  - Betrachtungsrichtung
- Kann etwas hinter der Kamera sichtbar sein?
  - Ja, wenn $near < 0$ gewählt wird



VII.4

Perspective Projection

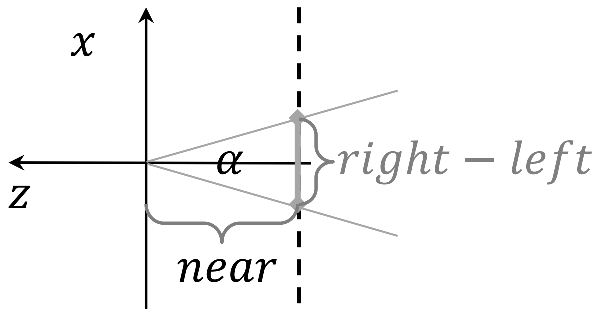
Frustum I

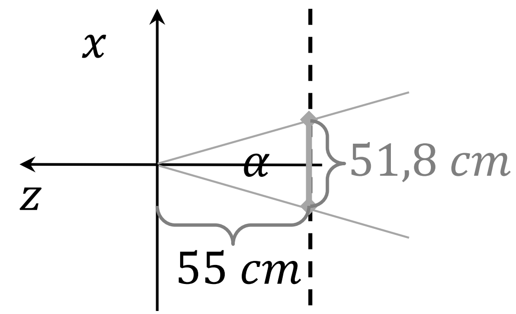


$-far$
$\begin{pmatrix} left \\ bottom \\ -near \end{pmatrix}$
$\begin{pmatrix} right \\ top \\ -near \end{pmatrix}$
y_{cc}
x_{cc}
z_{cc}

Beispiel: F(ield) O(f) V(iew)

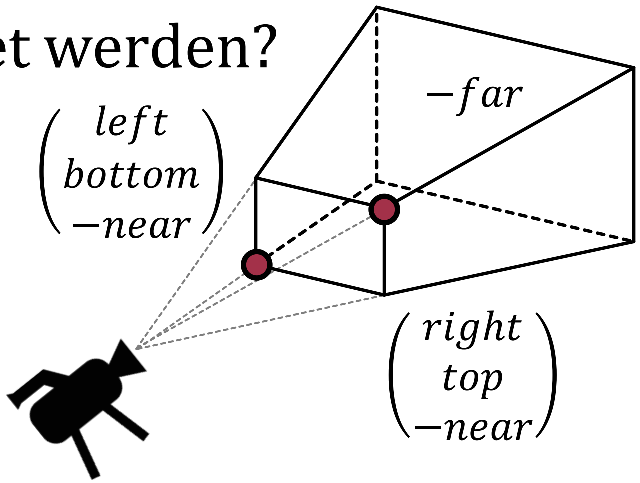
- Gegeben: Monitor (24 Zoll, 16:10, Breite 51,8 cm), Betrachterabstand: 55 cm
- Aufgabe: Stelle FOV für möglichst realistische Perspektive ein
- Breite-Distanz Verhältnis in Realität und Szene muss gleich sein
  - $(right - left)/near \ = \ 51{,}8/55$
  - Setze: $near \ = \ 0.01$ und $left \ = \ -right$
  - $200\ right = 0{,}94 \Rightarrow right \ = -left \ \approx 0{,}005$
- FOV: $\tan\left(\frac{\alpha}{2}\right) = \frac{right}{near} \Longrightarrow \alpha \approx 0{,}927 \approx 53°$
- Berechne Höhe über Seitenverhältnis:
  - $(right - left)/(top - bottom) \ = \ 16/10$
  - Mit $bottom = -top$ folgt: $\frac{0{,}005}{top} = \frac{16}{10} \Longrightarrow top = -bottom = \frac{0{,}05}{16} = 0{,}003125$





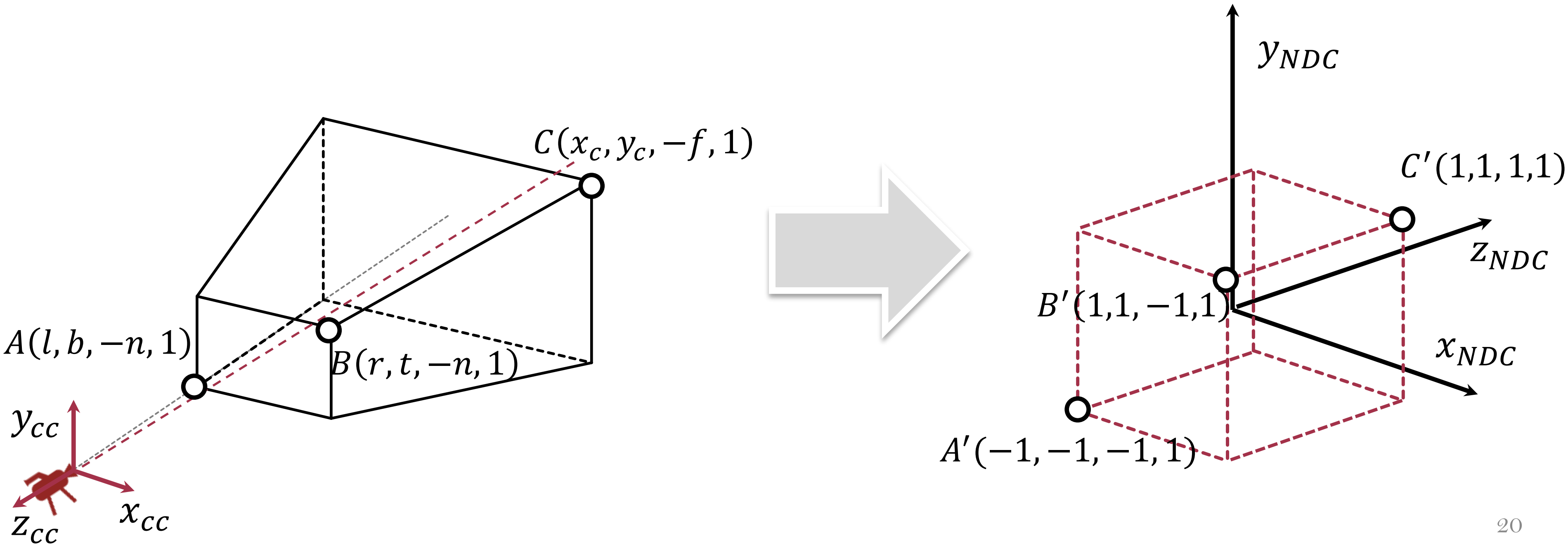
Frustum II

- Definiert sichtbaren Bereich der Szene bei perspektivischer Projektion
- Abgeschnittene Pyramide mit rechteckiger Grundfläche
- Definiert xy-Bildausschnitt auf der Near Plane ($z = -near$)
  - $l \leq x \leq r$ und b $\leq y \leq t$
  - Hinweis: für ($z \neq -near$) sind x, y-Intervalle anders
- Legt auch in z-Richtung Sichtbarkeit fest
  - $-f \leq z \leq -n < 0$
- Kann eindeutige Bildebene eingezeichnet werden?
  - Nein. Ganzes Volumen wird zum Bild



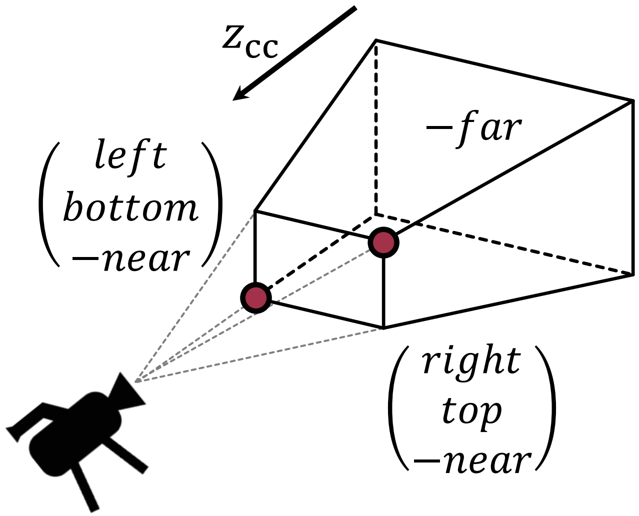
Vorgehen

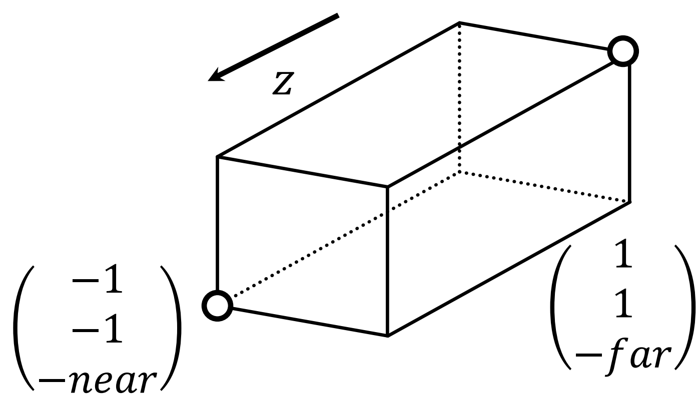
- Transformiere Frustum in CVV
  - Anschaulich: Dabei wird der Raum hinten gestaucht. Dadurch werden entferntere Objekte kleiner
  - z-Achse wird invertiert und aus dem rechtshändigen wird ein linkshändiges KS
- Später: Projiziere mit Parallelprojektion auf Ebene



Perspective Projection I

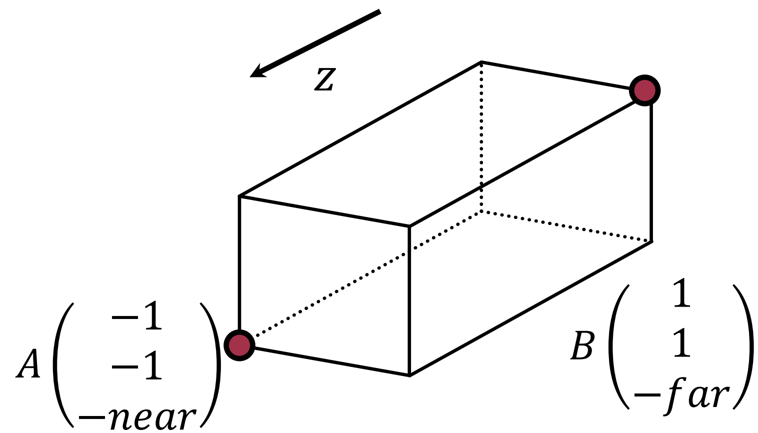
- Frustum definiert sichtbaren Bereich, mit:
  - $l \leq x \leq r$ und b $\leq y \leq t$ auf Near Plane
  - $-f \leq z \leq -n < 0$
- Ziel: Projiziere in CVV
- Bilde x und y der NP auf $[-1,1]$ linear ab:
  - $x'' = (x' - \frac{l+r}{2}) \cdot \frac{2}{r-l} = 2(\frac{x'-l}{r-l}) - 1$
  - $y'' = (y' - \frac{t+b}{2}) \cdot \frac{2}{t-b} = 2(\frac{y'-b}{t-b}) - 1$
- Mit persp. Proj. gilt auf der NP ($d = -near$):
  - $x' = -\frac{nx}{z}$
  - $y' = -\frac{ny}{z}$
- Einsetzen ergibt:
  - $x'' = \frac{2n}{r-l}(-\frac{x}{z}) - \frac{r+l}{r-l}$
  - $y'' = \frac{2n}{t-b}(-\frac{y}{z}) - \frac{t+b}{t-b}$

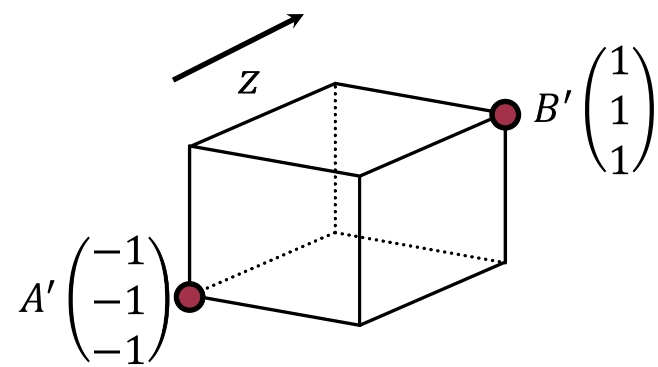




Perspective Projection II

- Bilde $z \in [-n, -f]$ auf $[-1,1]$ ab
  - Verwende: $z' = \frac{a}{z} + b$ $\qquad (I)$
  - Erhält eindeutige Ordnung, aber Richtung umgekehrt
  - Für perspektivische Interpolation nötig
- Transformiere Intervallgrenzen
  - $z = -n \ \rightarrow z' = -1$
  - $z = -f \ \rightarrow z' = 1$
- Einsetzen in (I)
  - $-1 = \frac{a}{-n} + b \ \wedge \ 1 = \frac{a}{-f} + b$
- Lösen des LGS
  - $a = \frac{2nf}{f-n} \ \wedge \ b = \frac{f+n}{f-n}$
- Einsetzen in (I):
  - $z' = -\frac{2nf}{f-n}\left(-\frac{1}{z}\right) + \frac{f+n}{f-n}$





Matrix

- Ergebnis d. Transformationen (nach Perspective Division):
  - $x' = \frac{2n}{r-l}\left(-\frac{x}{z}\right) - \frac{r+l}{r-l}$
  - $y' = \frac{2n}{t-b}\left(-\frac{y}{z}\right) - \frac{t+b}{t-b}$
  - $z' = -\frac{2nf}{f-n}\left(-\frac{1}{z}\right) + \frac{f+n}{f-n}$
  - $w' = 1$
- Multiplikation mit $(-z)$:
  - $-z \cdot\ x' = \frac{2n}{r-l}x + \frac{r+l}{r-l}z$
  - $-z \cdot y' = \frac{2n}{t-b}y + \frac{t+b}{t-b}z$
  - $-z \cdot z' = -\frac{2nf}{f-n} - \frac{f+n}{f-n}z$
  - $-z \cdot w' = -z$

$$\boldsymbol{M}_{\boldsymbol{frustum}} \coloneqq \begin{pmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2nf}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

- $\boldsymbol{M}_{\boldsymbol{frustum}}$ überführt Camera Coordinates in Clipping Coordinates

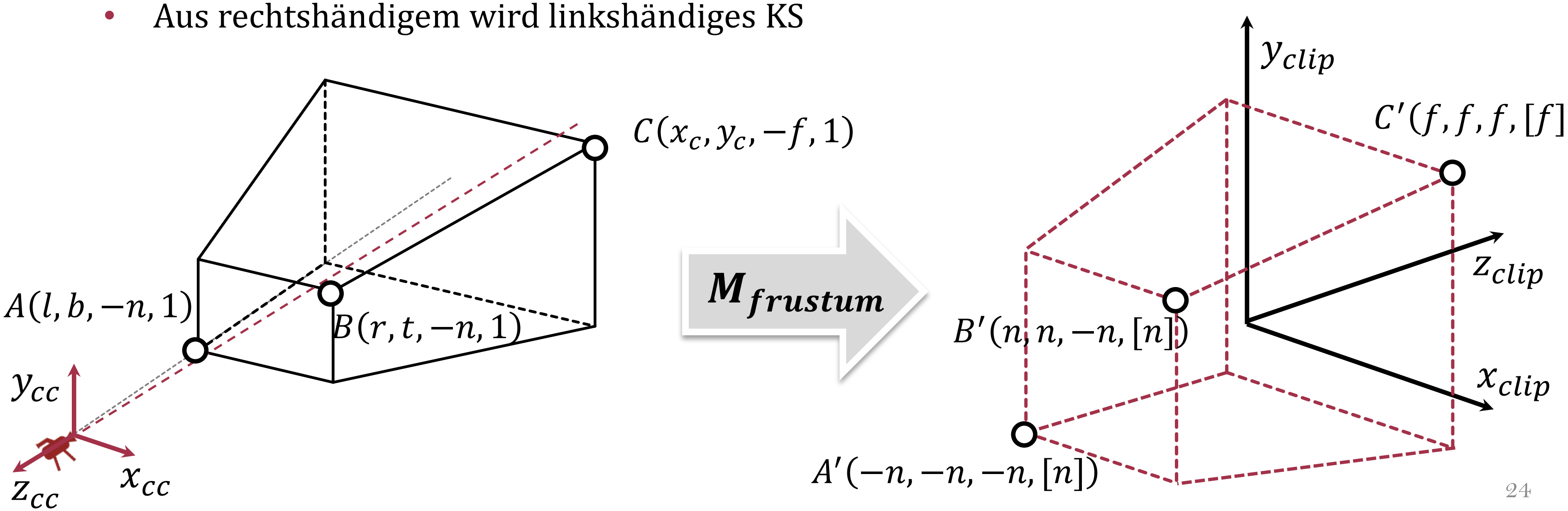
Ergebnis vor Perspective Division

$$x' = \frac{2n}{r-l}x + \frac{r+l}{r-l}z, \qquad y' = \frac{2n}{t-b}y + \frac{t+b}{t-b}z$$

- Nach Multiplikation mit $\boldsymbol{M_{frustum}}$: $z' = -\frac{2nf}{f-n} - \frac{f+n}{f-n}z \qquad -f \to f \quad -n \to -n$

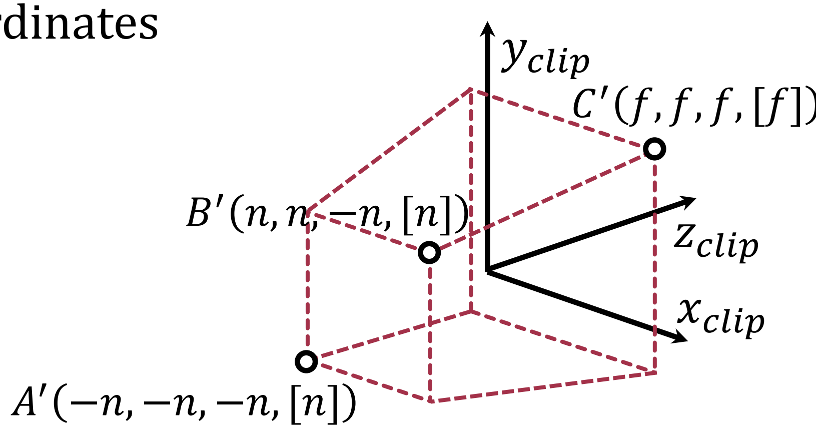
$$w' = -z$$

- Führt $\boldsymbol{M_{frustum}}$ perspektivische Projektion auf x, y, z Teil aus?
  - Nein. x und y noch nicht mit $1/z$ multipliziert.
- Was passiert:
  - Definiert sichtbaren Bereich
  - Enthält Info für perspektivische Projektion in w-Komponente
  - Aus rechtshändigem wird linkshändiges KS



Beispiel & Ausblick

- Gegeben:
  - Right, left, top, botton, near, far
  - Vertex mit Koordinate $\boldsymbol{p_{CC}}$ in Camera Coordinates
- Aufgaben
  1. Transformiere in Clipping Coordinates
  2. Stelle dann Sichtbarkeit fest
  3. Führe Perspective Projection aus
  4. Transformiere in xy-Ebene



- Lösung
  1. Berechne $\boldsymbol{p_{clip}} = \boldsymbol{M_{frustum}}(r, l, b, t, n, f) \cdot \boldsymbol{p_{cc}}$
  2. Falls $x, y, z \in [-w_p, w_p]$ ist der Vertex potentiell sichtbar
  3. Berechne $\boldsymbol{p_{ndc}} = \boldsymbol{p_{clip}}/w_p$ (Perspective Division, dann im CVV, später)
  4. Berechne: $\boldsymbol{P_o} \cdot \boldsymbol{p_{ndc}}$ (später)
- Sichtbarkeit nach Perspective Division?
  - Falls $x, y, z \in [-1, 1]$ ist der Vertex potentiell sichtbar

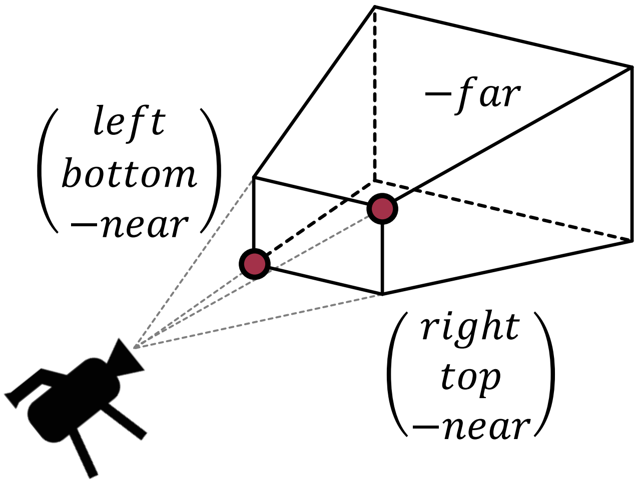
Fragen: z-Begrenzung

I. Warum soll nicht $z > 0$ sichtbar sein?
   - Wir wollen nichts hinter uns sehen

II. Warum nicht $z = 0$?
   - Ausdehnung in xy-Ebene dann verschwunden
   - Division durch 0

III. Warum Begrenzung $(z = -far)$ nach „hinten“
   - Ohne endliche Intervallgrenzen ev. Probleme mit Rechengenauigkeit
   - Wird so gewählt, dass es nicht „stört“
   - Gibt auch “nach hinten offene“ Varianten

IV. Was ist bei Wahl von $z = -near$ zu beachten?
   - Verhältnis zu $(right - left)$ muss gewünschtem FOV entsprechen
   - Möglichst nah, damit nichts abgeschnitten wird



Fragen: Implementation

- Viewing Transformation nötig ?
  - Nein
  - Camera Coordinates dann identisch mit World Coordinates
- “klassische“ parallele Projektion nötig ?
  - Ja, bereits implizit enthalten
  - CVV wird auf Bildebene abgebildet
- Perspective Projection nötig ?
  - Nein
- Implementation Viewing / Projection
  - Applikation
  - Vertex Shader
  - Anderer geometrieverarbeitender Shader
  - Typisch: Implementation im letzten geometrieverarbeitenden Schritt: Bei uns Vertex Shader

Viewing & Projection im VS

```
#version 330 core
in vec3 normalMC;
in vec4 posMC;

uniform mat4 mc2wc_Pos, view;
uniform mat3 mc2wc_Normal;
uniform vec3 inverseLightDir;

// Projection Transformation: Camera Coords -> NDC (OP) oder CC (PP)
uniform mat4 projection; // Perspektive oder Parallel Projection

out float brightness;

void main(){
  // 1) Objekte anordnen 2) Szenenbetrachtung 3) projizieren
  gl_Position   = projection * view * mc2wc_Pos * posMC;

  vec3 normalWC = mc2wc_Normal * normalMC;
  brightness = max(dot(normalWC, inverseLightDir), 0.0);
}
```

Auswirkungen der Projektionsmatrizen

In Klammern:
Sichtbarer Teil der Szene

IN

n Vertices

Camera Coordinates (RH)	
x	$\mathbb{R}$
y	$\mathbb{R}$
z	$\mathbb{R}$
w	1

$\boldsymbol{M_{ortho}}$ $\boldsymbol{M_{frustum}}$

OUT (M_{ortho})

n Vertices

Normalized Device Coords (LH)	
x	$[-1, 1]$
y	$[-1,1]$
z	$[-1,1]$
w	1

OUT ($M_{frustum}$)

n Vertices

Clipping Coordinates (LH)	
x	$[-w_p, w_p]$
y	$[-w_p, w_p]$
z	$[-w_p, w_p]$
w	$w_p = -z_{p,cc}$