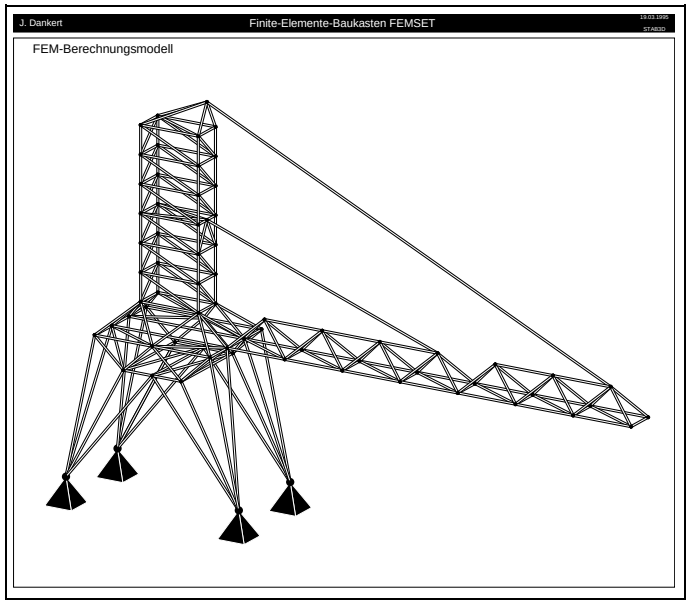
Vorlesung

INFORMATIK 2

Inhalt

1 Graphische Darstellung dreidimensionaler Objekte

Die Beschreibung der Form dreidimensionaler Objekte erfolgt mit Informationen über die **Geometrie** und die **Topologie**. Die Informationen zur Geometrie beschreiben die Lage von Punkten (z. B. durch die Angabe der Koordinaten der Punkte bezüglich eines geeigneten Koordinatensystems oder eine mathematische Beziehung, mit der die Punktkoordinaten berechnet werden können) oder die Abmessungen von Linien, Flächen und Körpern, die Informationen über die Topologie geben Auskunft, wie aus den geometrischen Informationen Linien, Flächen und Körper gebildet werden.



Für die Beschreibung des skizzierten Krans als Raum-Fachwerk genügen zwei Listen:

- Die Geometrie-Information kann als Koordinaten-Liste aller Fachwerksknoten (drei Koordinaten pro Knoten, bezogen auf ein beliebiges Koordinatensystem) gespeichert werden. Damit wäre ein "Knoten-Bild" darstellbar.
- Die Lage der Stäbe kann dann als rein topologische Information (z. B. in einer sogenannten "Koinzidenz-Liste") gespeichert werden: Für jeden Stab wird ein Wertepaar angegeben, mit dem auf die Knotennummern verwiesen wird, an denen der jeweilige Stab angeschlossen ist. Diese Liste "pointert" also auf die Geometrie-Information.

Mit diesen Informationen kann eine Prinzip-Skizze des Fachwerks angefertigt werden, selbst die Information, welche Stäbe bei einer bestimmten Ansicht von anderen verdeckt werden, ist enthalten (und wurde von dem Programm, das das Bild erzeugt hat, auch ausgewertet).

Abhängig von den Problemen, die mit einem rechnerintern gespeicherten Modell behandelt werden sollen, müssen weitere Informationen gespeichert werden. Die meisten Informationen "pointern" auf die Geometrie- und Topologie-Informationen oder werden als "Attribute" an an andere Informationen geknüpft. So können z. B. für ein Fachwerk-Berechnungsmodell die Informationen über die gelagerten bzw. belasteten Knoten in Listen zusammengestellt werden, die auf die Knotennummern verweisen, an die Stäbe können alle erforderlichen Attribute (z. B.: Querschnittsfläche, Materialeigenschaften, Oberflächenbeschaffenheit, Farbe ...) geknüpft werden. Die jeweils günstigste Art der rechnerinternen Speicherung der Informationen (verkettete Listen, Matrizen, ...) ist von dem Verwendungszweck abhängig (Berechnungsmodell, CAD-Modell, ...), folgt aber weitgehend einheitlichen Prinzipien.

In diesem Kapitel werden einige Probleme der graphischen Darstellung dreidimensionaler Objekte behandelt (perspektivische Ansichten, Transformationen, Sichtbarkeitstests). Da die

Geometrie-Informationen sich fast immer auf die Informationen über die Lage (und das "Schicksal" bei Bewegungen) einzelner Punkte reduzieren lassen, werden zunächst die Beschreibung und die Abbildung eines Punktes behandelt.

1.1 Homogene Koordinaten

Speziell für die Probleme der projektiven Geometrie wird mit erheblichem Vorteil die Darstellung eines Punktes im Raum durch vier Angaben (an Stelle der drei kartesischen Koordinaten) beschrieben. Diese sogenannten **homogenen Koordiaten** stehen mit den kartesischen Koordinaten in einem einfachen Zusammenhang:

$$\text{Homogene Koordinaten} \quad \rightarrow \quad \begin{bmatrix} x \\ y \\ z \\ \lambda \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} x/\lambda \\ y/\lambda \\ z/\lambda \end{bmatrix} \quad \leftarrow \quad \text{Kartesische Koordinaten}$$

Dabei kann λ einen beliebigen Wert ungleich Null haben, für den Spezialfall $\lambda = \mathbf{1}$ sind die ersten drei Komponenten in der Darstellung eines Punktes mit homogenen Koordinaten mit den kartesischen Koordinaten identisch.

Der Vorteil, der sich aus der Verwendung homogener Koordinaten ergibt, wird sich in der Möglichkeit der einheitlichen Darstellung der unterschiedlicher Transformationen zeigen, wodurch sich nacheinander auszuführende Transformationen mathematisch sehr übersichtlich verknüpfen lassen. Bei den in den folgenden Abschnitten behandelten Projektionen von Raumpunkten in eine Darstellungsebene ergeben sich die Abbildungen automatisch in (ebenen) homogenen Koordinaten.

Folgende Vorstellung kann mit der sicher zunächst etwas ungewöhnlichen Beschreibung eines Punktes durch vier Angaben verknüpft werden: Der darzustellende Punkt definiert gemeinsam mit dem Nullpunkt des Koordinatensystems eine Gerade. Wenn λ die Werte von $-\infty$ bis $+\infty$ durchläuft, dann werden alle Punkte auf dieser Geraden beschrieben (und im Vorgriff auf die Überlegungen in den folgenden Abschnitten: Wenn diese Gerade die "Blickrichtung" definiert, werden alle Punkte der Geraden auf den gleichen Punkt der "Projektionsebene" abgebildet).

1.2 Transformationen

Zwei verschiedene Transformationen werden betrachtet:

- Wenn sich ein Körper bezüglich eines **festen Koordinatensystems** bewegt, dann ändern sich die Koordinaten der Körperpunkte nach den Regeln der **geometrischen Transformation**.
- Wenn das Koordinatensystem selbst bewegt wird, dann ändern sich die Koordinaten der Punkte eines sich nicht bewegenden Körpers nach den Regeln der **Koordinaten-transformation**.

Es genügt, das "Schicksal" eines Punktes bei einer Transformation zu untersuchen. In allen Fällen wird im folgenden die "alte" Lage des Punktes durch die Koordinaten x, y und z beschrieben, die Lage nach der Transformation durch die Koordinaten x', y' und z'.

Es werden folgende Transformationen beschrieben:

- Bei einer **Translation** bewegen sich alle Punkte des Körpers (geometrische Transformation) bzw. des Koordinatensystems (Koordinatentransformation) auf kongruenten Bahnen. Bei der **geometrischen Translation** mögen die Koordinaten aller Punkte des Körpers die Veränderungen t_x, t_y und t_z erfahren, so daß die neue Lage eines Punktes durch

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

  beschrieben wird.

- Bei der **Skalierung** werden die Abmessungen des Körpers (geometrische Transformation) bzw. die Skaleneinteilung der Koordinatenachsen (Koordinatentransformation) vergrößert (Skalierungsfaktoren größer als 1) bzw. verkleinert. Die Skalierung bezieht sich immer auf einen zu definierenden Punkt, der dann selbst seine Lage nicht verändert, während alle anderen Punkte ihren Abstand vom Bezugspunkt vergrößern oder verkleinern. Bei der **geometrischen Skalierung bezüglich des Nullpunktes** mit den Skalierungsfaktoren s_x, s_y und s_z (jeweils in Richtung der Koordinatenachsen) berechnet sich die Lage des neuen Punktes nach

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} s_x x \\ s_y y \\ s_z z \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_z \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} .$$

- Bei einer **Rotation** dreht sich der Körper (geometrische Transformation) bzw. das Koordinatensystem (Koordinatentransformation) um einen bestimten Winkel **um eine vorzugebende Achse**, z. B. kann eine Koordinatenachse als Rotationsachse fungieren. Wie die Skalierung kann die Beziehung zwischen den Koordinaten eines Punktes vor bzw. nach der Drehung durch eine Transformationsmatrix beschrieben werden, mit der die "alten" Punktkoordinaten zu multiplizieren sind (Formeln finden sich im folgenden Abschnitt).

Da sich die Translation als einzige Transformation in kartesischen Koordinaten nicht durch eine Transformationsmatrix beschreiben läßt, kann hier erstmals mit Vorteil von den homogenen Koordinaten Gebrauch gemacht werden. Die kartesischen Koordinaten werden um $\lambda = 1$ ergänzt, und alle Transformationen lassen sich einheitlich durch Transformationsmatrizen beschreiben. Während für die Translation das Aufschreiben der Beziehung als Multiplikation "Matrix • Vektor" überhaupt erst möglich wird, werden die Transformationsmatrizen für die Skalierung und die Rotation um eine einfache Zeile bzw. Spalte ergänzt ("gerändert").

1.2.1 Geometrische Transformation mit homogenen Koordinaten

Translation um t_x, t_y und t_z:

$$\bar{x}' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \bar{T}_G \, \bar{x}$$

Skalierung bezüglich des Nullpunktes um s_x, s_y und s_z:

$$\bar{x}' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \bar{S}_G \, \bar{x}$$

Rotation um die x-Achse mit dem Winkel φ_x:

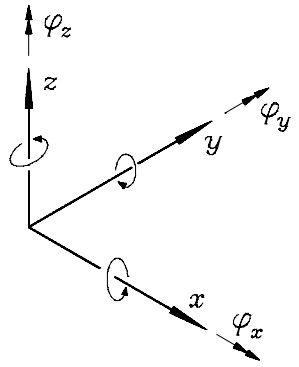
$$\bar{x}' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\varphi_x & -\sin\varphi_x & 0 \\ 0 & \sin\varphi_x & \cos\varphi_x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \bar{R}_{Gx} \, \bar{x}$$

Rotation um die y-Achse mit dem Winkel φ_y:

$$\bar{x}' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\varphi_y & 0 & \sin\varphi_y & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\varphi_y & 0 & \cos\varphi_y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \bar{R}_{Gy} \, \bar{x}$$

Rotation um die z-Achse mit dem Winkel φ_z:

$$\bar{x}' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\varphi_z & -\sin\varphi_z & 0 & 0 \\ \sin\varphi_z & \cos\varphi_z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \bar{R}_{Gz} \, \bar{x}$$



Definition positiver Drehwinkel

Mit diesen Formeln wird die **Bewegung eines Punktes im festen Koordinatensystem** beschrieben.

1.2.2 Koordinatentransformation mit homogenen Koordinaten

Translation um t_x, t_y und t_z:

$$\bar{x}' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & -t_x \\ 0 & 1 & 0 & -t_y \\ 0 & 0 & 1 & -t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \bar{T}_K \bar{x}$$

Skalierung bezüglich des Nullpunktes um s_x, s_y und s_z:

$$\bar{x}' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1/s_x & 0 & 0 & 0 \\ 0 & 1/s_y & 0 & 0 \\ 0 & 0 & 1/s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \bar{S}_K \bar{x}$$

Rotation um die x-Achse mit dem Winkel φ_x:

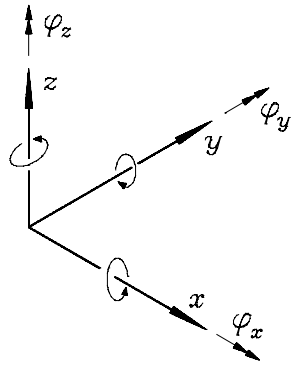
$$\bar{x}' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\varphi_x & \sin\varphi_x & 0 \\ 0 & -\sin\varphi_x & \cos\varphi_x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \bar{R}_{Kx} \bar{x}$$

Rotation um die y-Achse mit dem Winkel φ_y:

$$\bar{x}' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\varphi_y & 0 & -\sin\varphi_y & 0 \\ 0 & 1 & 0 & 0 \\ \sin\varphi_y & 0 & \cos\varphi_y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \bar{R}_{Ky} \bar{x}$$

Rotation um die z-Achse mit dem Winkel φ_z:

$$\bar{x}' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\varphi_z & \sin\varphi_z & 0 & 0 \\ -\sin\varphi_z & \cos\varphi_z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \bar{R}_{Kz} \bar{x}$$



Definition positiver Drehwinkel

Mit diesen Formeln werden die Veränderungen der Koordinaten eines festen Punktes bei **Bewegung des Koordinatensystem** beschrieben.

Die Koordinatentransformationen sind zu den entsprechenden geometrischen Transformationen invers: Wenn im Anschluß an eine geometrische Transformation sofort die entsprechende Koordinatentransformation mit den gleichen Transformationsparametern ausgeführt wird, ist der ursprüngliche Zustand wiederhergestellt.

Dieser Tatbestand läßt sich mathematisch folgendermaßen formulieren:

$$\bar{T}_K = \bar{T}_G^{-1} \quad ; \qquad \bar{S}_K = \bar{S}_G^{-1} \quad ;$$

$$\bar{R}_{Kx} = \bar{R}_{Gx}^{-1} \quad ; \qquad \bar{R}_{Ky} = \bar{R}_{Gy}^{-1} \quad ; \qquad \bar{R}_{Kz} = \bar{R}_{Gz}^{-1} \quad .$$

Da die Transformationsmatrizen, die die Rotationen definieren, sogenannte "Orthonormalmatrizen" sind (nur zueinander orthogonale normierte Zeilen bzw. Spalten haben), kann der Zusammenhang für diese Matrizen noch einfacher formuliert werden:

$$\bar{R}_{Kx} = \bar{R}_{Gx}^{T} \quad ; \qquad \bar{R}_{Ky} = \bar{R}_{Gy}^{T} \quad ; \qquad \bar{R}_{Kz} = \bar{R}_{Gz}^{T} \quad .$$

Man beachte, daß das Ergebnis mehrerer nacheinander ausgeführter Transformationen im allgemeinen bei unterschiedlicher Reihenfolge ansonsten gleichartiger Transformationen zu unterschiedlichen Ergebnissen führt.

Soll das Ergebnis mehrerer Transformationen rückgängig gemacht werden, so müssen die inversen Transformationen in exakt umgekehrter Reihenfolge ausgeführt werden.

Mit Hilfe der angegebenen Formeln für die elementaren Transformationen lassen sich fast alle benötigten allgemeinen Transformationen zusammensetzen, z. B. kann die Skalierung eines Körpers bezüglich eines beliebigen Punktes (x_0, y_0, z_0) in drei Schritten ausgeführt werden:

- Verschiebung des Koordinatensystems in den Punkt (x_0, y_0, z_0) mit der Translationsmatrix der Koordinatentransformation,
- Skalierung (geometrische Transformation) bezüglich des Nullpunktes,
- mit der inversen Verschiebungstransformation zum ersten Schritt (geometrische Translation) wird die Lage des Körpers (zumindest des nicht geänderten Bezugspunktes der Skalierung) bezüglich des Koordinatensystems wiederhergestellt.

In der Formel, die die gesamte Transformation beschreibt, ist die Reihenfolge der Transformationsmatrizen wichtig. Für die zuletzt ausgeführte Transformation steht die Transformationsmatrix ganz links:

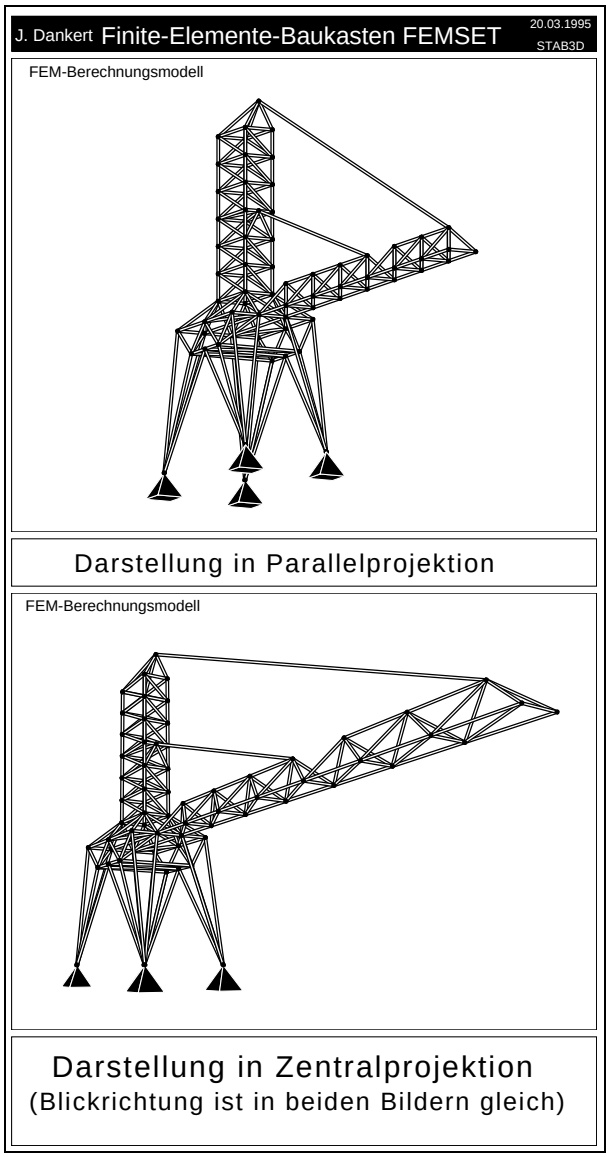
$$\bar{x}' = \bar{T}_G \, \bar{S}_G \, \bar{T}_K \, \bar{x} \quad .$$

Auf entsprechende Weise sind z. B. Rotations-Transformationen um beliebige Achsen realisierbar.

1.3 Projektionen

Das grundsätzliche Problem, dreidimensionale Objekte auf eine zweidimensionale Zeichenfläche abzubilden, wird durch **Projektion** der dreidimensionalen Punkte auf eine Ebene nach fest zu definierenden Regeln gelöst:

- Bei der **Zentralprojektion** wird von einem **Projektionszentrum ("Eye Point")** zu jedem Punkt des Körpers ein **Sehstrahl** gezogen. Der Schnittpunkt des Sehstrahls mit der Ebene, auf der die zweidimensionale Abbildung entstehen soll **(Projektionsebene)**, ist die Abbildung des 3D-Punktes in der Zeichenebene.
- Die **Parallelprojektion** kann als Sonderfall der Zentralprojektion angesehen werden, bei der das Projektionszentrum im Unendlichen liegt, so daß alle Sehstrahlen parallel verlaufen.



Während die Zentralprojektion (besonders mit nicht zu großen Entfernungen des Projektionszentrums vom Objekt, vgl. nebenstehendes Bild, "Eye Point" etwa in realer Augenhöhe) den räumlichen Eindruck besonders gut vermittelt, hat die Parallelprojektion unter anderem die angenehme Eigenschaft, vertikale Linien in der Projektionsebene auch vertikal darzustellen. Die für technische Zeichnungen üblichen Darstellungen (Vorderansicht, Seitenansicht, Draufsicht) sind Sonderfälle der Parallelprojektion.

1.3.1 Definition der Zentralprojektion

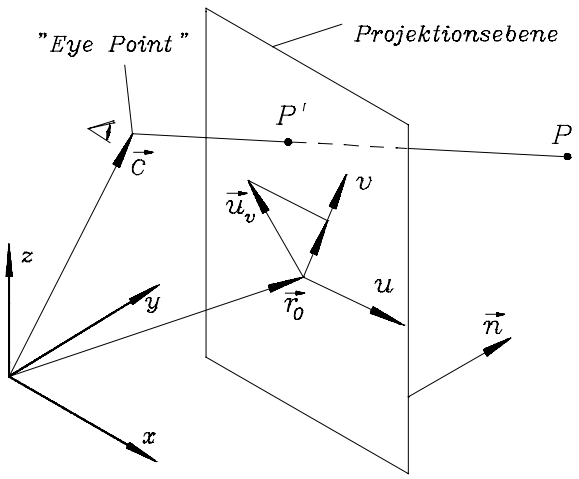
Eine Zentralprojektion wird definiert durch

- den durch einen Vektor $\vec{c}$ beschriebenen **"Eye Point"** (Projektionszentrum),
- die **Projektionsebene**, die durch einen **Referenzpunkt** (Vektor $\vec{r}_0$) und einen **Normalenvektor** $\vec{n}$ beschrieben wird, und
- einen sogenannten **"Up Vector"** $\vec{u}_v$ zur Definition des ebenen u-v-Koordinatensystems der Bildebene.

Die Vektoren $\vec{c}$, $\vec{r}_0$, $\vec{n}$, $\vec{u}_v$ werden in einem dreidimensionalen raumfesten ***x*-*y*-*z***-Koordinatensystem beschrieben, für den Vektor $\vec{n}$ wird vereinbart, daß er zu der Seite der Projektionsebene zeigt, auf der der "Eye Point" nicht liegt.

Das zweidimensionale Koordinatensystem der Bildebene definiert sich wie folgt:

- Der Ursprung fällt mit dem Referenzpunkt zusammen.
- Die Richtung der ***v***-Achse wird durch die Projektion des "Up Vectors" auf die Projektionsebene festgelegt.



Allgemeine Definition der Zentralprojektion

- Die Richtung der ***u***-Achse wird so festgelegt, daß ***u***, ***v*** und $\vec{n}$ in dieser Reihenfolge ein Linkssystem definieren ("Eye Point" befindet sich vor dem Bildschirm, $\vec{n}$ zeigt in den Bildschirm hinein und das ***u*-*v***-Koordinatensystem liegt in der Bildschirmebene).

Ein beliebiger Körperpunkt ***P***, der im dreidimensionalen ***x*-*y*-*z***-Koordinatensystem durch einen (in der Skizze nicht gezeichneten) Vektor $\vec{p}$ beschrieben wird und sowohl vor oder hinter und auch in der Projektionsebene liegen darf, wird auf die Projektionsebene abgebildet, indem der Schnittpunkt ***P'*** (nachfolgend durch den Vektor $\vec{p}'$ beschrieben) berechnet wird, den der **Sehstrahl**, der vom "Eye Point" zum Punkt ***P*** gezogen wird, mit der Projektionsebene hat.

Da die Koordinaten des Punktes ***P'*** für den Zeichenvorgang im ebenen ***u*-*v***-Koordinatensystem benötigt werden, werden zunächst zwei Vektoren $\vec{u}$ und $\vec{v}$ in Richtung der ***u***- bzw. ***v***-Achse bestimmt, die dann zu den Einheitsvektoren $\vec{u}_e$ und $\vec{v}_e$ des Bildebenen-Koordinatensystems normiert werden:

Der Normalenvektor $\vec{n}$ wird durch Division durch seinen Betrag zum Normalen-Einheitsvektor

$$\vec{n}_e = \frac{\vec{n}}{|\vec{n}|} \quad . \tag{1.1}$$

Der "Up Vector" $\vec{u}_v$ kann als Summe des Vektors $\vec{v}$ und eines Vektors $k\,\vec{n}_e$ (mit zunächst noch unbestimmten Faktor ***k***) aufgeschrieben werden:

$$\vec{u}_v = \vec{v} + k\,\vec{n}_e \quad . \tag{1.2}$$

Multiplikation dieser Beziehung mit $\vec{n}_e$ führt wegen $\vec{n}_e \cdot \vec{v} = 0$ (Vektoren stehen senkrecht aufeinander) und mit $\vec{n}_e \cdot \vec{n}_e = 1$ auf

$$k = \vec{n}_e \cdot \vec{u}_v \quad , \tag{1.3}$$

was in die Beziehung (1.2) eingesetzt werden kann. Aus dem daraus berechneten Vektor

$$\vec{v} = \vec{u}_v - (\vec{n}_e \cdot \vec{v}_u) \cdot \vec{n}_e \tag{1.4}$$

entsteht schließlich der Einheitsvektor in Richtung der ***v***-Achse:

$$\vec{v}_e = \frac{\vec{v}}{|\vec{v}|} \quad . \tag{1.5}$$

Der Einheitsvektor in Richtung der ***u***-Achse kann nun einfach aus dem Vektorprodukt

$$\vec{u}_e = \vec{n}_e \times \vec{v}_e \tag{1.6}$$

berechnet werden.

Mit den (gesuchten) Koordinaten ***u*** und ***v*** des Punktes ***P′*** kann der Vektor $\vec{p}'$ zu diesem Punkt formal aufgeschrieben werden als

$$\vec{p}' = \vec{r}_0 + u\,\vec{u}_e + v\,\vec{v}_e \quad . \tag{1.7}$$

Da der Endpunkt des Vektors $\vec{c}$ ("Eye Point") und die Punkte ***P′*** und ***P*** auf einer Geraden liegen (Sehstrahl), können sich die beiden Differenzvektoren $(\vec{p}' - \vec{c})$ und $(\vec{p} - \vec{c})$ nur um einen (zunächst ebenfalls noch unbekannten) Faktor unterscheiden:

$$(\vec{p}' - \vec{c})\,\lambda = (\vec{p} - \vec{c}) \quad . \tag{1.8}$$

Aus den Beziehungen (1.7) und (1.8) wird der Vektor $\vec{p}'$ eliminiert (die Koordinaten des Punktes ***P′*** im dreidimensionalen Koordinatensystem sind ohnehin nicht interessant), und es verbleibt mit

$$(\vec{r}_0 + u\,\vec{u}_e + v\,\vec{v}_e - \vec{c})\,\lambda = \vec{p} - \vec{c} \tag{1.9}$$

eine Vektorgleichung für die drei Unbekannten ***u***, ***v*** und λ. Die Vektoren werden durch ihre Komponenten dargestellt:

$$\vec{p} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad , \quad \vec{r}_0 = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} \quad , \quad \vec{c} = \begin{bmatrix} c_x \\ c_y \\ c_z \end{bmatrix} \quad , \quad \vec{u}_e = \begin{bmatrix} u_{ex} \\ u_{ey} \\ u_{ez} \end{bmatrix} \quad , \quad \vec{v}_e = \begin{bmatrix} v_{ex} \\ v_{ey} \\ v_{ez} \end{bmatrix} \quad , \tag{1.10}$$

für die Produkte der unbekannten Koordinaten ***u*** und ***v*** mit λ werden neue Unbekannte eingeführt:

$$\bar{u} = u\,\lambda \quad , \quad \bar{v} = v\,\lambda \quad . \tag{1.11}$$

Dann kann Gleichung (1.9) als lineares Gleichungssystem formuliert werden. Die Lösung von

$$\underbrace{\begin{bmatrix} u_{ex} & v_{ex} & x_0 - c_x \\ u_{ey} & v_{ey} & y_0 - c_y \\ u_{ez} & v_{ez} & z_0 - c_z \end{bmatrix}}_{\bar{P}} \underbrace{\begin{bmatrix} \bar{u} \\ \bar{v} \\ \lambda \end{bmatrix}}_{\vec{b}} = \begin{bmatrix} x - c_x \\ y - c_y \\ z - c_z \end{bmatrix}}_{\vec{p} - \vec{c}} \quad , \tag{1.12}$$

liefert den Vektor der (ebenen) homogenen Koordinaten des Punktes ***P′*** im Koordinatensystem der Bildebene:

$$\vec{b} = \begin{bmatrix} \bar{u} \\ \bar{v} \\ \lambda \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \bar{u}/\lambda \\ \bar{v}/\lambda \end{bmatrix} \quad . \tag{1.13}$$

Die Projektion versagt für $\lambda = \mathbf{0}$, was folgende Gründe haben kann:

- Der zu projizierende Punkt ***P*** ist identisch mit dem "Eye Point" **($\vec{p} = \vec{c}$)**, oder
- der Sehstrahl und der Normalenvektor der Projektionsebene stehen senkrecht aufeinander.

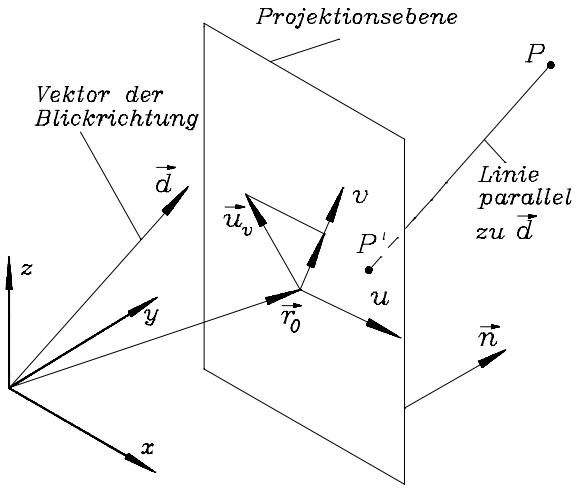
In beiden Fällen ist natürlich auch keine sinnvolle Projektion möglich. Formal liefert die Projektion auch Bildkoordinaten ***u*** und ***v***, wenn der Körperpunkt ***P*** hinter dem "Eye Point" liegt. Diese Fälle, die sich durch ein negatives λ äußern, sollten aussortiert werden.

1.3.2 Definition der Parallelprojektion

Eine Parallelprojektion wird definiert durch

- den "Vektor der Blickrichtung" $\vec{d}$,
- die **Projektionsebene**, die durch einen **Referenzpunkt** (Vektor $\vec{r}_0$) und einen **Normalenvektor** $\vec{n}$ beschrieben wird, und
- einen sogenannten **"Up Vector"** $\vec{u}_v$ zur Definition des ebenen ***u-v***-Koordinatensystems der Bildebene.

Die Vektoren $\vec{d}$, $\vec{r}_0$, $\vec{n}$, $\vec{u}_v$ werden in einem dreidimensionalen raumfesten ***x-y-z***-Koordinatensystem beschrieben, für den Vektor $\vec{n}$ wird vereinbart, daß er einen spitzen Winkel mit dem "Vektor der Blickrichtung" bildet.



Allgemeine Definition der Parallelprojektion

Die Definition des zweidimensionalen ***u-v***-Koordinatensystems der Bildebene erfolgt exakt nach der Vorschrift, die für die Zentralprojektion beschrieben wurde, so daß die Formeln (1.1) bis (1.7) unverändert gelten.

Ein beliebiger Körperpunkt ***P***, der im dreidimensionalen ***x-y-z***-Koordinatensystem durch einen (in der Skizze nicht gezeichneten) Vektor $\vec{p}$ beschrieben wird und sowohl vor oder hinter und auch in der Projektionsebene liegen darf, wird auf die Projektionsebene abgebildet, indem der Schnittpunkt ***P′*** (nachfolgend durch den Vektor $\vec{p}'$ beschrieben) berechnet wird, den **eine Parallele zum Vektor der Blickrichtung** durch ***P*** mit der Projektionsebene hat.

Der Vektor zum Punkt ***P*** kann also als Summe des Vektors zum Punkt ***P′*** und dem mit einem (zunächst unbekannten) Faktor multiplizierten "Vektor der Blickrichtung" $\vec{d}$ aufgeschrieben werden:

$$\vec{p} = \vec{p}' + \alpha \vec{d} \quad . \tag{1.14}$$

Da auch Formel (1.7) ihre Gültigkeit behält, kann aus (1.7) und (1.14) der Vektor $\vec{p}'$ (ähnlich zum Vorgehen bei der Zentralprojektion) eliminiert werden. Man erhält mit

$$\vec{r}_0 + u \vec{u}_e + v \vec{v}_e = \vec{p} - \alpha \vec{d} \tag{1.15}$$

eine Vektorgleichung für die drei Unbekannten ***u***, ***v*** und α. Die Vektoren werden durch ihre Komponenten dargestellt:

$$\vec{p} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \quad \vec{r}_0 = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix}, \quad \vec{d} = \begin{bmatrix} d_x \\ d_y \\ d_z \end{bmatrix}, \quad \vec{u}_e = \begin{bmatrix} u_{ex} \\ u_{ey} \\ u_{ez} \end{bmatrix}, \quad \vec{v}_e = \begin{bmatrix} v_{ex} \\ v_{ey} \\ v_{ez} \end{bmatrix}, \quad \textbf{(1.16)}$$

und Gleichung (1.15) kann als lineares Gleichungssystem formuliert werden. Die Lösung von

$$\begin{bmatrix} u_{ex} & v_{ex} & d_x \\ u_{ey} & v_{ey} & d_y \\ u_{ez} & v_{ez} & d_z \end{bmatrix} \begin{bmatrix} u \\ v \\ \alpha \end{bmatrix} = \begin{bmatrix} x - x_0 \\ y - y_0 \\ z - z_0 \end{bmatrix}, \qquad \textbf{(1.17)}$$

$$\bar{P}_p \qquad \vec{b}_p = \vec{p} - \vec{r}_0$$

liefert unmittelbar die Koordinaten ***u*** und ***v*** des Punktes ***P′*** im Koordinatensystem der Bildebene (und den nicht interessierenden Parameter α). Man beachte, daß die Lösung von (1.17) im Gegensatz zur Lösung von (1.12) keine homogenen Koordinaten liefert, sondern direkt die kartesischen Koordinaten des ebenen ***u***-***v***-Koordinatensystems.

1.4 Empfehlungen zur Definition von Projektionen

Die volle Allgemeinheit, mit der die Zentralprojektion und die Parallelprojektion in den Abschnitten 1.2 und 1.3 beschrieben wurden, sollte nicht dazu verführen, beliebige Kombinationen der jeweils vier Vektoren, die eine Projektion definieren, zu verwenden. Folgende Empfehlungen können allgemein gegeben werden:

- Bei der **Zentralprojektion** wird der Fußpunkt des Lotes vom "Eye Point" auf die Projektionsebene als "Hauptpunkt" bezeichnet. Es ist empfehlenswert, den Referenzpunkt der Projektionsebene (Vektor $\vec{r}_0$) mit dem Hauptpunkt zusammenfallen zu lassen. Da der Referenzpunkt der Ursprung des ebenen Koordinatensystems der Bildebene ist, steht die Projektionsebene dann senkrecht zum Sehstrahl auf den Nullpunkt des ***u***-***v***-Koordinatensystems.
- Bei der **Parallelprojektion** sollte die Projektionsebene senkrecht zum "Vektor der Blickrichtung" gelegt werden. Dann ist der Normalenvektor, der die Projektionsebene definiert, parallel zum "Vektor der Blickrichtung".
- Im allgemeinen ist ein "Up Vector", der parallel zur ***z***-Achse des dreidimensionalen Koordinatensystems liegt, eine gute Wahl (vermittelt den Eindruck, als würde der Beobachter auf einer zur ***x***-***y***-Ebene parallelen Ebene aufrecht stehen). Ein solcher "Up Vector" ist allerdings nicht möglich, wenn der "Eye Point" der Zentralprojektion selbst auf der ***z***-Achse liegt, oder der "Vektor der Blickrichtung" der Parallelprojektion parallel zu ***z***-Achse ist.

1.5 GI-Routinen für 3D-Graphik

Das "Graphics Interface" GI (Level 2 und 2E) unterstützt ab Version 2.0 auch 3D-Graphik (realisiert durch die "P-Routinen", die "T3-Routinen" und die "PT-Routinen", wobei die Bezeichnungen mit den Anfangsbuchstaben der Unterprogramme korrespondieren).

Den **"P-Routinen"** werden 3D-Punkte übergeben, die vor der Ausführung der Zeichenaktion mit der **"aktuellen Projektion"** in zweidimensionale Bildschirmkoordinaten überführt werden. Die 3D-Punkte beziehen sich auf ein beliebiges dreidimensionales Koordinatensystem (vom Benutzer zu definierende **"World Coordinates"**), die durch die Projektion zu 2D-"User Coordinates" werden. Die GI-Routinen unterstützen die **Zentralprojektion** und die **Parallelprojektion**.

Die **"T3-Routinen"** gestatten die **"Definition und Änderung einer räumlichen Transformation"**, speziell unterstützt werden **Translation, Rotation und Skalierung**. Es ist stets eine Transformationsmatrix gültig, mit der in allen Zeichenroutinen, deren Namen mit **PT** (**"PT-Routinen"**, Auswertung von Projektion **und** Transformation) beginnt, vor der Zeichenaktion die übergebenen "World Coordinates" transformiert werden, bevor sie dann mit der aktuellen Projektion in die Bildebenenkoordinaten umgerechnet und wie 2D-"User Coordinates" gezeichnet werden. Die 4*4-Transformationsmatrix bezieht sich auf homogene Koordinaten (XW,YW,ZW,1), was der Programmierer allerdings bei Nutzung der GI-Routinen nicht speziell beachten muß (Ausnahme: Routine T3MAM_GI, die nur für ganz spezielle Transformationen benötigt wird).

Die aktuelle Transformation wird als "Einheits-Transformation" initialisiert, so daß die "PT-Routinen" (z. B. **PTMOV_GI** und **PTLIN_GI** für das Bewegen des imaginären Zeichenstiftes bzw. das Zeichnen einer geraden Linie), zunächst so arbeiten wie die korrespondierenden "P-Routinen". Wenn allerdings die Transformationsmatrix durch die "T3-Routinen", die die Transformation beeinflussen, geändert wurde, dann werden alle den "PT-Routinen" übergebenen Koordinaten vor der Zeichenaktion geändert.

Die allgemeine Definition der Zentralprojektion, wie sie im Abschnitt 1.3.1 beschrieben wurde, wird in der GI-Toolbox verwaltet und kann bei Bedarf für spezielle Untersuchungen zugänglich gemacht werden. Allgemein zugänglich ist folgende sinnvolle Einschränkung zur

Definition einer Zentralprojektion bei der Arbeit mit der GI-Toolbox:

Es werden nur der **"Eye Point"** (Projektionszentrum) und ein Punkt der Projektionsebene (**"Referenzpunkt"**) im raumfesten ***x*-*y*-*z***-Koordinatensystem ("World Coordinates") definiert.

Die Projektionsebene wird von den GI-Routinen dann automatisch so gelegt, daß sie senkrecht zur Verbindungslinie vom "Eye Point" zum Referenzpunkt liegt (der Normalenvektor hat also die Richtung dieser Verbindungslinie, der Referenzpunkt ist gleichzeitig der sogenannte "Hauptpunkt" der Projektion).

Als "Up Vector" wird i. a. automatisch die ***z***-Achse gewählt (dies vermittelt den Eindruck, als würde der Beobachter auf einer zur ***x*-*y***-Ebene parallelen Ebene aufrecht stehen). Wenn der "Eye Point" selbst auf der ***z***-Achse liegt, wird entweder die positive ***y***-Achse ("Eye Point" liegt auf der positiven ***z***-Achse, "Draufsicht") oder die negative ***y***-Achse ("Eye Point" liegt auf der negativen ***z***-Achse) zum "Up Vector".

Auch die allgemeine Definition der Parallelprojektion, die im Abschnitt 1.3.2 beschrieben wurde, wird in der GI-Toolbox verwaltet und kann bei Bedarf für spezielle Untersuchungen zugänglich gemacht werden. Allgemein zugänglich ist folgende sinnvolle Einschränkung zur

Definition einer Parallelprojektion bei der Arbeit mit der GI-Toolbox:

Es werden nur der **"Vektor der Blickrichtung"** und ein Punkt der Projektionsebene (**"Referenzpunkt"**) im raumfesten ***x*-*y*-*z***-Koordinatensystem ("World Coordinates") definiert.

Die Projektionsebene wird von den GI-Routinen dann automatisch so gelegt, daß sie senkrecht zum Vektor der Blickrichtung liegt (der Normalenvektor hat also die gleiche Richtung wie der Vektor der Blickrichtung).

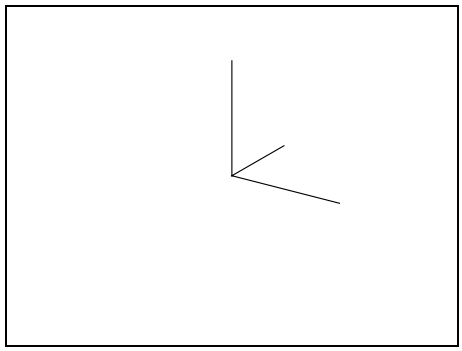
Als "Up Vector" wird i. a. automatisch die ***z***-Achse gewählt (dies vermittelt den Eindruck, als würde der Beobachter auf einer zur ***x*-*y***-Ebene parallelen Ebene aufrecht stehen). Wenn der "Vektor der Blickrichtung" parallel zur ***z***-Achse ist, wird entweder die positive ***y***-Achse ("Vektor der Blickrichtung" hat negative ***z***-Komponente, "Draufsicht") oder die negative ***y***-Achse ("Vektor der Blickrichtung" hat positive ***z***-Komponente) zum "Up Vector".

Für eine detaillierte Information über die Nutzung der GI-Toolbox wird auf die Dokumentation "GI-Toolbox 2.0" verwiesen (liegt im PC-Pool aus und kann vom Novell-Server als WordPerfect-File oder als PostScript-File kopiert werden).

1.6 Beispiel-Programme und Aufgaben

Auf dem Server des Novell-Netzes stehen zahlreiche Beispiel-Programme im Quell-Code zur Verfügung, die kopiert werden dürfen. Nachfolgend wird auf einige Programme aufmerksam gemacht, die die Verwendung der 3D-Graphik-Routinen aus der GI-Toolbox und die in den vorangegangenen Abschnitten beschriebenen theoretischen Grundlagen demonstrieren.

Das Programm **GRTEST70** definiert eine Zentralprojektion und zeichnet drei Linien in Richtung der drei Achsen des dreidimensionalen Koordinatensystems.



Ausgabe des Programms GRTEST70

Aufgabe 1.1: Das Programm **GRTEST70** ist zu analysieren und so zu modifizieren, daß die gleichen drei Linien (in anderer Farbe) in einer Parallelprojektion überlagert werden, die durch den gleichen Referenzpunkt wie die Zentralprojektion und einen zum "Eye Point"-Vektor negativen Vektor der Blickrichtung definiert wird.

Das Programm **GRTEST75** liest die Datei **STABWERK.DAT**, stellt das durch die Daten definierte Stabwerk dar und gestattet die Drehung des Stabwerks um die drei Koordinatenachsen. Da die darzustellende Struktur extern über Daten definiert wird, können mit diesem Programm auch beliebige andere Stabwerke dargestellt werden.

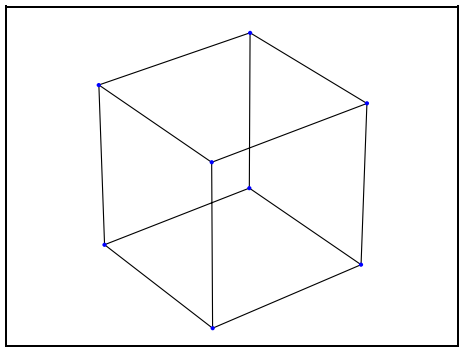
Ein Stabwerk läßt sich durch eine besonders einfache Datenstruktur darstellen. Das nebenstehend gezeigte Beispiel definiert eine Struktur mit 12 Stäben und 8 Knotenpunkten durch zwei Listen: Die erste Liste enthält alle Knotenkoordinaten, die zweite Liste (Koinzidenzliste) die Information, an welchen beiden Knoten jeder einzelne Stab verankert ist.

```
  12     8          Stäbe bzw. Knoten (Anzahl)
-2.5  -2.5   -2.5   Koordinaten des Knotens 1
 2.5  -2.5   -2.5   Koordinaten des Knotens 2
 2.5   2.5   -2.5   Koordinaten des Knotens 3
-2.5   2.5   -2.5   Koordinaten des Knotens 4
-2.5  -2.5    2.5   Koordinaten des Knotens 5
 2.5  -2.5    2.5   Koordinaten des Knotens 6
 2.5   2.5    2.5   Koordinaten des Knotens 7
-2.5   2.5    2.5   Koordinaten des Knotens 8
 1 2                 Knotennummern Stab  1
 2 3                 Knotennummern Stab  2
 3 4                 Knotennummern Stab  3
 4 1                 Knotennummern Stab  4
 5 6                 Knotennummern Stab  5
 6 7                 Knotennummern Stab  6
 7 8                 Knotennummern Stab  7
 8 5                 Knotennummern Stab  8
 1 5                 Knotennummern Stab  9
 2 6                 Knotennummern Stab 10
 3 7                 Knotennummern Stab 11
 4 8                 Knotennummern Stab 12
```

Datei STABWERK.DAT

Nach dem Programmstart wird das Stabwerk dargestellt. Die in der Datei STABWERK.DAT definierte Struktur erweist sich als würfelförmige Anordnung der 12 Stäbe.

Das dargestellte Objekt kann durch Drücken der Tasten ’X’, ’Y’ und ’Z’ gedreht werden. Es dreht sich jeweils um **10°** um die gewählte Achse. Da das vorherige Bild gelöscht wird, ensteht beim Niederhalten einer Taste der Eindruck eines sich drehenden Stabwerks.



Ausgabe des Programms GRTEST75

Aufgabe 1.2: Die Bildschirm-Darstellung des würfelförmigen Stabwerks läßt deutlich erkennen, daß die vertikalen Stäbe in der Zentralprojektion nicht vertikal dargestellt werden. Man zeige durch Modifikation des Programms, daß die Parallelprojektion die vertikalen Linien stets vertikal darstellt.

Aufgabe 1.3: Man definiere durch eine entsprechende Datei STABWERK.DAT ein Stabwerk mit 6 Knoten, die durch 12 oktaederförmig angeordnete Stäbe verbunden werden ("platonisches" Oktaeder, das von 8 gleichseitigen Dreiecken begrenzt wird).

Die Darstellungen der Stabwerke mit dem Programm GRTEST75 sind mit einem gravierenden Mangel behaftet: Sie enthalten keine Information darüber, welche Kanten vorn bzw. hinten liegen. Auch bei einfachen Gebilden kann diese Information vom Betrachter nicht ergänzt werden. Bei der oben gezeigten Darstellung des würfelförmigen Stabwerks kann man nicht entscheiden, ob es "von oben" oder "von unten" gesehen wird.

Da im Zusammenhang mit der eingestellten Projektion die Information über die Tiefenanordnung der Stäbe in den beiden Listen, die das Stabwerk definieren, aber enthalten ist, kann sie natürlich ausgewertet und zur Verbesserung der Darstellung genutzt werden. Bevor die dementsprechend arbeitenden Beispiel-Programme behandelt werden, wird im nächsten Abschnitt das "Überdeckungs-Problem" bei räumlichen Darstellungen allgemein diskutiert.

1.7 "Hidden Lines" und "Hidden Surfaces"

Die Darstellung eines dreidimensionalen Objektes auf zweidimensionalen Ausgabemedien stößt stets auf das Problem, das Informationsdefizit, das infolge der fehlenden Tiefenwirkung eigentlich nie zu vermeiden ist, in der Weise zu reduzieren, daß die Darstellung dem angestrebten Zweck gerecht wird. Während für möglichst realitätsnahe Ansichten die Sichtbarkeit der eigentlich verdeckten Kanten nicht tolerierbar ist, werden in technischen Zeichnungen verdeckte Kanten häufig bewußt eingezeichnet, weil eine möglichst komplette Information über das dargestellte Objekt wesentlich höher als die "Schönheit der Darstellung" bewertet werden muß.

Ob aber verdeckte Kanten ausgeblendet (realistische Darstellung) oder zum Beispiel gestrichelt dargestellt (technische Zeichnung) werden sollen, ändert am Problem nichts: Es muß ermittelt werden, welche Kanten dies sind.

Im folgenden wird vorausgesetzt, daß die Oberflächen der darzustellenden Körper nicht "durchsichtig" sind. Dann darf angenommen werden, daß nur die äußeren Flächen eines Körpers sichtbar sind.

Eine weitgehend fotorealistische Darstellung ist mit dem sogenannten **"Raytracing"** ("Strahlenverfolgung") möglich. Vom Projektionszentrum werden (für jeden Bildpunkt, der in der Projektionsebene dargestellt werden soll) Strahlen ausgesendet und verfolgt:

- Trifft ein Strahl auf keine Oberfläche der darzustellenden Objekte, so wird für den entsprechenden Bildpunkt die Hintergrundfarbe verwendet.
- Wenn Flächen der darzustellenden Körper vom Sehstrahl getroffen werden, müssen in der Regel die Schnittpunkte mit allen Flächen berechnet werden, um entscheiden zu können, welche Fläche dem Projektionszentrum am nächsten liegt. Nur der Schnittpunkt mit dieser Fläche muß weiter betrachtet werden.
- Wird eine nicht spiegelnde Fläche getroffen, erhält der Bildpunkt die Farbe des getroffenen Flächenpunktes. Bei spiegelnden Oberflächen (ähnlich müßte man bei durchscheinenden Körpern nach den Regeln der Lichtbrechung verfahren) muß der gespiegelte Strahl weiterverfolgt werden, um zusätzliche Farbanteile aus weiteren getroffenen Oberflächen (eventuell sogar von Körpern, die ansonsten außerhalb des Sichtbarkeitsbereiches liegen würden) zu ermitteln.
- Für eine fotorealistische Darstellung muß darüber hinaus noch der Einfluß der direkten Beleuchtung durch Lichtquellen, eventuell diffuse Hintergrundbeleuchtung, indirekte Beleuchtung durch Speigelung und Brechung auf die Farbe und die Helligkeit der Punkte untersucht werden (auch Schattenwurf, Glanzeffekte, ...).

Alle diese Faktoren, die das fotorealistische Bild beeinflussen, basieren auf eindeutigen physikalischen Gesetzen, und der Algorithmus zu ihrer Umsetzung ist durchaus beherrschbar. Der Rechenaufwand (auch für äußerst leistungsfähige Computer) ist jedoch enorm. Einige CAD-Systeme offerieren die Möglichkeit, weitgehend fotorealistische Bilder zu erzeugen (es ist schon faszinierend, wenn Werkstattzeichung und Glanzfoto für den Werbeprospekt aus der gleichen Datenbasis generiert werden können), in jedem Fall muß ein CAD-System jedoch eine schnellere (und natürlich der Konstruktionszeichnung angemessene) Möglichkeit bieten, wenigstens die unsichtbaren Flächen und Linien zu ermitteln (und nicht oder zum Beispiel gestrichelt darzustellen).

Da auch beim Verzeicht auf fotorealistische Darstellung (für Konstruktionszeichnungen ohnehin nicht erwünscht) die erforderliche Rechenzeit für das Ausblenden verdeckter Linien und Flächen erheblich sein kann, werden zur Erzielung eines akzeptablen Antwortverhaltens im Dialogbetrieb gelegentlich Kompromisse bei den Algorithmen in Kauf genommen, so daß in speziellen Fällen Fehler in der Darstellung nach dem Sichtbarkeitstest eher als unvertretbar hohe Rechenzeiten toleriert werden. Natürlich sollte immer das "Umschalten auf einen sauberen und aufwendigen Algorithmus" möglich sein.

Ein für die Bildschirmausgabe besonders schnelles Verfahren ist, **alle** Oberflächen des Körpers in der Reihenfolge einer "Prioritätenliste" zu zeichnen. Wenn diese Liste zum Beispiel durch die Entfernung der Flächen vom "Eye Point" bestimmt und die am weitesten entfernten Flächen zuerst gezeichnet werden, dann überzeichnen die Flächen mit kürzerer Entfernung automatisch die Flächen, die von ihnen ganz oder teilweise überdeckt werden. Dieses Verfahren bedarf am allgemeinen noch ein oder mehrerer Verfeinerungen:

- Da die "Entfernung" einer Fläche vom "Eye Point" sich natürlich immer nur auf einen (ziemlich willkürlich zu wählenden) Punkt der Fläche beziehen kann, sind bei großen und gekrümmten Flächen viele Fehler möglich (man denke daran, daß eine Kugel nur eine Fläche hat, welcher Punkt sollte für die Entfernungsbestimmung genommen werden). Abhilfe schafft man durch Einteilung aller Oberflächen in genügend kleine Teilflächen, die dann selbständig in der Prioritätenliste auftauchen.
- Ein wesentlicher Mangel des sehr schnellen Verfahrens ist, daß Körperkanten nicht automatisch sichtbar werden (man denke an einen Würfel, von dem am Ende zwar nur drei Flächen sichtbar wären, die sich jedoch nicht voneinander abgrenzen). Um diesem Mangel abzuhelfen, könnte man den Flächen unterschiedliche Farben geben, was jedoch vielfach kaum praktikabel und häufig auch nicht gewollt ist. Eine andere Möglichkeit ist, Helligkeitsunterschiede z. B. so zu generieren, daß der Winkel, den die Flächennormale mit einer Verbidungslinie zu **einer** gedachten Lichtquelle die Helligkeit bestimmt (je kleiner der Winkel, desto heller die Fläche). Auf diese Weise wird auch die Krümmung einer (in Teilflächen unterteilten) Oberfläche besonders deutlich.

Unabdingbar für technische Zeichnungen ist jedoch die Möglichkeit, auch (meist sogar ausschließlich) die gesamte gewünschte Information durch Linien auszudrücken, bei denen zwei Typen zu unterscheiden (und auf unterschiedliche Weise zu erzeugen) sind:

- **Kanten** entstehen an der Verbindungsstelle zweier Flächen, wenn diese nicht tangential aneinanderstoßen.
- **Sichtkanten** ("Silhouette Lines") begrenzen bei gekrümmten Flächen (z. B.: Kugel, Torus, ...) den sichtbaren vom verdeckten Teil der Oberfläche.

Auch für die zu zeichnenden Linien muß ermittelt werden, welche Teile von davor liegenden Flächen verdeckt werden. Ein recht effektives Verfahren kann die Kombination mit dem Zeichnen von Flächen nach Prioritätenliste sein. Dieses Verfahren ist nicht für alle Ausgabemedien geeignet (ein Stiftplotter kann nicht die schwarzen Linien anschließend mit weißen Flächen überdecken), bietet sich aber natürlich gerade für die Bildschirmausgabe, bei der ein schnelles Anwortverhalten gewünscht ist, an. Für PostScript-Ausgabe kann es direkt übernommen werden (im Speicher eines PostScript-Gerätes wird das Bild erst komplett

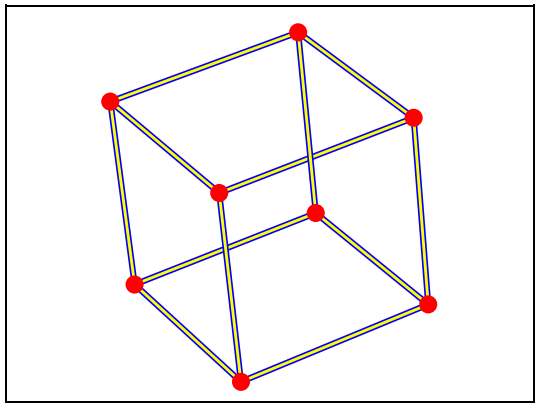
erzeugt und dann ausgegeben), für HPGL-Ausgabe ist diese Strategie des "Übereinanderzeichnens" ungeeignet.

Natürlich kann das beschriebene Verfahren die "Hidden Lines" (verdeckte Linien) und die "Hidden Surfaces" (verdeckte Flächen) nach der gleichen Strategie ermitteln und alle Ausgaben zunächst in einen programminternen Speicher bringen (wie es das PostScript-Gerät auch macht) und erst das fertige Bild zeichnen. Im interaktiven Betrieb ist es für den Konstrukteur im allgemeinen eher akzeptabel, wenn "auf dem Bildschirm wenigstens immer etwas passiert".

Eine Besonderheit ist für Berechnungsmodelle von Stabwerken, Drahtmodellen oder anderen "linienförmigen" räumlichen Objekten zu beachten. Die häufig fehlenden Querschnittsinformationen bei solchen Modellen gestatten natürlich keine Untersuchung des Verdeckens von Linien durch Flächen. In der GI-Toolbox ist für diese Fälle ein besonderer Linientyp vorgesehen, bei dem drei parallele Linien (einstellbarer Breite) gezeichnet werden, von denen die mittlere eine andere Farbe bekommen kann. Wenn dann die gleiche Strategie wie beim Zeichnen der Flächen nach "Prioritätenliste" angewendet wird, enthält die Zeichnung schließlich die Information, welcher Stab vor den anderen Stäben liegt.

1.8 Beispiel-Programme und Aufgaben

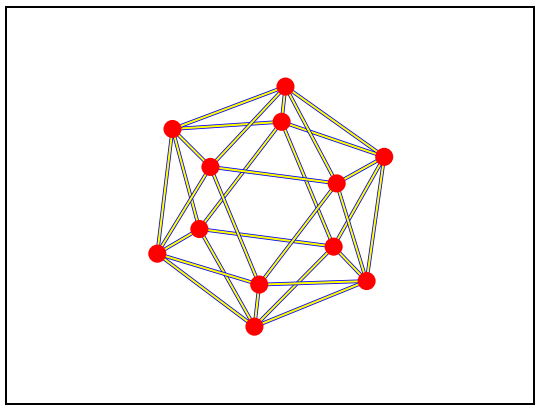
Das Programm **GRTEST76** arbeitet wie das Programm GRTEST75 und stellt dreidimensionale Stabwerke dar, die in der Datei STABWERK.DAT definiert sind, nutzt allerdings die GI-Routine **PTWDL_GI**, die eine "breite und zweifarbige Linie" zeichnet (Breite und Farben sind einstellbar). Wenn die gleiche Datei verwendet wird, die bereits für GRTEST75 genutzt wurde, erkennt man, daß dieser Linientyp gut geeignet ist, um zu zeigen, welche Linie vor einer anderen liegt, daß aber natürlich die Reihenfolge der Zeichenaktionen sinnvoll sein muß. Anderenfalls gibt es besonders "schöne Fehler" ("Escher-Bilder"), siehe nebenstehende Darstellung (oben).



Programm GRTEST76 (nach einer "y-Drehung")

Aufgabe 1.4: Man modifiziere das Programm GRTEST76 so, daß die Stäbe immer in der richtigen Reihenfolge gezeichnet werden. (Hinweis: Man ermittle die Entfernungen aller Stabmittelpunkte vom "Eye Point" und zeichne die Stäbe mit größeren Entfernungen zuerst).

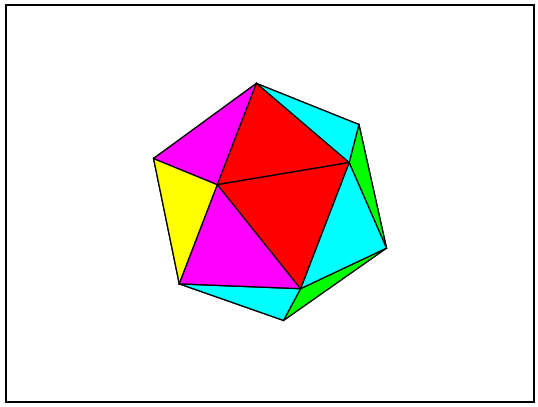
Nebenstehendes Bild zeigt das Ergebnis von Aufgabe 1.4, dargestellt wurde die Datei **DODESTAB.DAT**, die richtige Reihenfolge der Zeichenaktionen bringt korrekte "Überdeckungen".



Aufg. 1.4: Programm GRTEST76 (modifiziert)

Das Programm **GRTEST77** zeichnet Körper, die von "***N***-eckigen" Flächen begrenzt werden (***N*** muß für alle Flächen gleich sein). Die Beschreibung dieser Körper wird von der Datei **3DOBJEKT.DAT** gelesen, die eine ähnliche Struktur wie die Datei **STABWERK.DAT** hat. Zusätzlich muß nur in der ersten Zeile als dritter Wert ***N*** stehen, und in der "Koinzidenzliste" wird dann in jeder Zeile eine Fläche durch Angabe der ***N*** Punktnummern der zur Fläche gehörenden Knoten definiert. Zwei Beispiel-Files sind verfügbar: **WUERF3D.DAT** beschreibt einen Würfel, **DODEKAED.DAT** beschreibt ein Dodekaeder (Körper, der von 20 gleichseitigen Dreiecken begrenzt wird).

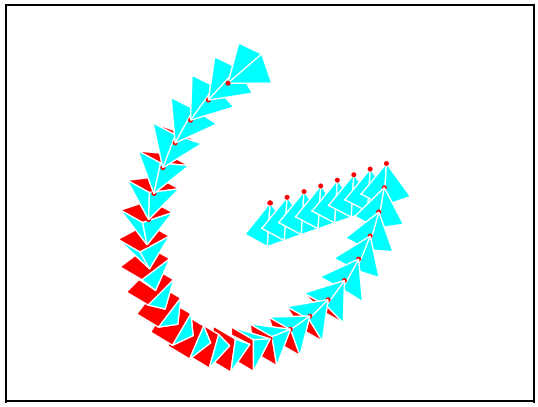
Bevor das Programm **GRTEST77** gestartet wird, muß die gewünschte Datei in **3DOBJEKT.DAT** umbenannt werden. Nebenstehendes Bild zeigt die Ausgabe für **DODEKAED.DAT**. Das Programm gestattet die Drehung des dargestellten Objekts um die drei Koordinatenachsen. Es enthält einen Sortier-Algorithmus, der eine "Prioritätenliste" definiert, mit der die Reihenfolge des Zeichnens der Flächen gesteuert wird, so daß der dargestellte Körper in jeder Lage nur die tatsächlich sichtbaren Flächen zeigt.



Ausgabe des Programms GRTEST77

Aufgabe 1.5: Man zeige durch Modifikation der Farbgebung für die Flächen (alle Flächen erhalten die Hintergrundfarbe des Bildschirms, nur die Kanten haben eine abweichende Farbe), daß der Algorithmus auch geeignet ist, um "Hidden-Line"-Bilder für Monochrom-Ausgabe zur erzeugen.

Das Programm **GRTEST79** demonstriert die Transformationen "Verschieben", "Rotation um Koordinatenachsen" und "Skalierung". Es wird ein Festlager-Symbol gezeichnet (mit der bereits mit dem Programm GRTEST77 demonstrierten "Hidden-Line"- und "Hidden-Surface"-Strategie), das mit den Tasten ’+’ und ’-’ vergrößert und verkleinert werden kann, mit den Cursor- und den Bild-Tasten wird es bewegt, mit ’X’, ’Y’ und ’Z’ rotiert es um die Koordinatenachsen. Im Programm GRTEST79 wird das "alte" Bild immer gelöscht, im Gegensatz zur nebenstehenden Skizze, die eine komplette Sequenz von Bewegungen zeigt.



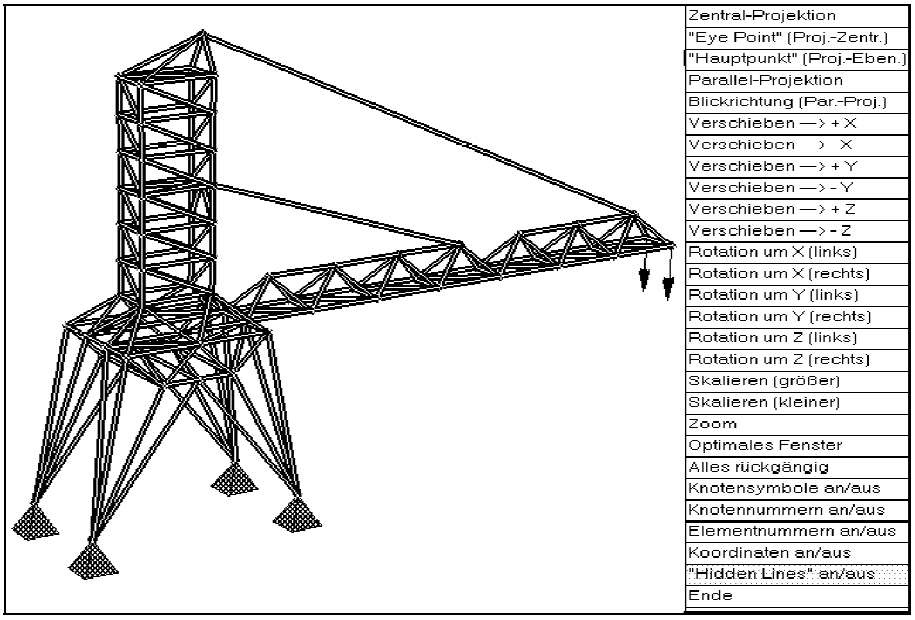
Das Programm **GRTEST80** demonstriert fast den kompletten Satz der 3D-Routinen aus der GI-Toolbox am Beispiel der Darstellung von räumlichen Fachwerken. Es ist menügeführt und gestattet die Einstellung und Änderung der Projektionsart und der Projektionsparameter (Referenzpunkt, "Eye Point", "Vektor der Blickrichtung"). Die Zeichnung kann verschoben und gedreht werden, Zoom-Funktion und andere Angebote ermöglichen eigentlich fast alle gewünschten Manipulationen der Darstellung.

Beim Compilieren und Linken des Programms GRTEST80 sind einige Besonderheiten zu beachten:

- Das Programm benutzt die Textausgabe-Funktionen der GI-Toolbox, die auf die MS-FORTRAN-Graphik-Fonts zugreifen. Diese müssen über die GI-Routine **GFTRG_GI** "angemeldet" werden. Man vergleiche vor dem Compilieren, ob der Pfad zu den Fonts, der als Parameter an GFTRG_GI übergeben wird, tatsächlich auf das Verzeichnis mit den Font-Files zeigt.
- Das Programm nutzt die Funktionalität des Parsers. Neben der GI-Library, der UTIL-Library (und damit auch der LLIO-Library) ist zusätzlich beim Linken die Parser-Library einzubinden.
- Das ausführbare Programm kann vom Linker mit der Standard-Segment-Vorgabe nicht erzeugt werden. Mit der Angabe von **/SE:1000** im Link-Befehl kann auch dieses Problem beseitigt werden.

Nach dem Starten des Programms wird nach der Datei gefragt, die das rechnerinterne Modell beschreibt. Es wird eine Datei erwartet, die den Informationsgehalt eines kompletten Finite-Elemente-Berechnungsmodells hat (neben den geometrischen Informationen also auch Informationen über Lager, Belastung, ...). Die Struktur dieser Datei entspricht exakt den **"FEMMOD"**-Dateien, die von den **FEMSET**-Programmen (bzw. den FEM-Programmen aus **CAMMPUS 4.2**) automatisch erzeugt werden.

Auf dem Server im PC-Pool ist die Datei **KRAN3D.DAT** zu finden. Wenn diese Datei dem Programm **GRTEST80** angeboten wird, wird das Objekt zunächst als einfache Strich-Graphik dargestellt. Nach der Wahl des Menüpunktes **"Hidden Lines an/aus"** (und einigen Verschiebungen und Rotationen) kann der Bildschirm etwa folgendermaßen aussehen:



GRTEST80: Darstellung eines 3D-Stabwerks und Menüangebot

Aufgabe 1.6: Mit dem Programm GRTEST80 sind die Auswirkungen von Änderungen der Projektionsparameter zu untersuchen (Hauptpunkt, "Eye Point" und "Blickrichtung"). Bei Darstellung in Zentralprojektion ist der "Eye Point" auch in das "Innere" des Objekts zu legen. Die für technische Zeichnungen typischen Darstellungen (z. B. "Draufsicht", "Seitenansicht", ...) sind als Parallelprojektionen zu erzeugen.