Kapitel 13

3D-Transformationen

13.1 Translation

Mit homogenen Koordinaten läßt sich der um den Translationsvektor $t = (t_x, t_y, t_z)$ verschobene Punkt $P = (x, y, z)$

$$(x', y', z') := (x + t_x, y + t_y, z + t_z)$$

in der folgenden Form darstellen:

$$[x', y', z', 1] = [x, y, z, 1] \cdot T(t_x, t_y, t_z)$$

mit

$$T(t_x, t_y, t_z) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ t_x & t_y & t_z & 1 \end{bmatrix}$$

13.2 Skalierung

Es liege der Fixpunkt im Ursprung:

$$(x', y', z') := (x \cdot s_x, y \cdot s_y, z \cdot s_z)$$

Die daraus resultierende Transformationsmatrix lautet:

$$S(s_x, s_y, s_z) = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Es liege der Fixpunkt bei (Z_x, Z_y, Z_z):

1. Translation um $(\Leftrightarrow Z_x, \Leftrightarrow Z_y, \Leftrightarrow Z_z)$,
2. Skalierung um (s_x, s_y, s_z),
3. Translation um (Z_x, Z_y, Z_z).

Die Transformationsmatrix lautet:

$$T(\Leftrightarrow Z_x, \Leftrightarrow Z_y, \Leftrightarrow Z_z) \cdot S(s_x, s_y, s_z) \cdot T(Z_x, Z_y, Z_z)$$

13.3 Rotation

Rotation um die z-Achse

$$\begin{aligned} x' &:= x \cdot \cos(\delta) \Leftrightarrow y \cdot \sin(\delta) \\ y' &:= x \cdot \sin(\delta) + y \cdot \cos(\delta) \\ z' &:= z \end{aligned}$$

Die daraus resultierende Transformationsmatrix lautet:

$$R_z(\delta) = \begin{bmatrix} \cos(\delta) & \sin(\delta) & 0 & 0 \\ \Leftrightarrow \sin(\delta) & \cos(\delta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation um die x-Achse

$$\begin{aligned} x' &:= x \\ y' &:= y \cdot \cos(\delta) \Leftrightarrow z \cdot \sin(\delta) \\ z' &:= y \cdot \sin(\delta) + z \cdot \cos(\delta) \end{aligned}$$

Die daraus resultierende Transformationsmatrix lautet:

$$R_x(\delta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\delta) & \sin(\delta) & 0 \\ 0 & \Leftrightarrow \sin(\delta) & \cos(\delta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation um die y-Achse

$$\begin{aligned} x' &:= z \cdot \sin(\delta) + x \cdot \cos(\delta) \\ y' &:= y \\ z' &:= z \cdot \cos(\delta) \Leftrightarrow x \cdot \sin(\delta) \end{aligned}$$

Die daraus resultierende Transformationsmatrix lautet:

$$R_y(\delta) = \begin{bmatrix} \cos(\delta) & 0 & \Leftrightarrow\sin(\delta) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(\delta) & 0 & \cos(\delta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation um eine beliebige Achse

Voraussetzung: Die Rotationsachse stimme nicht mit einer der Koordinatenachsen überein.

Idee: Transformiere Rotationsachse und Objekt so, daß die Rotationsachse mit der z-Achse übereinstimmt, rotiere um vorgegebenen Winkel δ, transformiere zurück.

1. Translation von Rotationsachse (und Objekt), so daß die Rotationsachse durch den Ursprung läuft.
2. Rotation der Rotationsachse um die x-Achse in die xz-Ebene.
3. Rotation der Rotationsachse um die y-Achse in die z-Achse.
4. Rotation des Objekts um die z-Achse mit Winkel δ.
5. Rücktransformation des gedrehten Objekts durch Anwendung der inversen Transformationen der Schritte (3), (2) und (1).

Ist die Rotationsachse durch die Punkte P_1, P_2 gegeben, so gilt

$$v = P_2 \Leftrightarrow P_1 = \begin{pmatrix} x_2 \Leftrightarrow x_1 \\ y_2 \Leftrightarrow y_1 \\ z_2 \Leftrightarrow z_1 \end{pmatrix}.$$

Die Länge dieses Vektors lautet

$$|v| = \sqrt{(x_2 \Leftrightarrow x_1)^2 + (y_2 \Leftrightarrow y_1)^2 + (z_2 \Leftrightarrow z_1)^2}.$$

Die Komponenten des zugehörigen Einheitsvektors

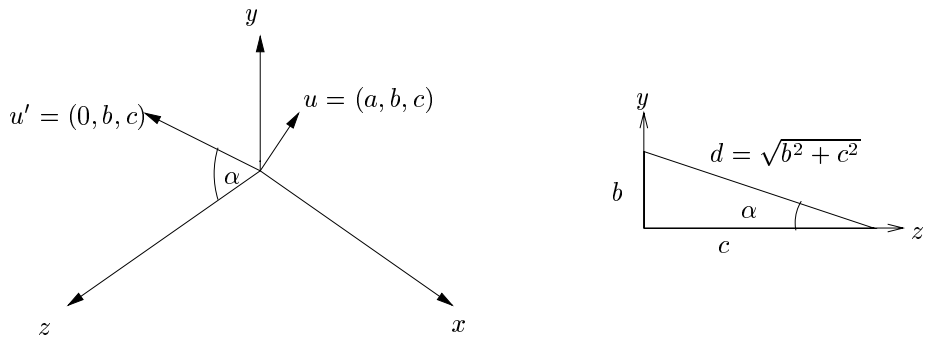
$$u = \frac{v}{|v|} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}, |u| = 1$$

lauten daher

$$a = \frac{x_2 \Leftrightarrow x_1}{|v|},\ b = \frac{y_2 \Leftrightarrow y_1}{|v|},\ c = \frac{z_2 \Leftrightarrow z_1}{|v|}.$$

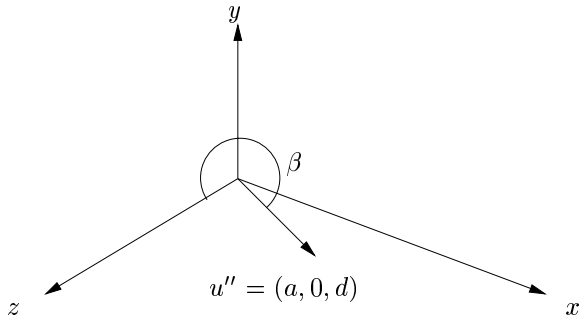
Schritt 1 läßt sich durch die Translation $T(\Leftrightarrow x_1, \Leftrightarrow y_1, \Leftrightarrow z_1)$ durchführen. Dadurch wird P_1, Ausgangspunkt des Einheitsvektors u, in den Ursprung verschoben.

Für Schritt 2 sind Sinus und Cosinus des Rotationswinkels α erforderlich, der zwischen der Projektion u' von u auf die yz-Fläche und der z-Achse, repräsentiert durch den Vektor $u_z = (0, 0, 1)$, liegt.

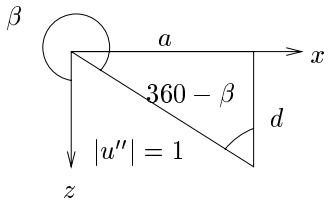


$\Rightarrow \cos(\alpha) = \frac{c}{d}$
$\Rightarrow \sin(\alpha) = \frac{b}{d}$

Nach Schritt 2 befindet sich der ursprüngliche Vektor u als u'' in der xz-Ebene:



Für Schritt 3 (Rotation um y-Achse) benötigt man Sinus und Cosinus des Rotationswinkels β.



$\Rightarrow \cos(\beta) = \cos(360^\circ \Leftrightarrow \beta) = d$
$\Rightarrow \sin(\beta) = \Leftrightarrow \sin(360^\circ \Leftrightarrow \beta) = \Leftrightarrow a$

Nach den ersten drei Schritten ist die Drehachse mit der z-Achse identisch, so daß Schritt (4) mit der Rotationsmatrix $R_z(\delta)$ durchgeführt werden kann. Schritt (5) beinhaltet die Anwendung der inversen Transformationen.

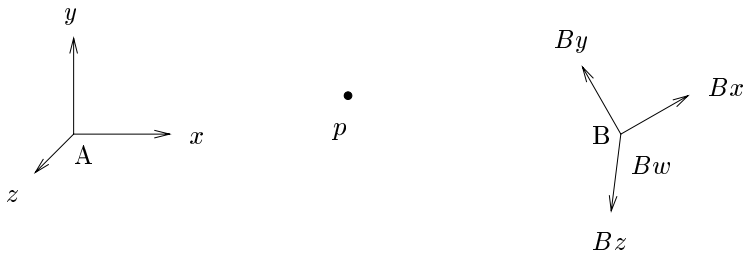
Die Rotation um die Achse $v = \overline{P_1 P_2}$ um den Winkel δ läßt sich daher wie folgt darstellen:

$$R(v, \delta) = T(\Leftrightarrow P_1) \cdot R_x(\alpha) \cdot R_y(\beta) \cdot R_z(\delta) \cdot R_y^{-1}(\beta) \cdot R_x^{-1}(\alpha) \cdot T(P_1)$$

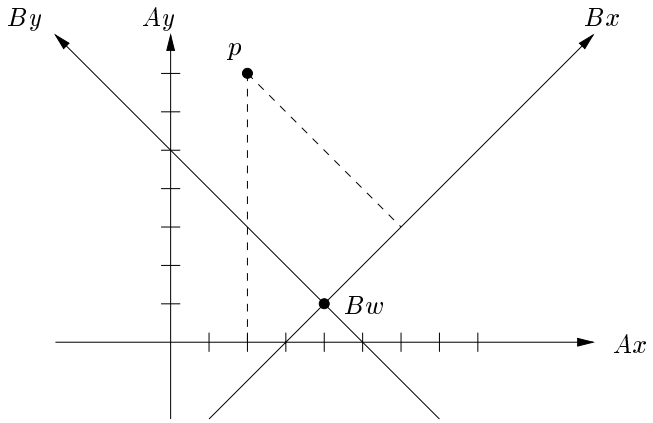
13.4 Transformation von Koordinatensystemen

Die Transformation von einem Koordinatensystem in ein anderes bedeutet einen Basiswechsel und läßt sich folgendermaßen formalisieren: Gegeben sei ein Koordinatensystem A, in dem ein anderes Koordinatensystem B durch die homogenen Koordinaten des Ursprungs (Bw) und die drei normierten Richtungsvektoren der Achsen (Bx, By, Bz) definiert ist. Zeilenweise angeordnet ergeben diese Vektoren die Matrix B_A, die den Übergang vom Koordinatensystem B zum Koordinatensystem A beschreibt:

$$B_A = \begin{bmatrix} Bx \\ By \\ Bz \\ Bw \end{bmatrix}$$



Beispiel (für den 2-dimensionalen Fall):



x-Achse:	Bx	lautet	$[\sqrt{\frac{1}{2}}, \sqrt{\frac{1}{2}}, 0]$
y-Achse:	By	lautet	$[\Leftrightarrow\sqrt{\frac{1}{2}}, \sqrt{\frac{1}{2}}, 0]$
Ursprung:	Bw	lautet	$[4, 1, 1]$
Punkt:	p_B	lautet	$[2 \cdot \sqrt{2}, 4 \cdot \sqrt{2}, 1]$

Der Aufbau der Matrix B_A repräsentiert die erforderliche Drehung und Verschiebung, um einen aus der Sicht von Koordinatensystem B beschriebenen Punkt p aus der Sicht von Koordinatensystem A zu beschreiben. Im zwei-dimensionalen Fall wird zunächst der Punkt

p um den Winkel α gedreht, der sich zwischen den Achsen der Koordinatensysteme A und B befindet. Dann wird eine Translation durchgeführt mit dem Wert der Ursprungsposition von B. Cosinus und Sinus des Drehwinkels α ergeben sich gerade aus den Werten a bzw. b, wobei die Einheitsvektoren (a, b) und $(\Leftrightarrow b, a)$ die Basis für Koordinatensystem B darstellen.

Also läßt sich in obigem Beispiel der Punkt p wie folgt transformieren:

$$[2 \cdot \sqrt{2}, 4 \cdot \sqrt{2}, 1] \cdot \begin{bmatrix} \sqrt{\frac{1}{2}} & \sqrt{\frac{1}{2}} & 0 \\ \Leftrightarrow\sqrt{\frac{1}{2}} & \sqrt{\frac{1}{2}} & 0 \\ 4 & 1 & 1 \end{bmatrix} = [2,\ 7,\ 1]$$

Um einen Punkt p_A des Koordinatensystems A im Koordinatensystem B zu spezifizieren, verwendet man die inverse Matrix zu B_A:

$$p_B \cdot (B_A) = p_A$$
$$\Rightarrow \quad p_B = p_A \cdot (B_A)^{-1}.$$

13.5 Transformation der Normalenvektoren

Die Normalenvektoren müssen bei der Transformation von Objektpunkten ebenfalls abgebildet werden. Allgemein gilt, daß bei der Transformation eines Objekts mit der Matrix M ein Normalenvektor n dieses Objekts mit der transponierten Inversen von M abgebildet werden muß:

$$n' = n \cdot (M^{-1})^T \ .$$

Beweis: Sei n der Normalenvektor einer Ebene, seien r_1 und r_2 beliebige Punkte in der Ebene. Betrachte n, r_1 und r_2 als Zeilenvektoren und arbeite mit dem Matrixprodukt (Zeile mal Spalte).

Dann gilt:

$$(r_1 \Leftrightarrow r_2) \cdot n^T = 0$$

Bei einem Wechsel des Koordinatensystems werden die Ortsvektoren r_1 und r_2 mit der Matrix M transformiert:

$$r_1' = r_1 \cdot M$$
$$r_2' = r_2 \cdot M$$

Es gilt:

$$\begin{aligned} & (r_1 \Leftrightarrow r_2) \cdot n^T && = 0 \\ \Leftrightarrow \quad & (r_1 \Leftrightarrow r_2) \cdot (M \cdot M^{-1}) \cdot n^T && = 0 \\ \Leftrightarrow \quad & ((r_1 \Leftrightarrow r_2) \cdot M) \cdot (M^{-1} \cdot n^T) && = 0 \\ \Leftrightarrow \quad & (r_1' \Leftrightarrow r_2') \cdot (n \cdot (M^{-1})^T)^T && = 0 \\ \Leftrightarrow \quad & (r_1' \Leftrightarrow r_2') \cdot (n')^T && = 0 \end{aligned}$$

$$\Rightarrow \quad n' = n \cdot (M^{-1})^T$$

Kapitel 14

Projektion

14.1 Bildebene

Für die Anzeige am zweidimensionalen Ausgabegerät muß eine Abbildung (*Projektion*) der räumlichen, dreidimensionalen Szene auf eine *Projektionsebene* erfolgen.

Gegeben sind

- das zu projizierende Objekt,
- die Bildebene,
- das Projektionszentrum.

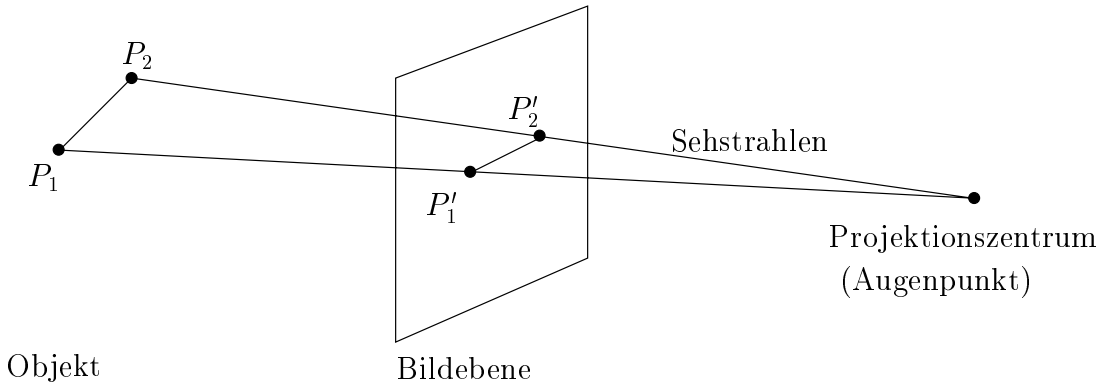
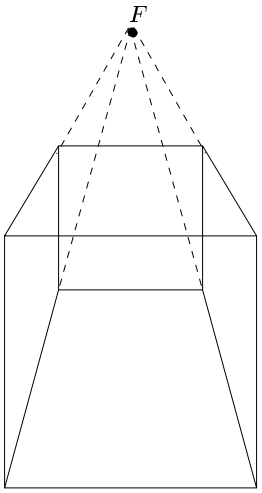


Abbildung 14.1: Zentralprojektion

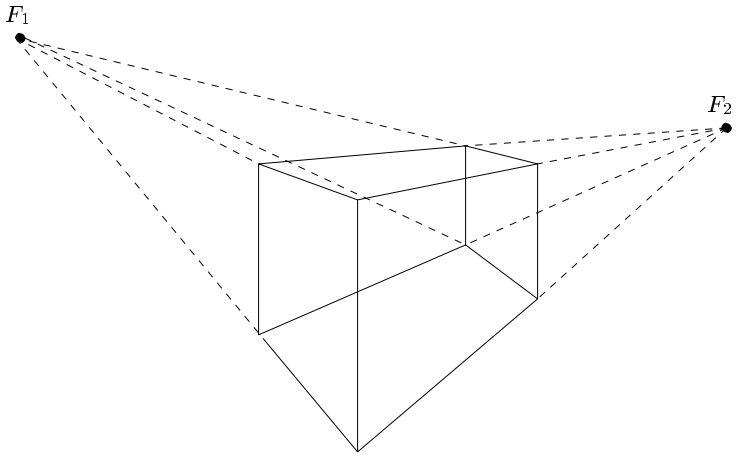
Ist der Abstand des Projektionszentrums von der Bildebene endlich, so handelt es sich um eine Zentralprojektion (perspektivische Projektion), andernfalls handelt es sich um eine Parallelprojektion (die Projektionsstrahlen sind zueinander parallel).

14.2 Perspektivische Projektion

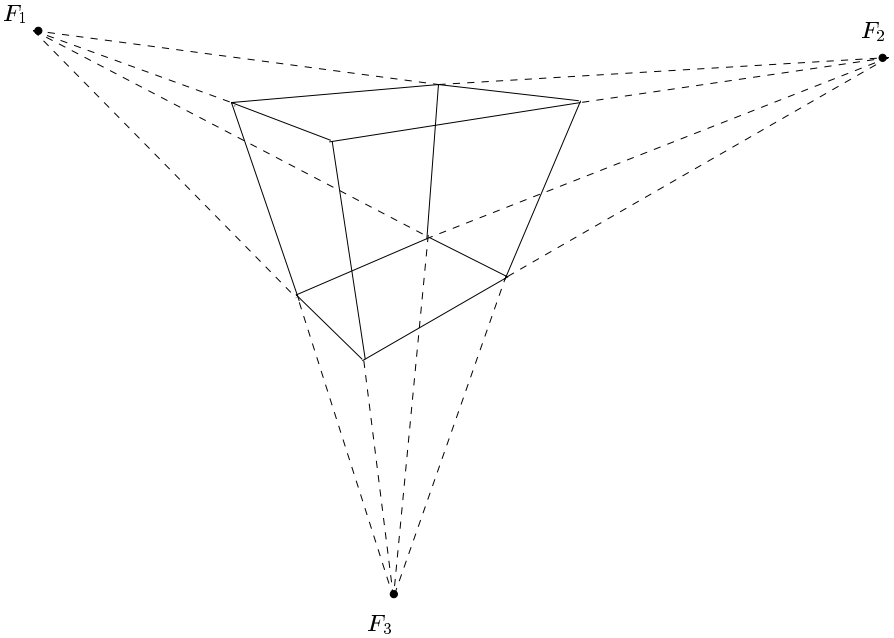
Die abgebildeten Objekte werden proportional zu ihrem Abstand zur Bildebene verkleinert. Je nachdem, ob die Bildebene eine, zwei oder drei der Koordinatenachsen schneidet, entstehen ein, zwei oder drei Fluchtpunkte.



1 Fluchtpunkt F



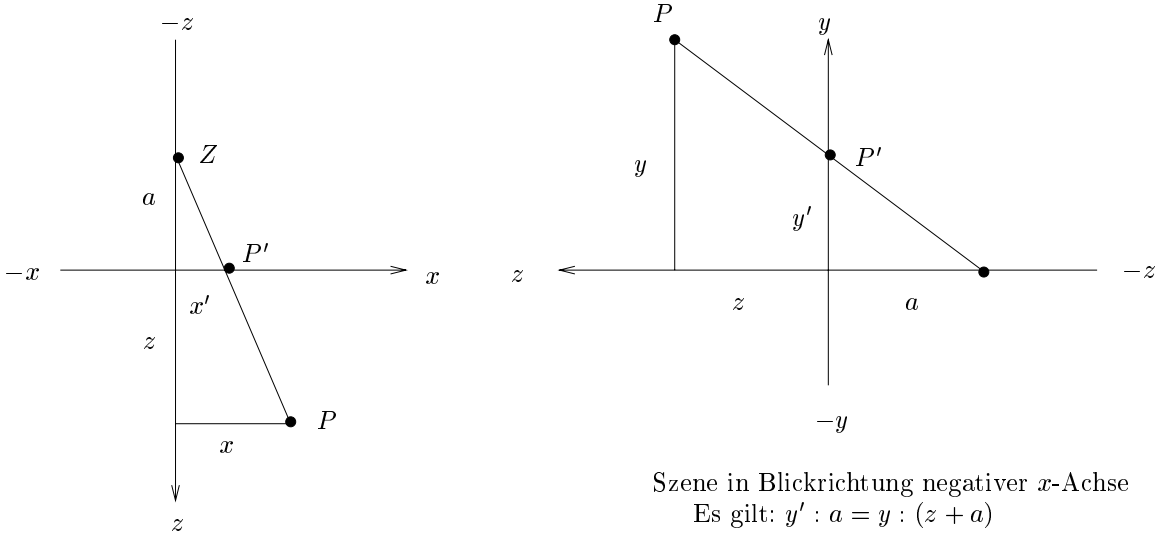
2 Fluchtpunkte F_1, F_2



3 Fluchtpunkte F_1, F_2, F_3

Abbildung 14.2: Zentralprojektionen mit unterschiedlich vielen Fluchtpunkten

OBdA sei die Bildebene gleich der xy-Ebene, und das Projektionszentrum liege auf der negativen z-Achse im Punkt $Z = (0, 0, \Leftrightarrow a)$. Gegeben Punkt $P = (x, y, z)$. Gesucht sind auf der Bildebene die Koordinaten des projizierten Bildpunktes $P' = (x', y', 0)$.



Szene in Blickrichtung negativer y-Achse
Es gilt: $x' : a = x : (a + z)$

Szene in Blickrichtung negativer x-Achse
Es gilt: $y' : a = y : (z + a)$

Abbildung 14.3: Anwendung der Strahlensätze

Betrachtet man die Szene "von oben" und "von der Seite", so erhält man aufgrund der Strahlensätze die Beziehung

$$x' = \frac{x}{1 + z/a} , \qquad y' = \frac{y}{1 + z/a} , \qquad z' = 0.$$

Die homogenen Koordinaten des projizierten Punktes lauten

$$P' = [x/w, y/w, 0, 1] = [x, y, 0, w] \text{ mit } w = 1 + z/a.$$

Für die Transformationsmatrix der perspektivischen Projektion ergibt sich also:

$$P_{persp_{xy}}(\Leftrightarrow a) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1/a \\ 0 & 0 & 0 & 1 \end{bmatrix} .$$

14.3 Parallelprojektion

14.3.1 Normalprojektionen

Stehen die Sehstrahlen normal zur Bildebene, liegt eine *orthogonale* Projektion vor.

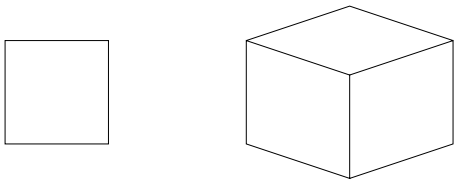


Abbildung 14.4: Grund-, Seiten-, Aufriss (a) bzw. axononometrische Projektion (b)

Die häufigste Anwendung orthogonaler Projektionen liegt in der Darstellung eines Objekts durch *Grund-*, *Auf-* und *Seitenriß* (Abbildung 14.4 a).

Eine weitere Form der Normalprojektionen sind die *axonometrischen Projektionen*, bei denen die Bildebene auf keiner der Körper-Hauptachsen senkrecht steht. Dadurch sind in der Abbildung mehrere zueinander normal stehende Flächen gleichzeitig sichtbar. Die resultierenden Darstellungen sind der perspektivischen Projektion ähnlich. Anstelle der zur Entfernung vom Auge proportionalen Verkürzung erfolgt aber eine gleichmäßige Verkürzung aller Kanten (Abbildung 14.4 b).

Die Transformationsmatrix für die orthogonale Parallelprojektion auf die xy-Ebene lautet

$$P_{ortho_{xy}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

da für jeden Punkt $P = [x, y, z, 1]$ die Koordinaten des projizierten Punktes gleich $[x, y, 0, 1]$ sind.

14.3.2 Schiefe Projektionen

Bei den *schiefen* Parallelprojektionen stehen die Sehstrahlen nicht normal auf der Bildebene, sondern schneiden sie unter dem Winkel β (Abbildung 14.5). Die schiefe Projektion auf die xy-Ebene entspricht einer Scherung der x- und y-Koordinaten proportional zu z.

Seien $P_0 = (x, y, 0)$ die orthogonale und $P' = (x', y', 0)$ die schiefe Projektion von Punkt $P = (x, y, z)$. Sei L die Entfernung von P_0 nach P'. Dann gilt

$$\begin{aligned} x' &= x \Leftrightarrow L \cdot \cos(\alpha) \\ y' &= y + L \cdot \sin(\alpha) \\ z' &= 0 \end{aligned}$$

Wegen $\tan(\beta) = \frac{z}{L}$ folgt

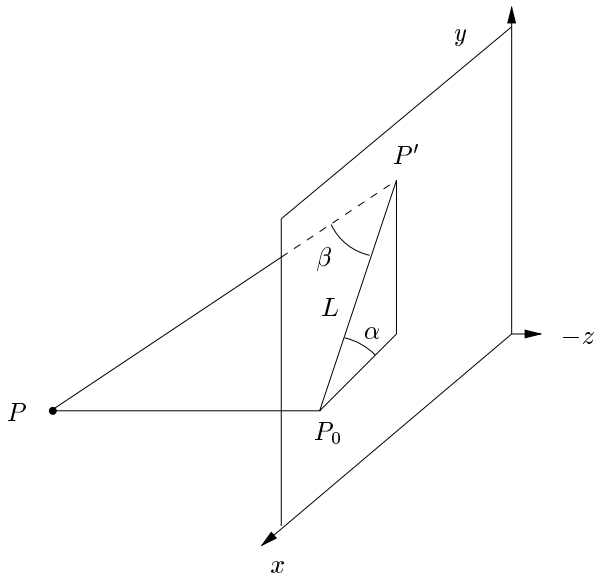


Abbildung 14.5: Bildebene bei der schiefen Projektion

$$x' = x \Leftrightarrow z \cdot \frac{\cos(\alpha)}{\tan(\beta)}$$
$$y' = y + z \cdot \frac{\sin(\alpha)}{\tan(\beta)}$$

Der Winkel β regelt im projizierten Bild das Verhältnis von x-Ausdehnung zu z-Ausdehnung. Die Koordinaten zweier projizierter Punkte P_1', P_2' lauten:

$$x_1' = x_1 \Leftrightarrow z_1 \cdot \cos(\alpha)/\tan(\beta)$$
$$y_1' = y_1 + z_1 \cdot \sin(\alpha)/\tan(\beta)$$
$$x_2' = x_2 \Leftrightarrow z_2 \cdot \cos(\alpha)/\tan(\beta)$$
$$y_2' = y_2 + z_2 \cdot \sin(\alpha)/\tan(\beta)$$

Für zwei Punkte, die sich nur bzgl. ihrer z-Koordinaten unterscheiden, betragen die Abstände ihrer Bilder in x- bzw. y-Richtung

$$|x_1' \Leftrightarrow x_2'| = |(z_1 \Leftrightarrow z_2) \cdot \cos(\alpha)/\tan(\beta)|$$
$$|y_1' \Leftrightarrow y_2'| = |(z_1 \Leftrightarrow z_2) \cdot \sin(\alpha)/\tan(\beta)|$$

für den Abstand $|P_1' \Leftrightarrow P_2'| = \sqrt{|x_1' \Leftrightarrow x_2'|^2 + |y_1' \Leftrightarrow y_2'|^2}$ ergibt sich

$$|P_1' \Leftrightarrow P_2'| = \sqrt{\frac{(z_1 \Leftrightarrow z_2)^2}{\tan^2(\beta)} \cdot (\cos^2(\alpha) + \sin^2(\alpha))} = \frac{z_1 \Leftrightarrow z_2}{\tan(\beta)}$$

wegen $\cos^2(\alpha) + \sin^2(\alpha) = 1$.

Für die Berechnung der Transformationsmatrix benötigt der Algorithmus den Verkürzungsfaktor d und den Scherwinkel α. d gibt an, um welchen Faktor zur Bildebene normal stehende

Strecken verkürzt werden. Es gilt

$$d = \frac{1}{\tan(\beta)} .$$

α definiert den Winkel zur Horizontalen, unter dem diese Kanten aufgetragen werden. Für die Koordinaten des so projizierten Punktes $P = (x, y, z, 1)$ gilt

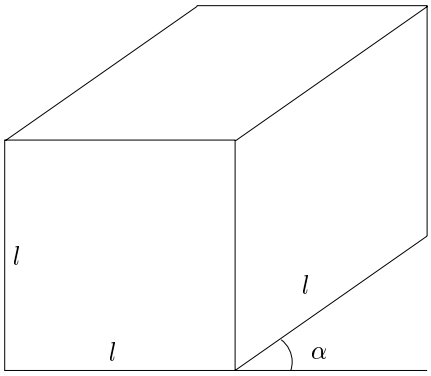
$$\begin{aligned} x' &= x \Leftrightarrow z \cdot (d \cdot \cos(\alpha)), \\ y' &= y + z \cdot (d \cdot \sin(\alpha)), \\ z' &= 0, \\ w' &= 1 \end{aligned}$$

Die entsprechende Transformationsmatrix lautet

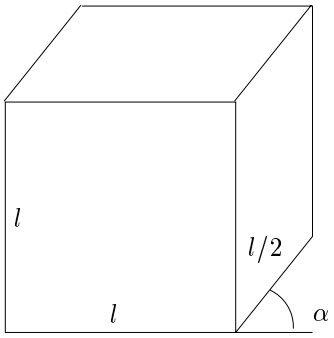
$$P_{schief_{xy}}(\alpha, d) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \Leftrightarrow d \cdot \cos(\alpha) & d \cdot \sin(\alpha) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} .$$

Für $d = 0$ ($\beta = 90°$) ergibt sich daraus die orthogonale Projektion. Bei schiefen Projektionen ist der Wert für d immer ungleich Null.

Zwei häufig als Ersatz für Perspektive verwendete Projektionen haben die Werte $d = 1$ ($\beta = 45°$), $\alpha = 35°$ (*Kavalierprojektion*) und $d = 0.5$ ($\beta = 63.43°$), $\alpha = 50°$ (*Kabinettprojektion*). Bei der Kavalierprojektion werden alle auf der Bildebene normal stehenden Strecken unverkürzt abgebildet. Bei der Kabinettprojektion ergibt sich eine Verkürzung auf die Hälfte ihrer ursprünglichen Länge.



Kavalierprojektion: $\beta = 45°$
kombiniert mit $\alpha = 35°$



Kabinettprojektion: $\beta = 63.43°$
kombiniert mit $\alpha = 50°$

Abbildung 14.6: Zwei schiefe Projektionen

Kapitel 15

Betrachtungstransformationen

Die Abbildung dreidimensionaler Objekte auf den Bildschirm wird in eine Reihe von Elementartransformationen zerlegt:

- Konstruktion von komplexen Szenen aus elementaren Objekten (*Modeling*),
- Festlegen der Bildebene (*View Orientation*),
- Projektion auf ein normiertes Gerät (*View Mapping*),
- Abbildung auf ein Ausgabegerät (*Device Mapping*).

15.1 Die synthetische Kamera

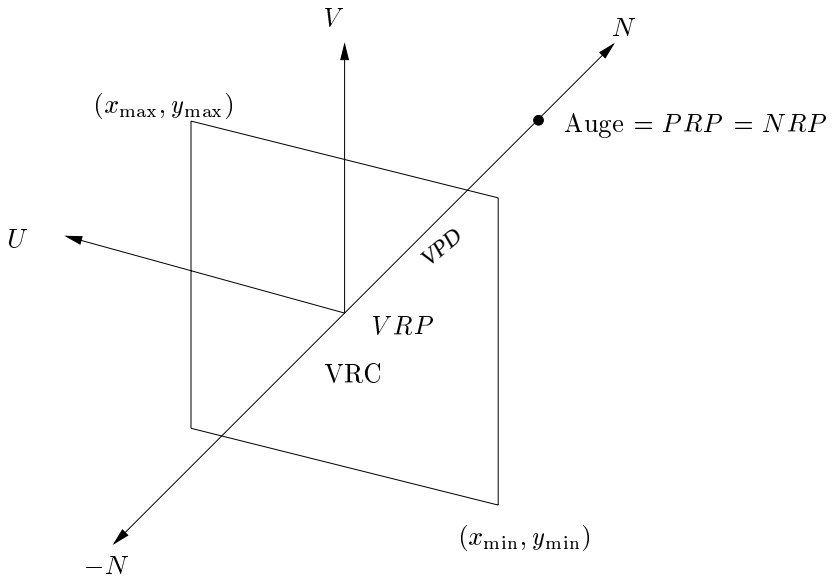


Abbildung 15.1: Parameter zur Beschreibung der syntetischen Kamera

Die Parameter zur Durchführung der 3D-auf-2D-Abbildung (*viewing transformation*) sind im wesentlichen die Position und Orientierung einer *synthetischen Kamera* und ein Punkt in der Szene, auf den die Kamera fokussiert ist: Der View Reference Point (VRP).

Die View Plane ist die Bildebene, auf die die Szene projiziert wird. Diese ist definiert durch die View Plane Normal N, die Richtung, aus der die Kamera die Szene aufnimmt, und den VRP. Die Kamera befindet sich im NRP (Projection oder Normal Reference Point), der dem Augenpunkt eines natürlichen Betrachters entspricht.

Die Orientierung der Kamera wird durch den View Up Vector (VUV) festgelegt. Durch Projektion des VUV in die View Plane erhält man den Vektor V. Der Vektor U in der View Plane wird so gewählt, daß U, V, N ein rechtshändiges kartesisches Koordinatensystem bilden, das sogenannte View Reference Coordinate System VRC mit dem VRP als Ursprung und dem Augenpunkt NRP auf der positiven N-Achse bei $N = VPD$ (View Plane Distance).

Die beiden Punkte in den UV-Koordinaten der Viewplane $(x_{\min}, y_{\min})$, $(x_{\max}, y_{\max})$ legen das sogenannte View Window fest, d.h., was von der Szene zu sehen ist (entspricht der Brennweite eines Kameraobjektivs). Der sichtbare Teil kann noch weiter eingeschränkt werden durch Angabe einer *front plane* und einer *back plane*.

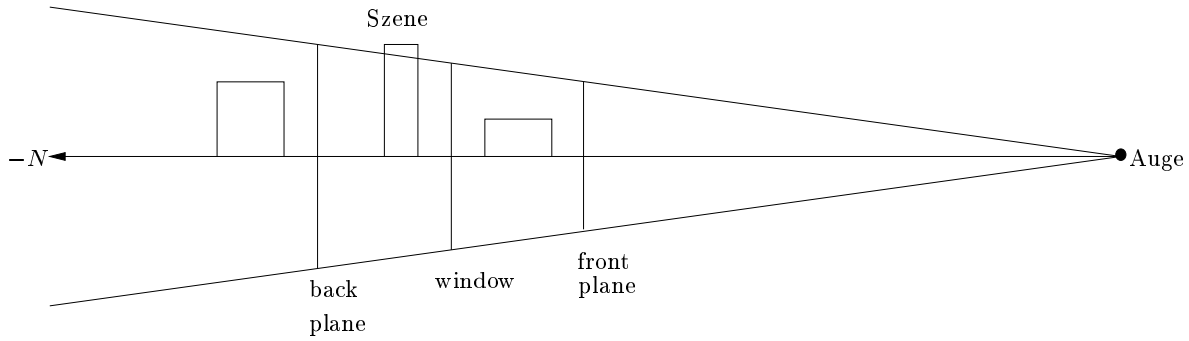


Abbildung 15.2: Einschränkung des sichtbaren Teils

Bemerkung: Zur Berechnung der Projektion auf die Bildebene ist es unerheblich, wo die Bildebene in der Szene arrangiert ist.

15.2 Viewing Pipeline

Folgende Informationen sind erforderlich:

- Jedes Objekt wird beschrieben durch Modellkoordinaten (z.B. Würfel ist beschrieben durch Mittelpunkt $(0, 0, 0)$ und Kantenlänge 1).
- Die Szene wird beschrieben durch eine Menge von Objekten, deren Lage und Größe in Weltkoordinaten beschrieben sind.
- Die synthetische Kamera wird beschrieben durch U, V, N, VPD, $(x_{\max}, y_{\max})$, $(x_{\min}, y_{\min})$.

Folgende Abkürzungen werden verwendet:

MC	model coordinate system
WC	world coordinate system
VRC	view reference coordinate system
NPC	normalized projection coordinate system
DC	device coordinate system

Jedes Polygon durchläuft die folgende Viewing-Pipeline:

1.) Modeling: MC $\rightarrow$ WC
Beschreibe Polygonpunkte in Weltkoordinaten (dies geschieht durch Translation, Rotation und Skalierung).

2.) View Orientation: WC $\rightarrow$ VRC
Beschreibe Polygonpunkte bzgl. des UVN-Systems (dies geschieht durch Wechsel des Koordinatensystems, anschließend ist die Szene beschrieben mit xy-Ebene = Bildebene, das Auge liegt bei $z = VPD$).

3.) View Mapping: VRC $\rightarrow$ NPC
Transformiere die Szene derart, daß der sichtbare (= im Pyramidenstumpf aus view window, front plane, back plane liegende) Teil abgebildet wird auf einen Einheitswürfel, dessen Vorder- und Rückseite der *front plane* bzw. *back plane* entsprechen. Clippe jede Polygonkante am Einheitswürfel.

4.) Device Mapping: NPC $\rightarrow$ DC
Bilde jeden Bildpunkt auf Gerätekoordinaten ab = übernimm x, y-Koordinaten; z liefert Tiefeninformation.

Beim Flächenmodell werden nach dem Clipping in Schritt 3.) die Polygone in Dreiecke zerlegt. Die Dreiecke sind versehen mit den Normalen bzgl. des Weltkoordinatensystems. Diese werden benötigt für das Schattieren der Dreiecke.

15.2.1 Modeling-Transformationen

Punkt 1.) wird *Modeling* genannt: Das Anordnen von Objekten aus dem *Modellkoordinatensystem* (MC) zu einer Szene in dem sogenannten *Weltkoordinatensystem* (WC). Im MC liegen die Objekte als Prototypen vor. Ihre Definitionspunkte sind unabhängig von der späteren Größe und Position im WC in Modellkoordinaten gegeben. Der Ursprung des MC befindet sich sinnvollerweise im Zentrum des Objekts. Erst durch das Modeling erhalten die Objekte im WC ihre individuelle Größe, Orientierung und Position. Dieses wird im wesentlichen durch drei Arten von Transformationen realisiert: Translation, Rotation und Skalierung.

15.2.2 View Orientation

Zur Abbildung einer dreidimensionalen Szene aus den Weltkoordinaten auf den Bildschirm muß eine natürliche Betrachtersicht (*View Orientation*) definiert werden. Eine solche Betrachtersicht besteht aus den Viewing-Parametern

- Betrachterstandpunkt NRP (*Normal Reference Point*; auch *Eyepoint*),
- Blickrichtung VRP (*View Reference Point*),
- vertikale Orientierung VUV (*View Up Vector*) oder VUP (*View Up Point*),
- Blickwinkel (Brennweite).

Der fiktive Betrachter blickt vom NRP in Richtung VRP und kippt dabei die Kamera so um die Blickrichtung, daß VUP von ihm aus gesehen vertikal nach oben zeigt. Der Blickwinkel entspricht der Brennweite und regelt den Bildausschnitt (*Zooming*).

Durch diese Parameter wird das VRC (*View Reference Coordinate System*) definiert mit VRP als Ursprung und NRP auf der positiven z-Achse. Die xy-Ebene wird als *View Plane* bezeichnet und steht senkrecht auf der Blickrichtung. Die y-Achse ist gegeben durch die Projektion von VUP in die View Plane. Deshalb darf VUP nicht parallel zur Blickrichtung sein.

Das VRC hat die gleiche Metrik wie das WC. Es handelt sich also um einen Wechsel des Koordinatensystems.

Der Abstand zwischen VRP und NRP wird als VPD (*View Plane Distance*) bezeichnet.

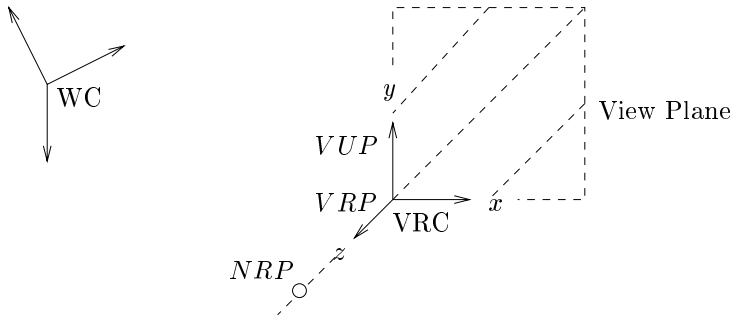


Abbildung 15.3: Definition des VRC

Für die Anwendung des Viewings ergeben sich im Programm verschiedene Möglichkeiten:

- Soll ein Objekt von verschiedenen Seiten betrachtet werden, so wird VRP in das Objekt gelegt. Durch NRP läßt sich nun die Blickrichtung frei wählen. Der Vektor VUP kann dabei konstant bleiben, sofern nicht senkrecht von oben oder unten geschaut wird.
- Soll sich die Szene bei konstantem Standpunkt um den Betrachter drehen, wird NRP festgesetzt. Durch Veränderung des VRP bewegt sich dann die Szene vor dem Auge des Betrachters. Der VUP kann dabei konstant bleiben, solange er nicht kollinear zum Vektor durch VRP und NRP wird.
- Eine weitere Möglichkeit sind Animationen, bei denen sich z.B. der Beobachter vorwärts in die Szene hineinbewegt. Dieser Effekt läßt sich durch Verschieben von VRP und NRP um denselben Vektor erzielen.

15.2.3 View Volume

Bei einer Kamera ist der maximal abbildbare Ausschnitt einer Szene durch das Objektiv festgelegt. Im Programm wird zur Erzielung desselben Effekts ein rechteckiger Ausschnitt aus der Bildebene — das *Window* — gewählt. Das Seitenverhältnis entspricht dem des Ausgabegeräts. Durch das View Window und die gewählte Projektion wird ein Bildraum definiert, der *View Volume* genannt wird. Nur jene Objekte und Objektteile, die sich innerhalb dieses Bildraumes befinden, werden auf die Bildebene (und dadurch in das View Window) abgebildet.

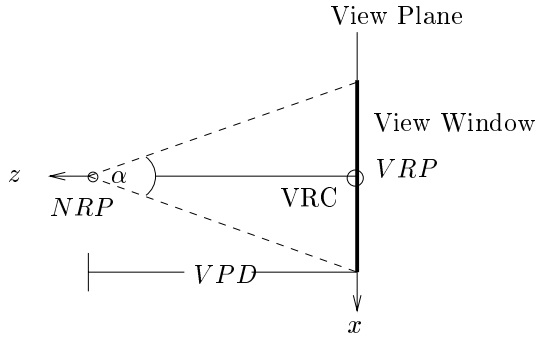


Abbildung 15.4: Definition des View Windows

Die Form des Bildraumes ist bei der perspektivischen Projektion eine unendliche Pyramide, deren Spitze im Projektionszentrum PRP (*Projection Reference Point*) liegt und deren Kanten durch die Eckpunkte des View Windows verlaufen. Hierbei wird im Programm der Betrachterstandpunkt NRP als Projektionszentrum gewählt.

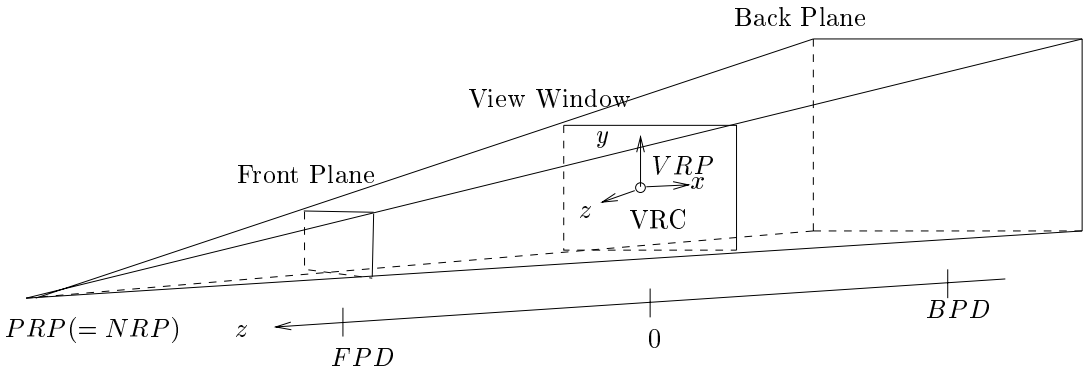


Abbildung 15.5: Bildraum bei perspektivischer Projektion

Die Begrenzung des Bildraumes in Richtung der z-Achse des VRC erfolgt durch zwei zur View Plane parallele Ebenen, die *Front* und *Back Plane* genannt werden. Die Front Plane liegt vom Projektionszentrum PRP aus gesehen vor der Back Plane. Ihre z-Komponenten werden als *Front Plane Distance* (FPD) und *Back Plane Distance* (BPD) bezeichnet. Es gilt $BPD < FPD$. Mit diesen Ebenen können Teile der Szene von der Projektion auf die Bildebene ausgeschlossen werden. Speziell bei der perspektivischen Projektion erweist sich diese Möglichkeit als notwendig, da sonst sehr nahe Objekte alle anderen verdecken bzw. sehr

weit entfernte als nicht mehr erkennbare (d.h. zu kleine) Figuren abgebildet würden. Mit Front und Back Plane ergibt sich als View Volume ein Pyramidenstumpf.

15.2.4 View Mapping

Statt nun das View Volume aus dem VRC direkt auf den Bildschirm zu projizieren, wird im Programm eine weitere Transformation durchgeführt, die nicht nur zur Effizienzsteigerung des Algorithmus führt, sondern auch die Projektion der Szene auf die Bildebene erleichert: Der Bildraum wird in einen zur Bildebene normal ausgerichteten Einheitswürfel ($0 \leq x \leq 1; 0 \leq y \leq 1; 0 \leq z \leq 1$) umgewandelt. Anstelle mehrerer unterschiedlicher Projektionen muß so anschließend nur noch die orthogonale Parallelprojektion auf die Ebene $z = 0$ durchgeführt werden. Das sogenannte *View Mapping* wird deshalb auch als Ausgabe auf einen normierten Bildschirm bezeichnet. Das Koordinatensystem, in dem sich der Einheitswürfel befindet, heißt *Normalized Projection Coordinate System* (NPC). Der Betrachterstandpunkt befindet sich im Unendlichen auf der positiven z-Achse und hat die homogenen Koordinaten $(0, 0, 1, 0)$. Vom Betrachter aus gesehen liegt im NPC der Punkt $(0, 0, 0)$ "links unten hinten" und $(1, 1, 1)$ "rechts oben vorne".

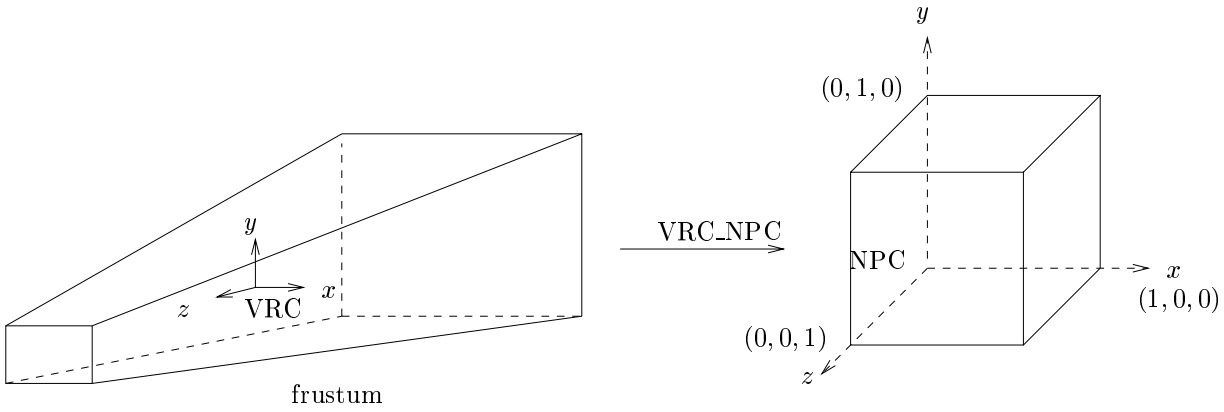


Abbildung 15.6: View Mapping

Bei der perspektivischen Projektion wird in einem ersten Schritt der Pyramidenstumpf, der den Bildraum darstellt und *frustum* genannt wird, auf einen regelmäßigen Pyramidenstumpf abgebildet. Dessen Grundfläche ist ein Quadrat, das mit jeder Seitenfläche einen Winkel von 45° einschließt. Dabei werden die z-Koordinaten nicht und die x- und y-Koordinaten proportional zur z-Koordinate verändert, was einer Scherung an der z-Achse entspricht. Im zweiten Schritt wird der regelmäßige Pyramidenstumpf in den normierten Einheitswürfel transformiert. Dazu müssen die x- und y-Koordinaten proportional zu den reziproken z-Werten skaliert werden. Die Front Plane entspricht im NPC der Ebene $z = 1$ und die Back Plane der Ebene $z = 0$.

Beide Schritte werden im Programm zu einer Transformationsmatrix zusammengefaßt. Bei deren Anwendung auf die homogenen Koordinaten ist zu beachten, daß die resultierenden Punkte im allgemeinen w-Werte ungleich 1 haben, die affinen Koordinaten also erst nach der Division durch w vorliegen. Außerdem ist diese Abbildung nur bezüglich der x- und y-

Koordinaten linear. In z-Richtung werden die Werte durch die Scherung im zweiten Schritt reziprok verzerrt, d.h., äquidistante Punkte längs der z-Achse im VRC häufen sich im NPC bei $z = 0$ nahe der Back Plane.

Zur Erleichterung der Herleitung wird zunächst das Koordinatensystem so transformiert, daß der PRP im Ursprung sitzt. Danach wird das Koordinatensystem an der xy-Ebene gespiegelt, indem die z-Koordinaten mit -1 multipliziert werden. Danach ist das Koordinatensystem linkshändig. Es seien d_{min}, d, d_{max} die Abstände der Frontplane, Bildebene, Backplane vom Augenpunkt.

Zur Durchführung von Punkt 3.) der Viewing Pipeline wird zunächst die abgeschnittene Pyramide (= frustum) transformiert in einen symmetrischen Pyramidenstumpf mit quadratischer Grundfläche und Kanten unter 45°.

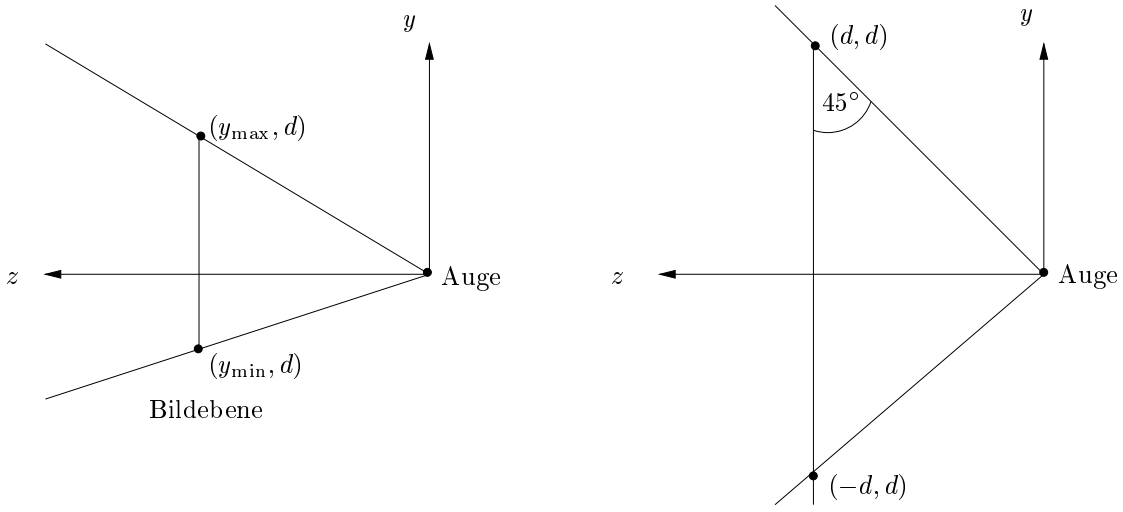


Abbildung 15.7: Überführung in Pyramidenstumpf

Es müssen also die y-Koordinaten proportional zur z-Koordinate verändert werden. Dies entspricht einer Scherung an der z-Achse und Skalierung, d.h.

$$\begin{aligned} y' &= k_1 \cdot y + k_2 \cdot z \\ z' &= z \end{aligned}$$

Punkt $(y_{\max}, d)$ soll abgebildet werden auf (d, d);
Punkt $(y_{\min}, d)$ soll abgebildet werden auf $(\Leftrightarrow d, d)$.

Durch Lösen des Gleichungssystems

$$\begin{aligned} d &= k_1 \cdot y_{\max} + k_2 \cdot d \\ \Leftrightarrow d &= k_1 \cdot y_{\min} + k_2 \cdot d \end{aligned}$$

erhält man

$$k_1 = \frac{2d}{y_{\max} \Leftrightarrow y_{\min}}, \quad k_2 = \Leftrightarrow \frac{y_{\max} + y_{\min}}{y_{\max} \Leftrightarrow y_{\min}}$$

Analoge Scherung der x-Werte ergibt:

$$k_1 = \frac{2d}{x_{\max} \Leftrightarrow x_{min}}, \quad k_2 = \Leftrightarrow \frac{x_{\max} + x_{\min}}{x_{max} \Leftrightarrow x_{min}}$$

Als nächstes wird die regelmäßige Pyramide in den Einheitswürfel ($0 \leq x \leq 1, 0 \leq y \leq 1, 0 \leq z \leq 1$) transformiert. Die *front plane* entspricht der Ebene $z = 0$ und die *back plane* der Ebene $z = 1$.

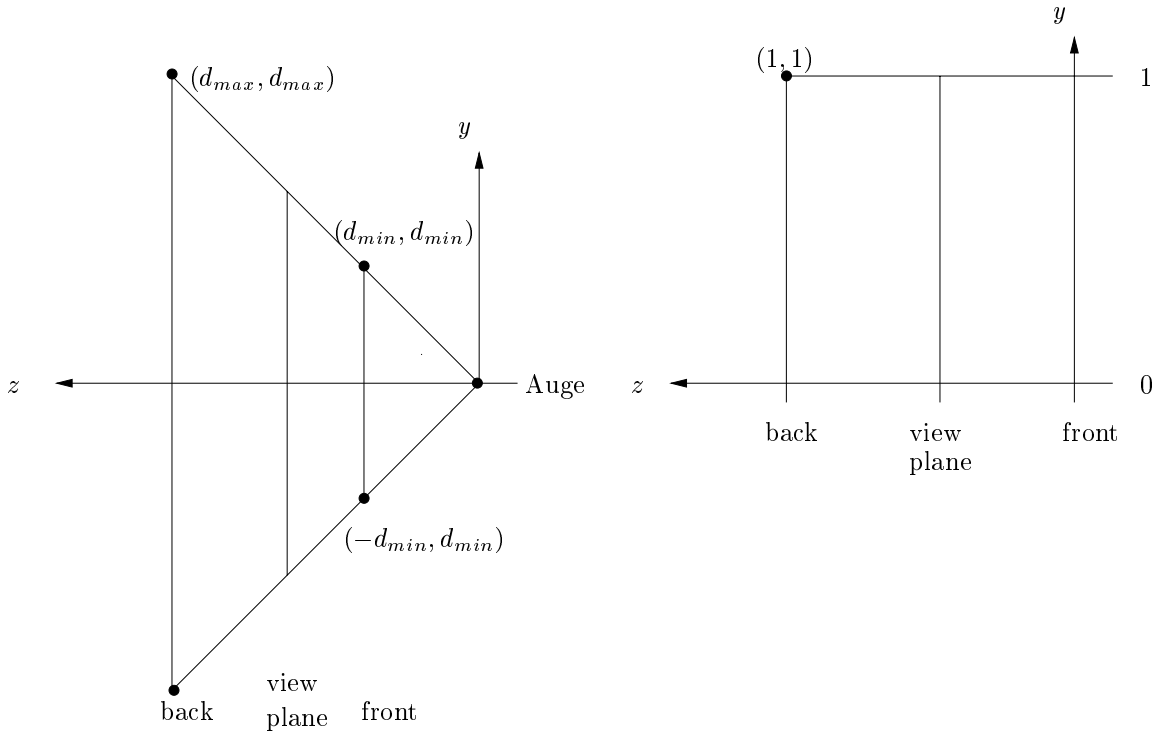


Abbildung 15.8: Überführung in Einheitswürfel

Da die y-Werte proportional zu den reziproken z-Werten skaliert werden müssen, ergibt sich als Transformation

$$\begin{aligned} y' &= k_1 + k_2 \cdot \frac{y}{z} \\ z' &= k_3 + k_4 \cdot \frac{1}{z} \end{aligned}$$

Der Kehrwert von z im Term zu z' ist erforderlich wegen der Verwendung von homogenen Koordinaten mit 4×4-Transformationsmatrizen.

Punkt $(d_{\max}, d_{\max})$ soll abgebildet werden auf $(1, 1)$,
Punkt $(\Leftrightarrow d_{\min}, d_{\min})$ soll abgebildet werden auf $(0, 0)$.
Durch Lösen des Gleichungssystems

$$\begin{aligned} 1 &= k_1 + k_2 \cdot \frac{d_{\max}}{d_{\max}} \\ 0 &= k_1 + k_2 \cdot \frac{\Leftrightarrow d_{\min}}{d_{\min}} \end{aligned}$$

erhält man

$$k_1 = \frac{1}{2}, \quad k_2 = \frac{1}{2}.$$

Durch Lösen des Gleichungssystems

$$\begin{aligned} 1 &= k_3 + k_4 \cdot \frac{1}{d_{\max}} \\ 0 &= k_3 + k_4 \cdot \frac{1}{d_{\min}} \end{aligned}$$

erhält man

$$k_3 = \frac{d_{\max}}{d_{\max} \Leftrightarrow d_{\min}}, \quad k_4 = \Leftrightarrow \frac{d_{\min} \cdot d_{\max}}{d_{\max} \Leftrightarrow d_{\min}}.$$

Die x-Werte werden analog zu den y-Werten skaliert. Insgesamt ergibt sich somit

$$\begin{aligned} x' &= \frac{1}{2} + \frac{x}{2} \cdot \frac{1}{z} \\ y' &= \frac{1}{2} + \frac{y}{2} \cdot \frac{1}{z} \\ z' &= \frac{d_{\max}}{d_{\max} \Leftrightarrow d_{\min}} \Leftrightarrow \frac{d_{\min} \cdot d_{\max}}{d_{\max} \Leftrightarrow d_{\min}} \cdot \frac{1}{z}. \end{aligned}$$

Zwar wird hierdurch die Szene im vorderen z-Bereich nicht-linear gestaucht, zur Bestimmung der Sichtbarkeit reichen die ermittelten z-Werte jedoch aus, da ihre Ordnung erhalten bleibt.

Durch Verknüpfen der beiden letzten Transformationen erhält man in Schritt 3.) als Transformationsmatrix

$$\begin{bmatrix} \frac{d}{x_{\max} \Leftrightarrow x_{\min}} & 0 & 0 & 0 \\ 0 & \frac{d}{y_{\max} \Leftrightarrow y_{\min}} & 0 & 0 \\ \frac{1}{2}(1 \Leftrightarrow \frac{x_{\min} + x_{\max}}{x_{\max} \Leftrightarrow x_{\min}}) & \frac{1}{2}(1 \Leftrightarrow \frac{y_{\min} + y_{\max}}{y_{\max} \Leftrightarrow y_{\min}}) & \frac{d_{\max}}{d_{\max} \Leftrightarrow d_{\min}} & 1 \\ 0 & 0 & \frac{\Leftrightarrow d_{\max} \cdot d_{\min}}{d_{\max} \Leftrightarrow d_{\min}} & 0 \end{bmatrix}$$

die den sichtbaren Teil der Szene in den Einheitswürfel transformiert.

Um wieder ein rechtshändiges Koordinatensystem zu erhalten, wird zunächst das Koordinatensystem so transformiert, daß die Back Plane in die xy-Ebene verschoben wird. Abschließend wird wieder an der xy-Ebene gespiegelt.

Für das Clippen von Kanten am Einheitswürfel bietet sich die dreidimensionale Erweiterung des Algorithmus von Cohen & Sutherland an: Der Bereichs-Code benötigt 6 Bits.

15.2.5 Device Mapping

Die abschließende Projektion der Szene in Schritt 4.) aus dem NPC auf den Bildschirm wird als *Device Mapping* bezeichnet. Der Einheitswürfel enthält dank der vorangegangenen Transformationen die gesamte darzustellende Szeneninformation. Die Abbildung muß lediglich die x- und y-Koordinaten aus dem NPC so in die Bildschirmkoordinaten DC (*Device Coordinate System*) transformieren, daß eine anschließende Rundung die ganzzahligen Koordinaten der Pixel ergibt.

DC ist auf den meisten Bildschirmen ein linkshändiges Koordinatensystem. (Die y-Achse zeigt nach unten, oben links ist der Ursprung $(0, 0)$.) Die Anzahl der Pixel im Ausgabefenster ist flexibel und betrage horizontal $xsize$ und vertikal $ysize$. In x-Richtung muß dann das Intervall $[0, 1]$ auf die diskreten Werte $\{0, ..., xsize \Leftrightarrow 1\}$ und in y-Richtung $[0, 1]$ umgekehrt auf $\{ysize \Leftrightarrow 1, ..., 0\}$ abgebildet werden. Die z-Koordinaten dienen später zur Bestimmung und Unterdrückung verdeckter Flächen, die sich durch die Staffelung der Objekte in der Tiefe des Bildraumes ergeben (siehe z-Buffer-Algorithmus).

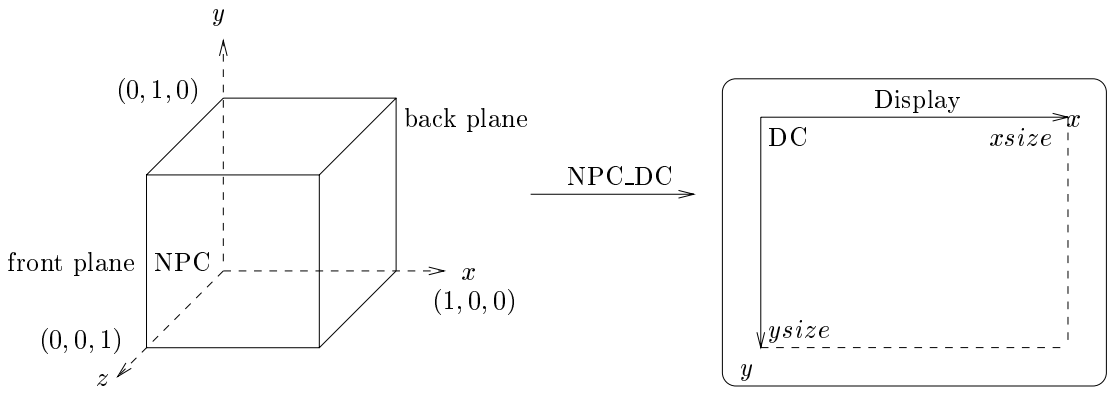


Abbildung 15.9: Device Mapping

Die Transformationsmatrix entspricht einer Skalierung um den Vektor $(xsize, \Leftrightarrow ysize, 1)$ konkateniert mit einer Translation des Ursprungs in die linke untere Ecke des Bildschirms $(0, ysize, 0)$.

$$T_{\text{NPC_DC}} = \begin{bmatrix} xsize & 0 & 0 & 0 \\ 0 & \Leftrightarrow ysize & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & ysize & 0 & 1 \end{bmatrix}$$

Für die Projektion werden einfach die x- und y-Werte übernommen; die z-Werte sind erforderlich zur Regelung der Sichtbarkeit beim Tiefenpufferalgorithmus.

15.2.6 Zusammenfassung

Abbildung 15.10 zeigt die schematische Transformationspipeline im Überblick. Diese wird bei der Ausführung des Programms von Polygonen durchlaufen, aus denen sich die Elementarobjekte zusammensetzen.

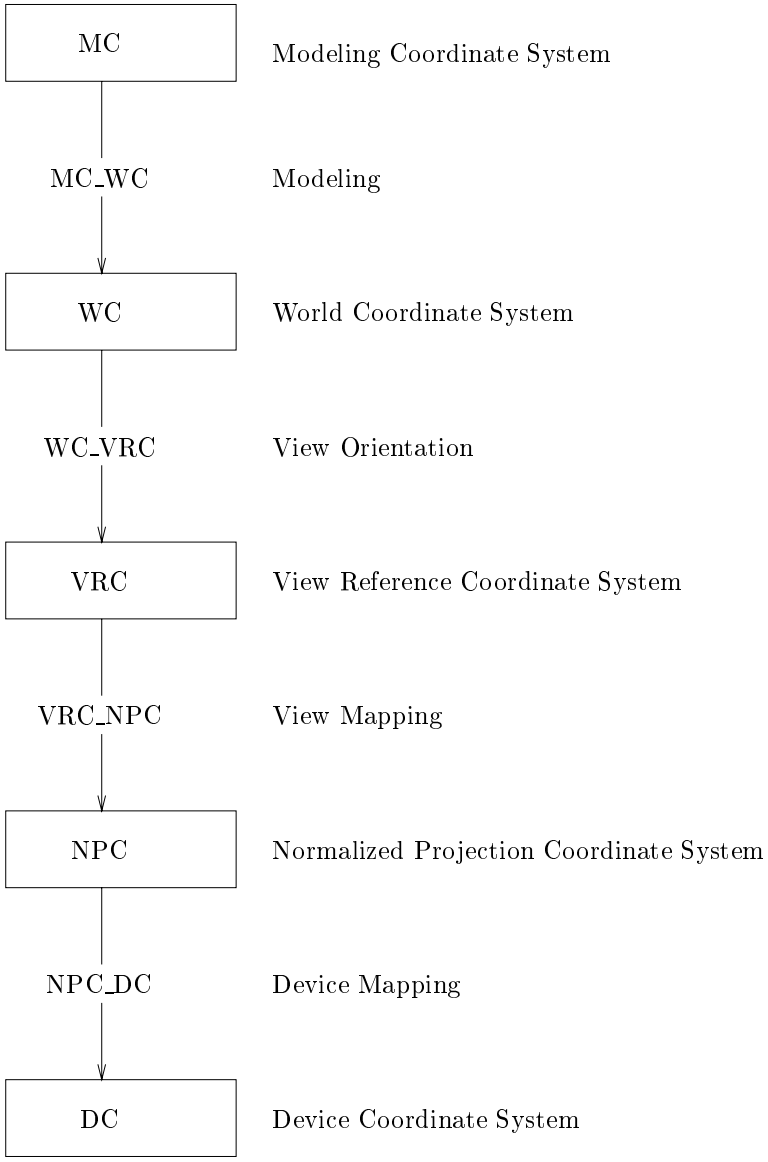


Abbildung 15.10: Transformationspipeline

Kapitel 16

3D-Objekte

Im Flächenmodell werden Objekte durch ihre Begrenzungsflächen beschrieben und diese wiederum durch ihre Kanten mit den Eckpunkten im Modellkoordinatensystem. Außerdem wird noch ein Normalenvektor im MC benötigt. Auf diese Weise kann ein Programm Polyeder exakt darstellen und gekrümmte Oberflächen durch konvexe Polygone beliebig gut approximieren.

16.1 Polyeder

Abbildung 16.1 zeigt eine vom Projektionsalgorithmus erzeugte Szene mit Würfel und Tetraeder in der Drahtmodell-Darstellung mit gestrichelten Rückkanten.

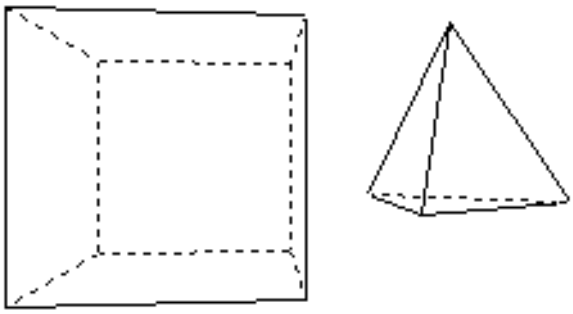


Abbildung 16.1: Vom Projektionsalgorithmus berechnete Drahtgitter-Darstellung

Am Beispiel des Würfels soll die Repräsentation von Polyedern im Halbkantenmodell erläutert werden.

16.1.1 Würfel

Der *Würfel* mit der Kantenlänge 1 hat im MC die homogenen Eckpunktkoordinaten

$$\begin{array}{llll} (+0.5, & +0.5, & +0.5, & 1), \\ (+0.5, & \Leftrightarrow 0.5, & +0.5, & 1), \\ (\Leftrightarrow 0.5, & +0.5, & +0.5, & 1), \\ (\Leftrightarrow 0.5, & \Leftrightarrow 0.5, & +0.5, & 1), \end{array} \qquad \begin{array}{llll} (+0.5, & +0.5, & \Leftrightarrow 0.5, & 1), \\ (+0.5, & \Leftrightarrow 0.5, & \Leftrightarrow 0.5, & 1), \\ (\Leftrightarrow 0.5, & +0.5, & \Leftrightarrow 0.5, & 1), \\ (\Leftrightarrow 0.5, & \Leftrightarrow 0.5, & \Leftrightarrow 0.5, & 1). \end{array}$$

Da jede der sechs Flächen des Würfels eben ist, gibt es pro Fläche nur einen Normalenvektor, der durch das Kreuzprodukt zweier benachbarter Kanten berechnet werden kann. Die Reihenfolge der Punkte ist dabei signifikant, da jeweils aufeinanderfolgende Punkte eine Kante bilden. Ein *Quader* entsteht aus dem Würfel durch ungleichmäßige Skalierung beim Modeling.

Auch alle anderen Polyeder werden im MC in der Halbkantendarstellung definiert, ein Tetraeder beispielsweise mit Kantenlänge 1 und Schwerpunkt im Ursprung.

16.2 Gekrümmte Flächen

Eine gekrümmte Oberfläche, die analytisch beschrieben werden kann, läßt sich beliebig gut durch Polygone approximieren. Die Gruppe der Objekte mit analytisch gekrümmter Oberfläche besteht aus zwei Typen: Während z.B. die Kugel nur aus einer solchen Fläche besteht, sind Zylinder und Kegel aus mehreren Flächen zusammengesetzt.

Die Repräsentation der gekrümmten Körper erfolgt im Flächenmodell mit einer Flächen- und Eckpunkteliste sowie einer Normalenliste, die pro Eckpunkt die zugehörige Normale enthält.

16.2.1 Zylinder

Der zur z-Achse symmetrische *Zylinder* mit Höhe 2 und Radius 1 besteht aus einer Mantelfläche und zwei Deckflächen. Die beiden Kreisscheiben bei $z = 1$ und $z = \Leftrightarrow 1$ lassen sich durch regelmäßige n-Ecke darstellen. Die Mantelfläche wird durch n Rechtecke approximiert. Dabei wachsen mit n sowohl die Genauigkeit der Approximation als auch der Rechenaufwand. Sei $\alpha = (2\pi)/n$, so lauten für ein solches Rechteck die vier Eckpunkte und die zugehörigen Normalenvektoren:

Eckpunkt	Normalenvektor
$(\cos(\phi), \sin(\phi), +1, 1)$	$(\cos(\phi), \sin(\phi), 0, 0)$
$(\cos(\phi+\alpha), \sin(\phi+\alpha), +1, 1)$	$(\cos(\phi+\alpha), \sin(\phi+\alpha), 0, 0)$
$(\cos(\phi+\alpha), \sin(\phi+\alpha), \Leftrightarrow 1, 1)$	$(\cos(\phi+\alpha), \sin(\phi+\alpha), 0, 0)$
$(\cos(\phi), \sin(\phi), \Leftrightarrow 1, 1)$	$(\cos(\phi), \sin(\phi), 0, 0)$

mit $\phi = k \cdot \alpha$, $k \in \{0, ..., n \Leftrightarrow 1\}$.

Als Normalenvektor in einem Eckpunkt dieser Fläche wird also der tatsächliche Normalenvektor der Mantelfläche genommen. Dadurch erhalten aneinandergrenzende Flächen in ihren gemeinsamen Eckpunkten auch denselben Normalenvektor. Auf diese Weise erzeugt das Programm bei der späteren Beleuchtung den Eindruck eines stetigen Übergangs zwischen den Flächen. Der Betrachter nimmt statt einer n-eckigen Säule den Zylinder wahr.

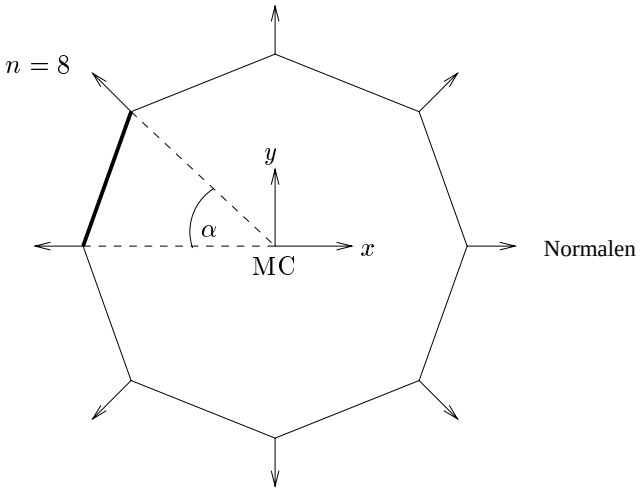


Abbildung 16.2: Zylinder im Grundriß mit Normalenvektoren der Mantelfläche

16.2.2 Kugel

Die Oberfläche einer *Kugel* mit Radius 1 kann beschrieben werden durch

$$(\sin(\theta)\cos(\phi), \sin(\theta)\sin(\phi), \cos(\theta)), 0 \leq \phi < 2\pi, 0 \leq \theta < \pi \ .$$

Zur Approximation durch Flächen wird der Vollwinkel in n Teile zerlegt:

$$\alpha = (2\pi)/n, \quad n \in \mathbb{N} \text{ gerade} \ .$$

Dadurch entstehen auf der Kugel $n/2$ gleichgroße Längenkreise und $(n/2 \Leftrightarrow 1)$ verschieden große Breitenkreise. Diese schneiden n Dreiecke an jedem Pol und $n(n/2 \Leftrightarrow 2)$ Vierecke aus der Kugeloberfläche. Die Ortsvektoren der Eckpunkte eines Dreiecks am Nordpol ($\theta = 0$) lauten

$$(0, 0, +1, 1),$$

$$(\sin(\alpha)\cos(\phi), \sin(\alpha)\sin(\phi), \cos(\alpha), 1),$$

$$(\sin(\alpha)\cos(\phi + \alpha), \sin(\alpha)\sin(\phi + \alpha), \cos(\alpha), 1),$$

mit $\phi = k \cdot \alpha$, $k \in \{0, ..., n \Leftrightarrow 1\}$.

Die Eckpunkte eines der Vierecke haben die Ortsvektoren

$$(\sin(\theta)\cos(\phi), \sin(\theta)\sin(\phi), \cos(\theta), 1),$$

$$(\sin(\theta)\cos(\phi + \alpha), \sin(\theta)\sin(\phi + \alpha), \cos(\theta), 1),$$

$$(\sin(\theta + \alpha)\cos(\phi + \alpha), \sin(\theta + \alpha)\sin(\phi + \alpha), \cos(\theta + \alpha), 1),$$

$$(\sin(\theta + \alpha)\cos(\phi), \sin(\theta + \alpha)\sin(\phi), \cos(\theta + \alpha), 1),$$

mit $\phi = k \cdot \alpha, k \in \mathbb{N}, k < n$ und $\theta = l \cdot \alpha, l \in \mathbb{N}, 0 < l < (n/2 \Leftrightarrow 1)$.

Als Normalenvektor wird in jedem Eckpunkt der Ortsvektor als Richtungsvektor ($w = 0$) eingetragen, denn der Radiusvektor steht senkrecht auf der Kugeloberfläche. Einen *Ellipsoid* erzeugt der Rendering-Algorithmus aus der Kugel durch ungleichmäßige Skalierung beim Modeling.

Abbildung 16.3 zeigt eine vom Projektionsalgorithmus erzeugte Szene mit Kugel und Zylinder in der Drahtmodell-Darstellung ohne Rückkanten ($n = 32$).

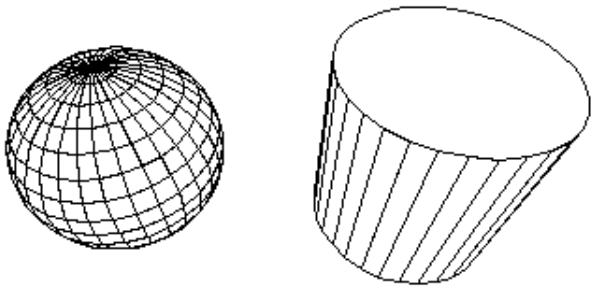


Abbildung 16.3: Kugel und Zylinder als Drahtgitter