This document contains a report written as project at the Technical University of Dresden.

It was written by Nils Schmeisser and Kerstin Schuster at Technical University of Dresden. You may not redistribute nor resale this document or parts of it.
If you use the document or parts of it you should place the folloing in your index section:

"3-dimensionale Grafik von Funktionen der Art z=f(x,y), Math. Grundpraktikum"; Nils Schmeisser und Kerstin Schuster, Technische Universität Dresden

3-dimensionale Grafik von Funktionen der Art z=f(x,y)

Mathematisches Grundpraktikum

Autoren: Kerstin Schuster und Nils Schmeißer

Betreuer: **Dr. Scholz (Inst. für wiss. Rechnen)**

Zeitraum: **Dezember 1992/Januar 1993**

Inhalt

1.	Aufgabenstellung	3
2.	Vorbetrachtungen	4
2.1.	Projektionsarten	4
2.1.1.	Parallelprojektion	4
2.1.2.	Zentralprojektion	6
2.1.3.	Festlegung der Projektionsparameter	6
2.2.	Darstellungsformen	8
2.2.1.	Rasterlinienmodell	8
2.2.2.	Höhenlinienmodell	9
2.2.3.	Flächenfüllmodell	11
2.3.	Sichtbarkeitsalgorithmen	12
2.3.1.	Methode des gleitenden Horizonts	13
2.3.2.	Sehstrahl-Algorithmus	14
2.3.3.	Sortier-Algorithmus	15
2.3.4.	Schattenwurf (Raytrace-Modell)	15
2.3.5.	Vergleich der Sichtbarkeitsalgorithmen	16
2.4.	Analglyphendarstellung	17
3.	Implementation	18
3.1.	Allgemeines zur Bildschirmdarstellung	18
3.2.	Projektion	19
3.3.	Sichtbarkeitstests	20
3.3.1.	Methode des gleitenden Horizonts	20
3.3.2.	Sehstrahl-Algorithmus	21
3.3.3.	Sortier-Algorithmus	21
3.4.	Analglyphenmodell	22
3.5.	Höhenlinienmodell	23
3.6.	Die Unit "VISUAL"	24
3.6.1.	Datentypen	24
3.6.2.	Plot-Prozedur	25
3.6.3.	Aufruf	26
3.6.4.	Fehlerbehandlung	27
4.	Beispiel	28
5.	Anhang	30
	Beispiele	
	Listing	

1. Aufgabenstellung

Das Ziel der vorliegenden Arbeit besteht in der Implementation von Algorithmen zur Parallel- und Zentralprojektion räumlicher Objekte (hier speziell von explizit gegebenen Funktionen der Art z = f(x,y)), unter Einbeziehung von Sichtbarkeitsalgorithmen.
Um die Variabilität zu erhöhen, sollen die Objekt-Oberflächen (allgemein: Flächen) mit dem Rasterlinien-, dem Höhenlinien- und dem Flächenfüllmodell dargestellt werden können.
Um realistisch wirkende Bilder zu erzeugen, sollen Farbdarstellungen der Höhenstufen ("Landkartenmodell"), das Raytrace-Prinzip (Lichtquellen, nur Schattenwurf, keine Reflexion) und eine echte Stereo-Darstellung (Analglyphendarstellung) zur Anwendung kommen.

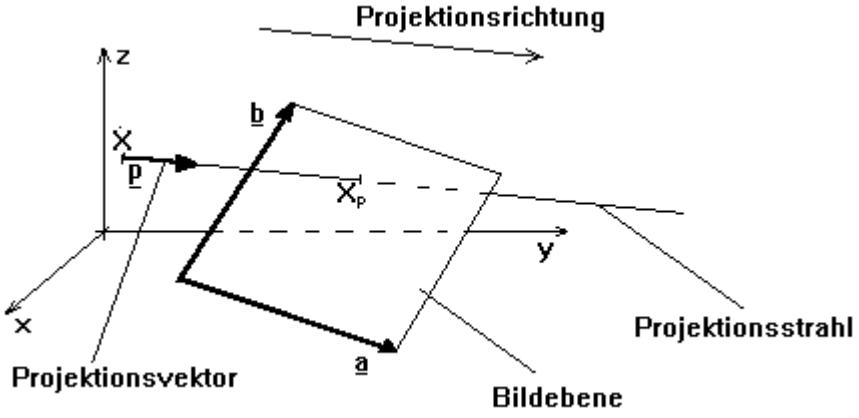
2. Vorbetrachtungen

2.1. Projektionsarten

2.1.1. Parallelprojektion

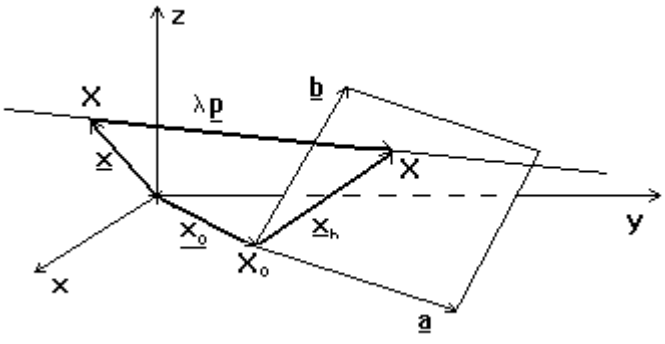
Bei der Parallelprojektion findet man die Bildpunkte als Schnittpunkte der Bildebene mit dem Projektionsstrahl, der vom Objektpunkt mit einer festgelegten Richtung ausgeht. Die Projektionsrichtung ist gleich dieser für alle Projektionsstrahlen vorgeschriebenen Richtung.

Zur mathematischen Beschreibung der Parallelprojektion ist damit die Festlegung der Bildebene und eines Projektionsvektors notwendig. Im weiteren wird jeder Projektionsstrahl mit diesem Vektor identifiziert.



Berechnung der Bildpunkte:

Im weiteren soll die Berechnung der Bildpunkte erläutert werden. Anhand der folgenden Skizze ist ersichtlich, daß sich die Koordinaten des Bildpunktes leicht mit Hilfe von Vektoradditionen berechnen lassen.



Es gilt: $\underline{x} = \underline{x}_0 + \underline{x}_h - \lambda\underline{p}$, wobei $\underline{x}_h = x\underline{b} + y\underline{a}$.
Dabei sind x und y die Koordinaten in der Bildebene. λ gibt die Entfernung zwischen Bild- und Originalpunkt an (bei normiertem $\underline{p}$).
In Komponentenschreibweise erhält man:

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} x_{0_1} \\ x_{0_2} \\ x_{0_3} \end{pmatrix} + x \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} + y \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} - \lambda \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix}$$

Daraus folgt:

$$\begin{pmatrix} x_1 - x_{0_1} \\ x_2 - x_{0_2} \\ x_3 - x_{0_3} \end{pmatrix} = \begin{pmatrix} a_1 & b_1 & -p_1 \\ a_2 & b_2 & -p_2 \\ a_3 & b_3 & -p_3 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ \lambda \end{pmatrix}$$

Man erhält also ein Gleichungssystem für x, y und λ, das immer eindeutig lösbar ist, da die Vektoren $\underline{a}$, $\underline{b}$ und $\underline{p}$ linear unabhängig sind. Das System läßt sich mit Hilfe der Cramer'schen Regel einfach lösen.
Man erhält somit:

$$x = \frac{D_x}{D}$$

$$y = \frac{D_y}{D}$$

$$\lambda = \frac{D_\lambda}{D}$$

wobei

$$D = \begin{vmatrix} a_1 & b_1 & -p_1 \\ a_2 & b_2 & -p_2 \\ a_3 & b_3 & -p_3 \end{vmatrix}$$

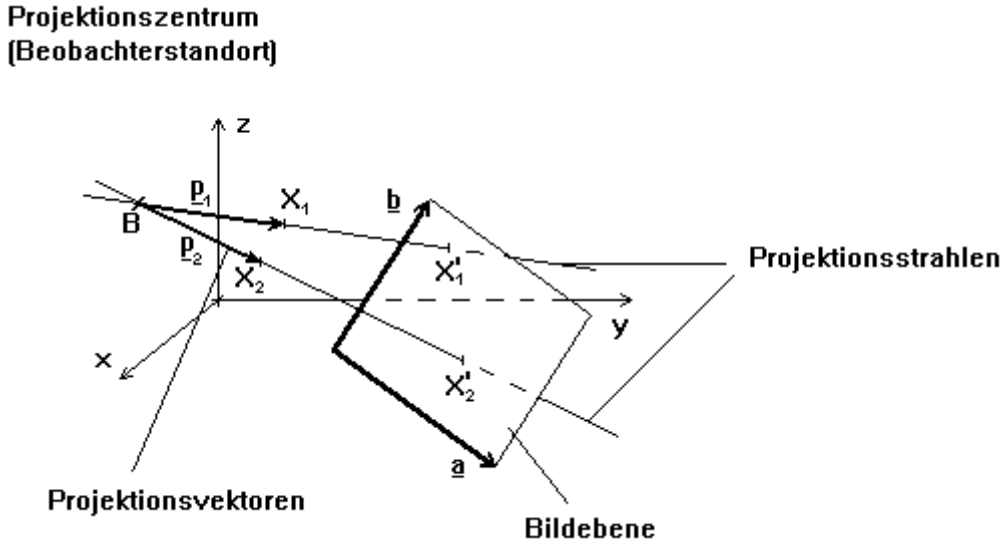
$$D_x = \begin{vmatrix} (x_1 - x_{0_1}) & b_1 & -p_1 \\ (x_2 - x_{0_2}) & b_2 & -p_2 \\ (x_3 - x_{0_3}) & b_3 & -p_3 \end{vmatrix}$$

$$D_y = \begin{vmatrix} a_1 & (x_1 - x_{0_1}) & -p_1 \\ a_2 & (x_2 - x_{0_2}) & -p_2 \\ a_3 & (x_3 - x_{0_3}) & -p_3 \end{vmatrix}$$

$$D_\lambda = \begin{vmatrix} a_1 & b_1 & (x_1 - x_{0_1}) \\ a_2 & b_2 & (x_2 - x_{0_2}) \\ a_3 & b_3 & (x_3 - x_{0_3}) \end{vmatrix}$$

2.1.2. Zentralprojektion

Im Gegensatz zur Parallelprojektion sind die Projektionsstrahlen für verschiedene Objektpunkte nicht parallel, sondern schneiden sich in einem Punkt, dem Projektionszentrum. In den meisten Fällen wird dieser Punkt mit dem Beobachterstandort identifiziert.



Zur formelmäßigen Beschreibung der Zentralprojektion ist also die Festlegung der Bildebene und des Projektionszentrums notwendig.
Die Richtung der Projektionsstrahlen (in Abhängigkeit vom Objektpunkt) ergibt sich in folgender Weise:

$$\underline{p} = \underline{x} - \underline{x}_B$$

wobei $\underline{x}$ der Ortsvektor des Objektpunktes X und $\underline{x}_B$ der Ortsvektor des Projektionszentrums ist.
In gewissen Fällen (Analglyphendarstellung) ist es sinnvoll, den Projektionsvektor zu normieren.
Die Berechnung der Bildpunkte erfolgt dann analog zur Parallelprojektion.

2.1.3. Festlegung der Projektionsparameter

Sowohl die Bildebene als auch der Projektionsvektor bei der Parallelprojektion werden mit Hilfe eines Winkels u bzw. u_p in der x-y-Ebene (von der x-Achse aus gemessen) und eines Winkels v bzw. v_p zwischen x-y-Ebene und Bildebene bzw. Projektionsvektor

charakterisiert. Zur genauen Festlegung der Bildebene ist zusätzlich ein Punkt X_0 (in Bezug auf das Ausgangskoordinatensystems) der Bildebene anzugeben, der im weiteren als Koordinatenursprung des Koordinatensystems in der Bildebene dient. Die Winkel und der Punkt sind vom Nutzer frei wählbar.
Aus den Winkeln werden die Vektoren $\underline{a}$ und $\underline{b}$, die ein Koordinatensystem in der Bildebene bilden, sowie im Falle der Parallelprojektion die Koordinaten des Projektionsvektors $\underline{p}$ berechnet (jeweils in Bezug auf das Ausgangskoordinatensystems).

Berechnung von $\underline{a}$, $\underline{b}$ und $\underline{p}$:

$$\underline{a} = \begin{pmatrix} \cos u \\ \sin u \\ 0 \end{pmatrix}$$

$$\underline{b} = \begin{pmatrix} \cos v \cdot \cos\left(90^\circ + u\right) \\ \cos v \cdot \sin\left(90^\circ + u\right) \\ \sin v \end{pmatrix}$$

$$\underline{p} = \begin{pmatrix} \cos v_p \cdot \cos u_p \\ \cos v_p \cdot \sin u_p \\ \sin v_p \end{pmatrix}$$

Bei freier Wahl der Winkel ist es möglich, daß bei der Parallelprojektion der Projektionsstrahl parallel zur Bildebene verläuft. In diesem Fall ist eine Projektion nicht möglich. Um diese Möglichkeit auszuschließen , wird getestet, ob der Projektionsstrahl senkrecht zum Normalenvektor der Bildebene verläuft. Wenn ja, wird die Projektion nicht durchgeführt.
Projektionsstrahl und Normalenvektor stehen genau dann senkrecht aufeinander, wenn das Skalarprodukt von Projektionsvektor und Normalenvektor gleich Null ist.
Der Normalenvektor berechnet sich aus den Komponenten a_i (i=1..3) des Vektors $\underline{a}$ und den Komponenten b_i (i=1..3) des Vektors $\underline{b}$ wie folgt:

$$\underline{n} = \begin{vmatrix} \underline{i} & \underline{j} & \underline{k} \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{vmatrix}$$

$$= (a_2 b_3 - a_3 b_2)\,\underline{i} + (a_3 b_1 - a_1 b_3)\,\underline{j} + (a_1 b_2 - a_2 b_1)\,\underline{k}$$

Für die Zentralprojektion ist es notwendig, das Projektionszentrum außerhalb der Bildebene zu wählen.

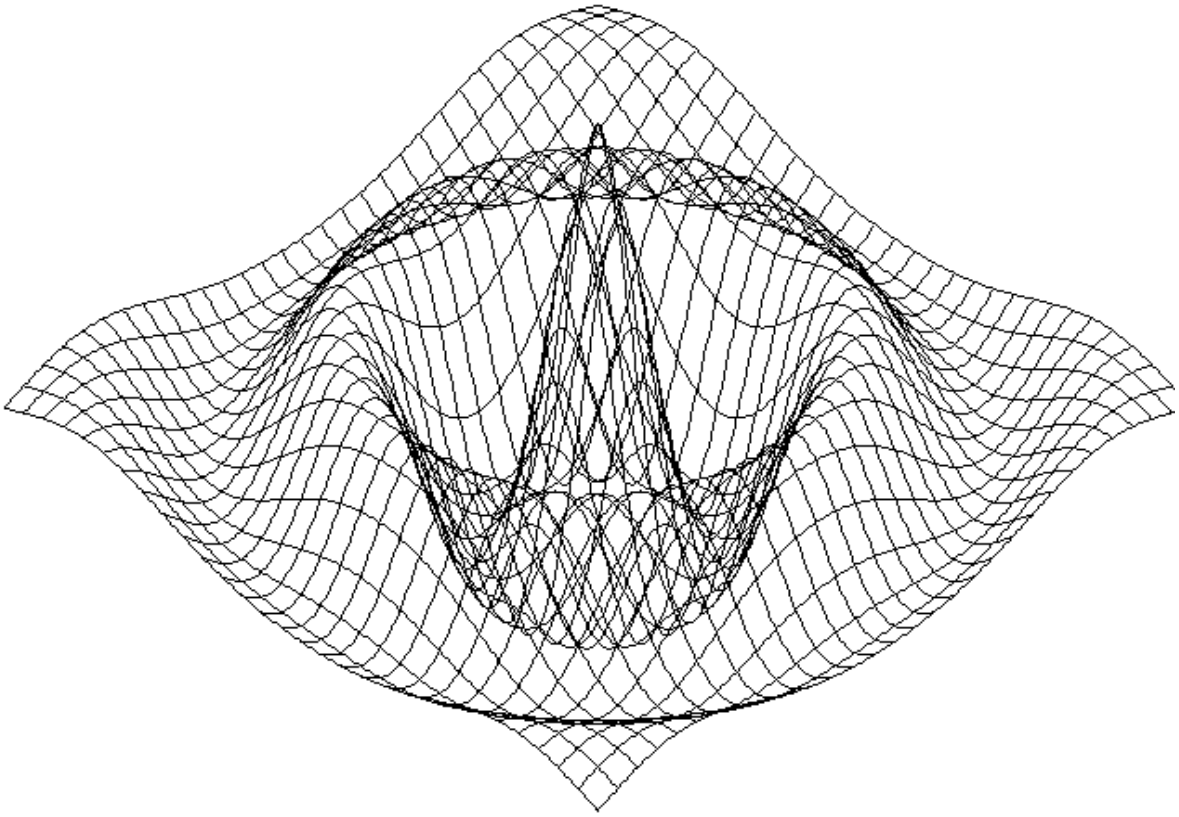
2.2. Darstellungsformen

2.2.1. Rasterlinienmodell

Die folgende Abbildung zeigt zunächst die Funktion

$$z = f(x,y) \;=\; 10 \cdot \; e^{-\frac{\sqrt{x^2+y^2}}{5}} \;\cdot \cos\left(\sqrt{x^2+y^2}\right) \quad ,$$

die im weiteren als Standardfunktion verwendet wird.



Die, die Fläche charakterisierenden Kurven werden durch folgende Gleichungen erzeugt. Man wählt zunächst genau eine der Veränderlichen x oder y beliebig aus dem Definitionsbereich, aber fest. Sei o.B.d.A. x fest gleich x_0, dann heißt

$$C(x_0) = \left\{(x_0, y, z) \;\in\; E^3 \;:\; z = r(y) = f(x_0, y)\right\}$$

Rasterlinie auf der Fläche.

Sei

$$D = \left\{(x,y) \;\in\; R^2 \;:\; x_{min} \le x \le x_{max} \;\wedge\; y_{min} \le y \le y_{max}\right\}$$

der Definitionsbereich und n_y bzw. n_x die Anzahl der Rasterlinien, deren Urbilder parallel

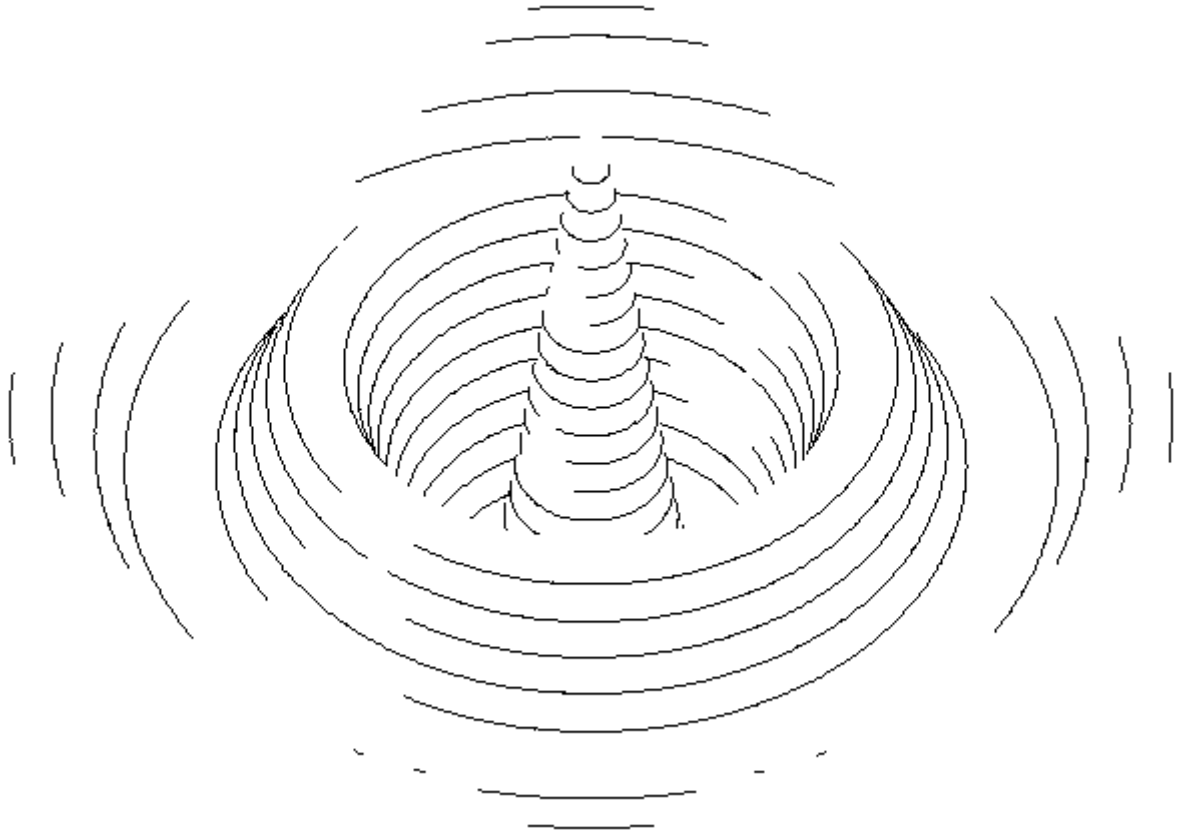
zur y-Achse bzw. x-Achse verlaufen, so ist

$$C_y = \left\{ C(x_0) : x_0 = x_{min} + (x_{max} - x_{min}) \cdot \frac{i}{n_y} \quad \text{mit} \quad i = 0..n_y \right\}$$

die Menge aller darzustellenden Rasterlinien in y-Richtung. Analog erhält man die Menge aller Rasterlinien in x-Richtung.

2.2.2. Höhenlinienmodell

Die Charakteristik einer Höhenliniendarstellung ist dem folgenden Bild zu entnehmen.

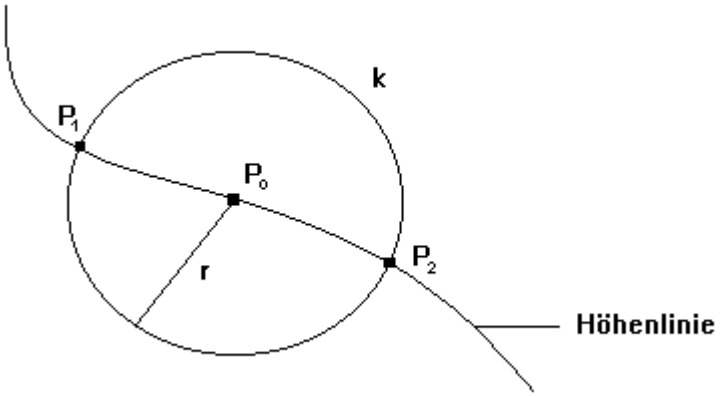


Auch hier lassen sich alle Kurven, die die Fläche beschreiben, zunächst sehr einfach formelmäßig erfassen. Man wählt $z=z_0$ (z_0 beliebig, aber fest) und sucht alle "Kurven gleichen Höhenniveaus"

$$C(z_0) = \left\{ (x, y, z_0) \in E^3 : z_0 = f(x,y) \right\} .$$

Um eine dieser Kurven zu finden, muß man also die Gleichung $z_0 = f(x,y)$ eindeutig nach x oder y auflösen können.

Geht man davon aus, daß zunächst ein Startwert (x_0,y_0) mit der Eigenschaft $z_0 = f(x_0,y_0)$ vorgegeben ist, so kann man eine iterative Lösung angeben. Man kann sich die hierfür erforderlichen Überlegungen an folgender Skizze veranschaulichen:



Wählt man r hinreichend klein, und setzt man voraus, daß die durch $z_0=f(x,y)$ beschriebenen Kurven stetig sind, so findet man vom Startwert ausgehend weitere Punkte der Höhenlinie (P_1, P_2) nach folgendem Verfahren:

1. Lege Kreis um P_0 mit Radius r
2. Suche Punkte P_i des Kreises $k(P_0,r)$, für die $z_0=f(x_i,y_i)$ ist
3. Wiederhole das Verfahren mit jedem P_i als Startwert, bis eine Abbruchbedingung erfüllt ist

Die Höhenlinie wird jetzt durch den, die Punkte P_i verbindenden Polygonzug approximiert.

Abbruchbedingungen:

1. Der Punkt P_i liegt außerhalb des Definitionsbereiches
2. Der Punkt P_i liegt auf einem schon gefundenen Teil der Höhenlinie

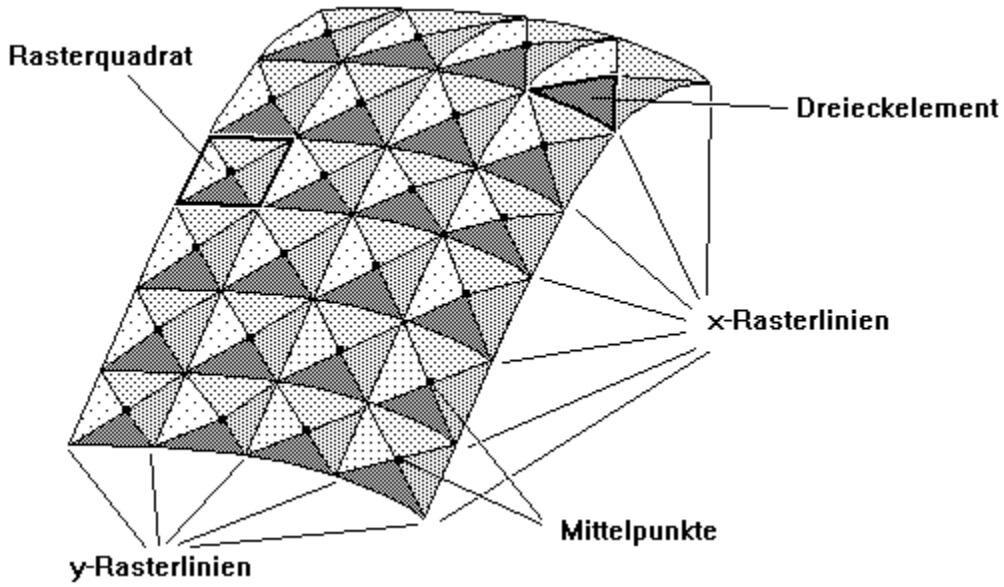
Damit beschreibt

$$C_h = \left\{ C(z_0) : z_0 = z_{min} + (z_{max} - z_{min}) \cdot \frac{i}{n_h} \quad \text{mit} \quad i = 0..n_h \right\}$$

die Menge aller darzustellenden Höhenlinien. n_h gibt die Anzahl der Höhenstufen an. z_{min} bzw. z_{max} sind das Minimum bzw. Maximum der Funktion z=f(x,y) auf dem Definitionsbereich.

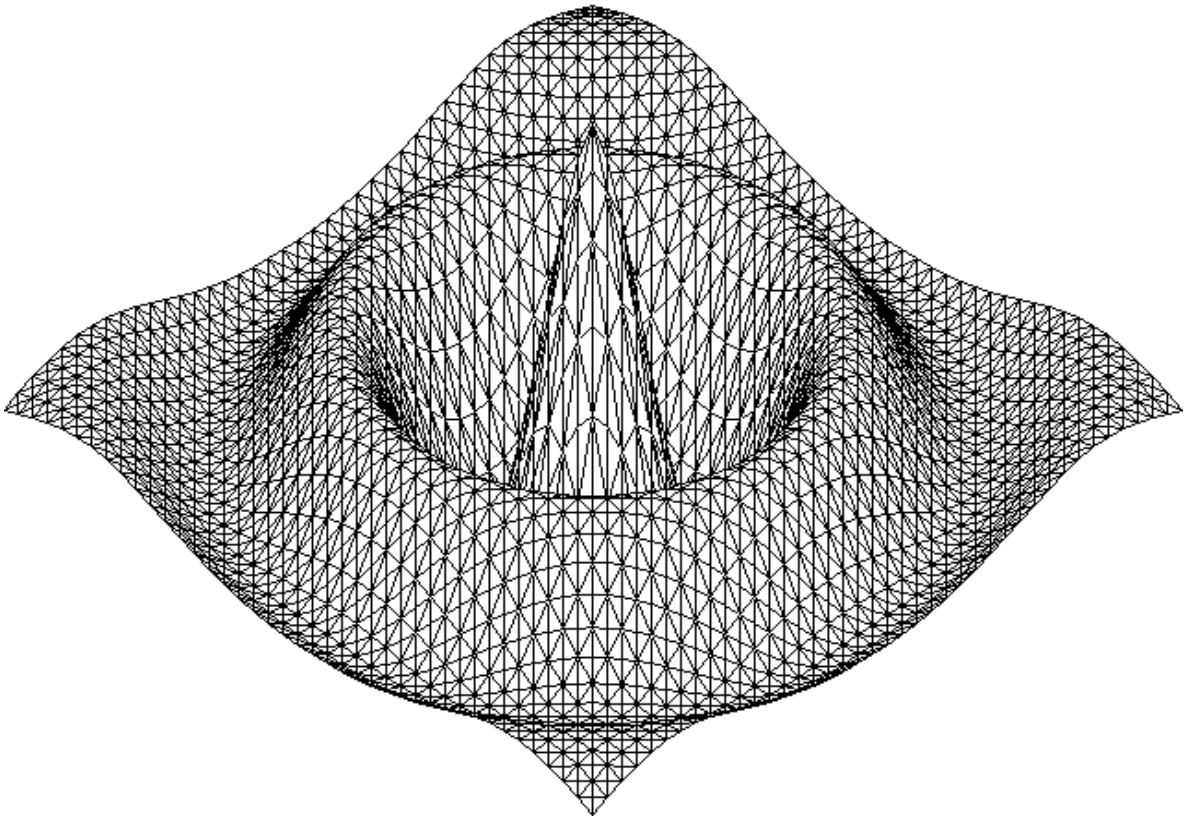
2.2.3. Flächenfüllmodell

Wie aus folgender Abbbildung ersichtlich, kann man dieses Modell aus dem Rasterlinienmodell ableiten.



Man zerlegt zuerst die Fläche in Rasterquadrate (durch Rasterlinien begrenzt) und dann jedes dieser Quadrate in vier Dreieckelemente. Die Fläche wird dann durch eine Menge ebener Flächenstücke approximiert. Man wählt Dreiecke, da diese keine Verwindungen bezüglich der Eckpunkte aufweisen. Die Dreieckelemente werden einfach mit einem bestimmten Füllmuster und einer bestimmten Farbe ausgefüllt.

Das folgende Bild zeigt die Standardfunktion:



2.3. Sichtbarkeitsalgorithmen

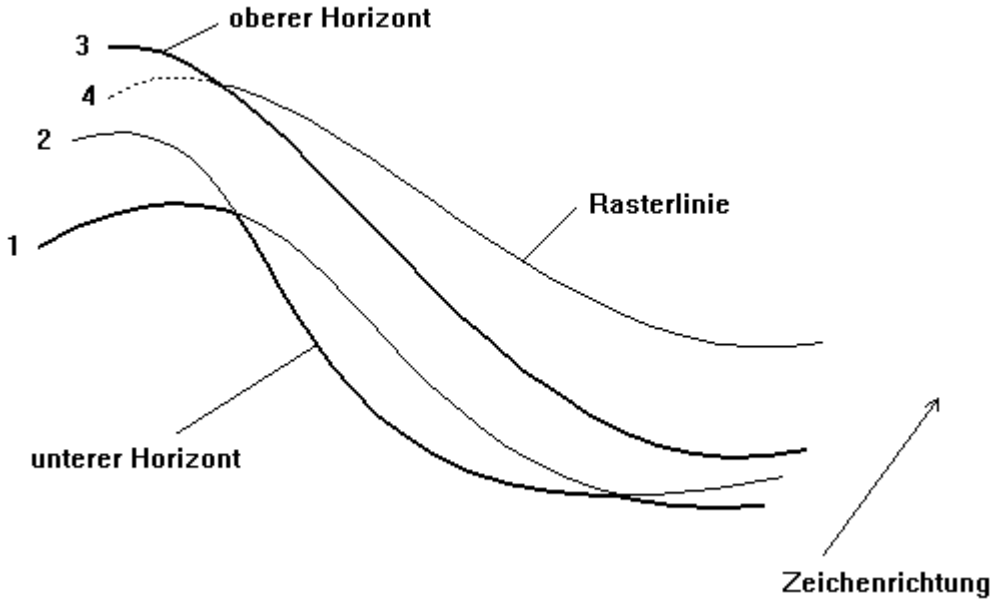
Um ein reales Abbild der Wirklichkeit zu schaffen, muß man natürlich auch die Sichtbarkeitsverhältnisse beachten. Dazu stehen zwei grundlegende Typen von Tests zur Verfügung. Hierbei handelt es sich um:

1. Objektraum-Algorithmen: Die Sichtbarkeit eines Punktes wird ausschließlich unter Zuhilfenahme von Informationen aus dem Objektraum (E^3) bestimmt.
2. Bildraum-Algorithmen: Nur Informationen, die aus den projizierten Objekten herleitbar sind, werden verwendet.

Diese Algorithmen können auch kombiniert werden.

2.3.1. Methode des gleitenden Horizonts

Hierbei handelt es sich um einen Bildraum-Algorithmus. Er soll an Hand der folgenden Skizze erläutert werden.



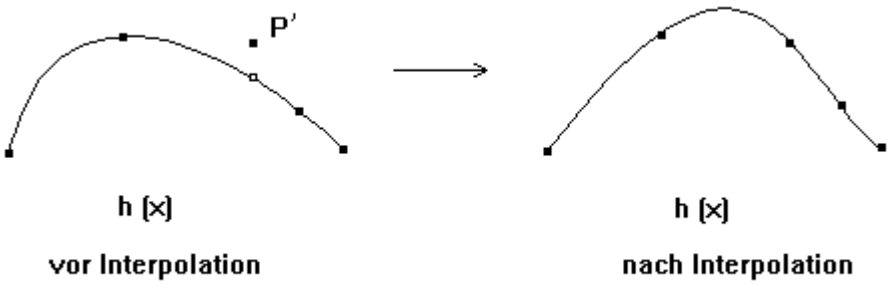
Zunächst muß sichergestellt werden, daß die Zeichenrichtung von "vorn" nach "hinten" verläuft (! Information aus dem Objektraum).
An der Skizze verdeutlicht man sich, daß ein Punkt genau dann unsichtbar ist, wenn er sich über dem unteren und unter dem oberen Horizont befindet. Anderenfalls ist er sichtbar.

Formelmäßig kann dies in folgender Weise erfaßt werden:

Setze $h_o(x´)=-g$ und $h_u(x´)=g$, wobei g hinreichend groß ist
(Stellt sicher, daß die erste Rasterlinie sichtbar ist).
$h_o(x´)$ und $h_u(x´)$ sind stetig und auf ganz R definiert.

1. Zu testender Punkt P(x,y,z) -> P´(x´,y´)
2. Ist $y´ \geq h_u(x´)$ **und** $y´ \leq h_o(x´)$ so ist P unsichtbar -> STOP
3. P ist sichtbar, setze:
   falls $y´ < h_u(x´)$, dann $h_u(x´):=y´$
   falls $y´ > h_o(x´)$, dann $h_o(x´):=y´$

Da in der Praxis nur diskrete Punktmengen betrachtet werden können, ist es notwendig h_o und h_u nach jedem sichtbaren Punkt neu zu interpolieren.

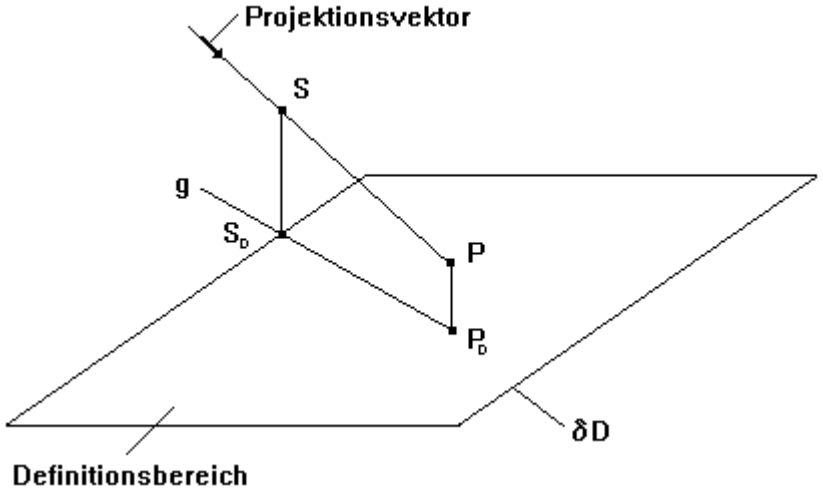


Im einfachsten Fall genügt lineare Interpolation.

2.3.2. Sehstrahl-Algorithmus

Dieser Objektraum-Algorithmus läßt sich in einem Satz formulieren:

> Laufe vom zu testenden Punkt P entgegengesetzt zur Richtung des Projektionsvektors bis zum Rand des Definitionsbereiches und teste, ob die Projektionsgerade die durch z=f(x,y) bestimmte Fläche schneidet. Ist dies der Fall, so ist der Punkt P unsichtbar, anderenfalls ist er sichtbar.



Man bestimmt zunächst den Punkt S, indem man zuerst den Punkt S_D mit:

$$S_D = g \cap \partial D\,, \quad \text{mit} \quad g: \underline{x} = -\underline{p}\Big|_{z=0} \cdot t + \underline{x}_p\Big|_{z=0} \quad \text{und} \quad t \geq 0$$

ermittelt. Dann ergibt sich

$$S: \underline{s} = -\underline{p} \cdot t_S + \underline{x}_P$$

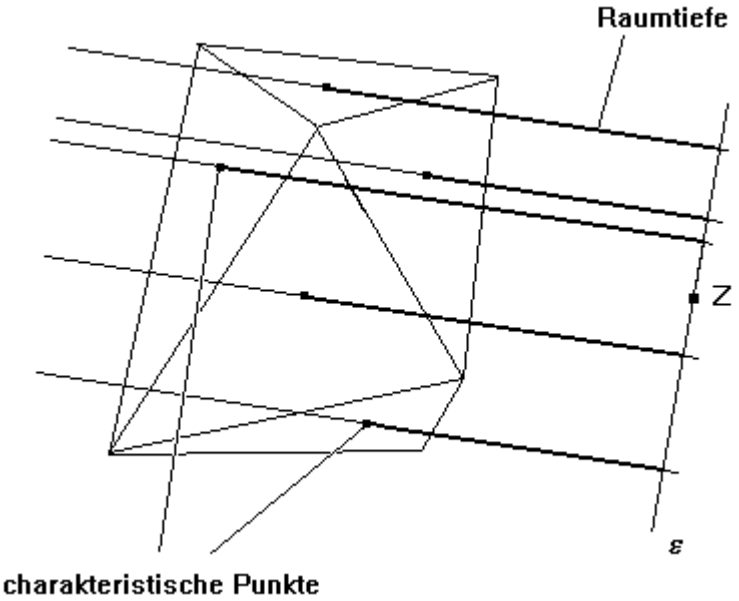
Ist t_S=0, so ist der Punkt P sichtbar (P liegt auf dem Rand des Definitionsbereiches). Anderenfalls sucht man t´ mit :

$$-p_3 \cdot t' + x_{P_3} = f\left(-p_1 \cdot t' + x_{P_1},\ -p_2 \cdot t' + x_{P_2}\right) \quad \text{und} \quad 0 < t' \leq t_S$$

Existiert so ein t´, so ist der Punkt P unsichtbar, anderenfalls ist er sichtbar.

2.3.3. Sortier-Algorithmus

Hiebei handelt es sich um einen gemischten Algorithmus. Man zerlegt zunächst die darzustellende(n) Fläche(n) in einfache geometrische Objekte, wie zum Beispiel Quadrate oder Dreiecke. Für jedes dieser Oberflächenelemente wird ein charakteristischer Punkt bestimmt und mit der zugehörigen Raumtiefe markiert.



charakteristische Punkte

Um das Funktionieren des Algorithmus zu garantieren, wählt man Z so, daß alle Objekte und die Bildebene "auf einer Seite von Z liegen". Bei der Zentralprojektion legt man Z in das Projektionszentrum.
Die Raumtiefe eines Punktes P erhält man jetzt, indem man durch Z eine Ebene ε legt, die Senkrecht zur zum Punkt P zugehörigen Projektionsgeraden liegt und anschließend die Entfernung von P zu ε bestimmt (=Raumtiefe).
Die Oberflächenelemente werden jetzt beginnend mit der größten Raumtiefe dargestellt. Dieser Algorithmus ist nur bei Flächenfüllmodellen anwendbar.

2.3.4. Schattenwurf (Raytrace-Modell)

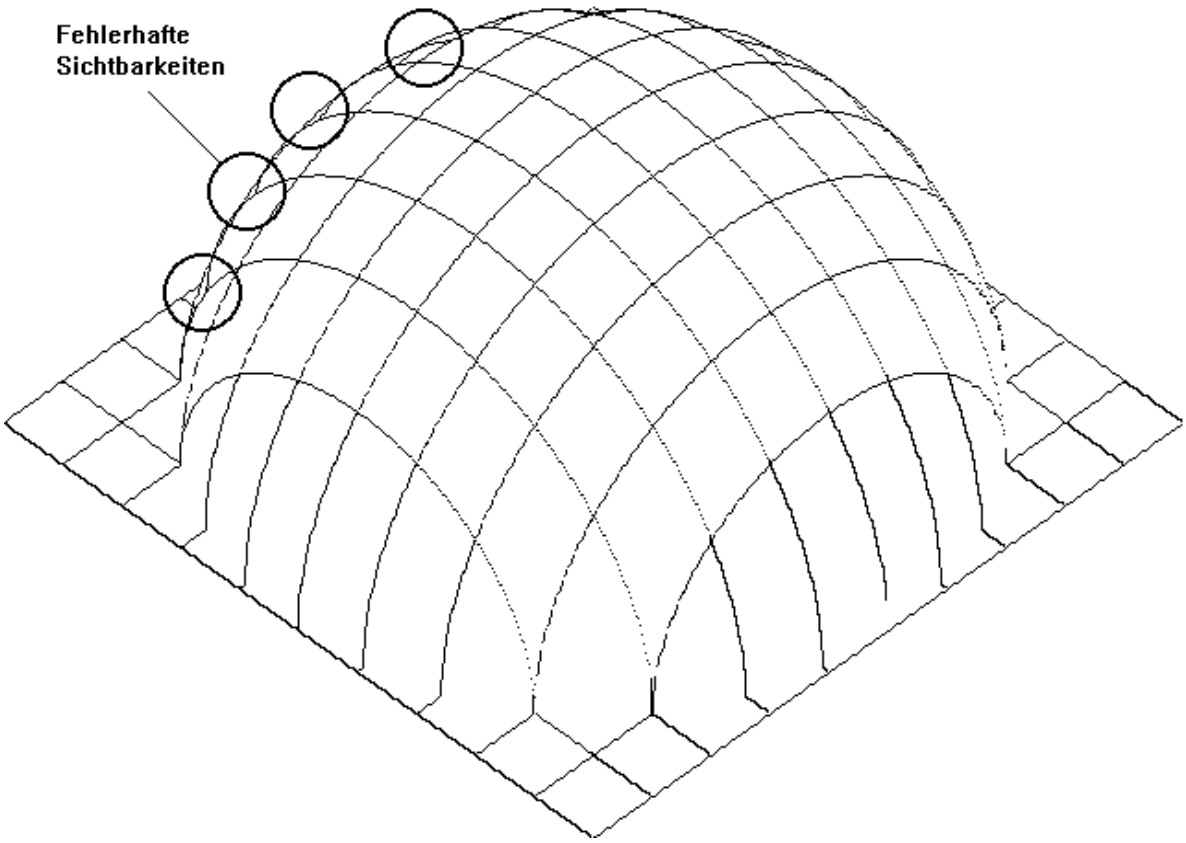
Fügt man in den Objektraum genau eine Lichtquelle ein und werden Reflexionserscheinungen vernachlässigt, so reduziert sich die Frage, ob ein Punkt beleuchtet oder unbeleuchtet ist, auf das Problem, ob er von der Lichtquelle aus sichtbar oder unsichtbar ist. Von den oben angeführten Algorithmen ist die Methode des gleitenden Horizonts hierfür nicht anwendbar.

2.3.5. Vergleich der Sichtbarkeitsalgorithmen

Von allen drei Algorithmen ist der *Sehstrahl-Algorithmus* der langsamste, aber auch der genaueste.

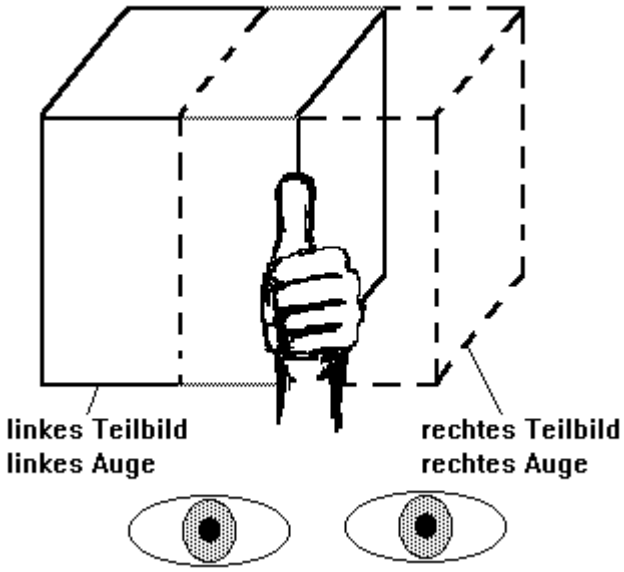
Der *Sortier-Algorithmus* liefert im Gegensatz zu den beiden anderen Algorithmen meist nur an Polyeder erinnernde Bilder. Er ist darüber hinaus äußerst speicherintensiv (n*10 Byte, wobei n die Anzahl der Oberflächenelemente ist, die Methode des gleitenden Horizonts benötigt dagegen nur <maximale x-Auflösung des Bildschirms>*2 Byte).

Die *Methode des gleitenden Horizonts* liefert in manchen Fällen nicht akzeptable Ergebnisse (s. Abb.).



2.4. Analglyphendarstellung

Das Prinzip des stereoskopischen Sehens beim Menschen läßt sich mit Hilfe des bekannten "Daumensprung-Experiments" erläutern.



Führt man dieses Experiment mit unterschiedlichen Abständen zwischen Augen und Daumen durch, so stellt man fest, daß Δs um so größer ist, je näher sich der Daumen an den Augen befindet. Näherungsweise kann man $\Delta s \sim (1/\text{Daumenabstand})$ setzen.
Umgekehrt induzieren verschiedene Δs verschiedene Abstände vom Beobachter.

Da man die Raumtiefe und damit das einem Objektpunkt P zugeordnete Δs berechnen kann, ist es möglich, die dem linken bzw. rechten Auge zugehörigen Teilbilder zu zeichnen. Stellt man sicher, das dem linken bzw. rechten Auge jeweils **nur** das zugehörige Teilbild sichtbar ist, so entsteht für den Betrachter der Eindruck, ein reales Objekt des E^3 zu sehen.

Die einfachste Methode hierfür besteht darin, die Teilbilder in unterschiedlichen Farben zu malen, die durch voneinander verschiedene Farbfilter jeweils vollständig reflektiert werden. Der Betrachter trägt dann eine Spezialbrille, die eben diese Farbfilter als Gläser hat.

Zwei professionellere Varianten sind

1. die Verwendung von Bilschirmen mit höherer Bildwiederholfrequenz, bei denen die beiden Teilbilder abwechselnd eingeblendet werden, in Kombination mit einer "LCD-Brille", sowie
2. dem unter dem Schlagwort "Virtual Reality" bzw. "Cyberspace" bekannten 2-Bildschirmprinzip.

Beim Analglyphenverfahren berechnet man zunächst wieder die Koordinaten (x',y') des projizierten Punktes, sowie weiterhin das zugehörige Δs und zeichnet dann den Punkt im linken Teilbild mit $(x'-\Delta s/2,y')$ und im rechten Teilbild mit $(x'+\Delta s/2,y')$.

3. Implementation

3.1. Allgemeines zur Bildschirmdarstellung

Für die vorliegende Arbeit wurde davon ausgegangen, daß es sich bei der vorhandenen Hardware um Rasterbildschirme nach gängigem Standard handelt. Insbesondere erfordern die Farbdarstellung und die Analglyphendarstellung im Minimum eine EGA-, besser eine VGA-Grafikkarte und Farbmonitor.
Charakteristisch für diese Geräte ist eine begrenzte Anzahl von Punkten (bei der VGA-Grafik: 640*480) und Punktkoordinaten im Bereich
(0..maximales_X) x (0..maximales_Y). Damit läßt sich eine Menge von Punkten aus dem E^2 mit den Ausdehnungen (x_{min},x_{max}) x (y_{min},y_{max}) über folgende Vorschrift vollständig darstellen:

$x_B:=[(x-x_{min})*s_x]$
$y_B:=[(y-y_{min})*s_y]$, mit

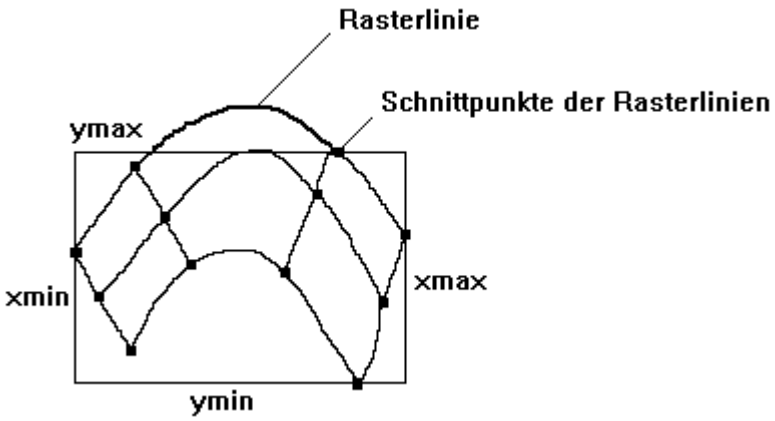
$s_x:=maxx_B/(x_{max}-x_{min})$ und
$s_y:=maxy_B/(y_{max}-y_{min})$,

wobei $maxx_B$ bzw. $maxy_B$ die maximale Anzahl der Punkte des Bildschirms in x bzw. y Richtung sind. Im Allgemeinen sind s_x und s_y voneinander verschieden. Dies würde bei der Darstellung eine Verzerrung im Verhältnis $s_x:s_y$ bedeuten, die man aber ausgleicht, indem man einen Faktor s einführt, mit

$$s:=\min\{s_x,s_y\}.$$

Bevor man an das Plotten eines Bildes geht, ist es also sinnvoll, zuerst dessen Ausdehnung und damit dann den Faktor s zu bestimmen.

Da hier nur spezielle Bilder behandelt werden, die generell von Rasterlinien überzogen werden können, ist es sehr einfach, die gerade gestellte Forderung zu erfüllen. Man projiziert einfach alle Schnittpunkte der Rasterlinien auf der darzustellenden Fläche in die Bildebene (E^2) und bestimmt damit die Ausdehnung der enstandenen Punktmenge.



Die Skizze verdeutlicht gleichzeitig das Problem, das bei dieser Vorgehensweise auftreten kann. Weicht die Ausdehnung des Bildes zu graß von den, aus den Rasterlinien gewonnenen Daten ab, so wird das Bild eventuell nicht ganz vollständig auf dem Bildschirm dargestellt. Man kann aber davon ausgehen, daß bei einer großen Anzahl von Rasterlinien ($\geq$ 10) dieser Fehler vernachläßigt werden kann.
Damit ergibt sich jetzt eine kompakte Darstellung für die Darstellung eines beliebigen Punktes der Fläche auf dem Bildschirm:

0. *Durchlaufe alle Schnittpunkte der Rasterlinien und bestimme x'_{min}, x'_{max}, y'_{min}, y'_{max} sowie den Faktor $s:=min\{ maxx_B/(x'_{max}-x'_{min}) , maxy_B/(y'_{max}-y'_{min}) \}$ (dieser Schritt ist nur einmal auszuführen)*

Für jeden Punkt:

1. *Bestimme die projizierten Koordinaten (x',y')*
2. *$x_B:=INTEGER((x'-x'_{min})*s)$*
   $y_B:=INTEGER((y'-y'_{min})*s)$
3. *Zeichne den Punkt (x_B,y_B)*

3.2. Projektion

Die Projektion erfolgt nach dem unter 2.1.1. angegebenen Verfahren.
Der komplette Projektionsalgorithmus für einen Punkt hat damit folgende Gestalt:

0. *Gegeben: $\underline{e}_1$, $\underline{e}_2$, $\underline{x}_0$ und bei Parallelprojektion der Projektionsvektor $\underline{p}$*
1. *Falls Zentralprojektion, dann berechne den Projektionsvektor $\underline{p}$:*
   $\underline{p}=\underline{x}-\underline{x}_{Beobachter}$
2. *Berechne x', y' aus*

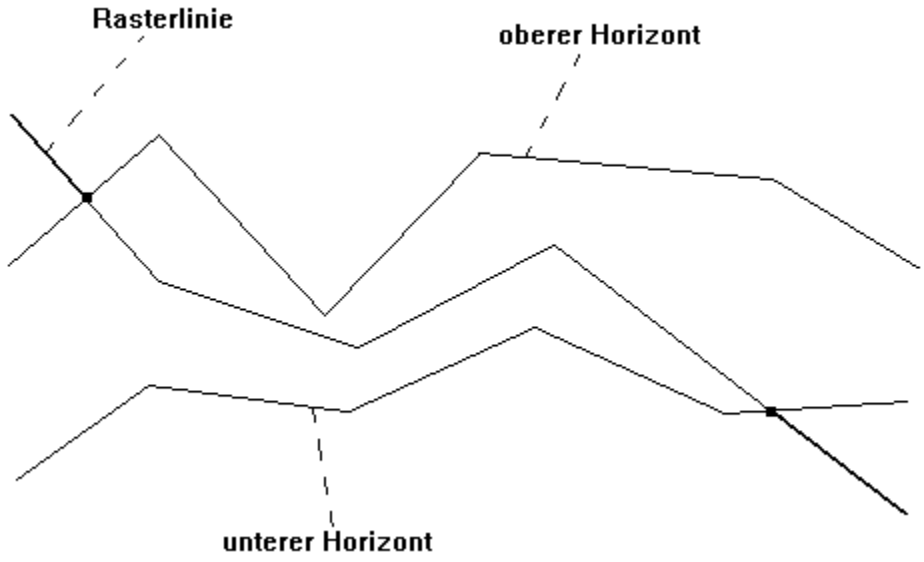
$$\begin{bmatrix} x' \\ y' \\ \lambda \end{bmatrix} = \begin{bmatrix} e_{11} & e_{21} & -p_1 \\ e_{12} & e_{22} & -p_2 \\ e_{13} & e_{23} & -p_3 \end{bmatrix}^{-1} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Die Vektoren $\underline{e}_1$, $\underline{e}_2$ und $\underline{p}$ sind linear unabhängig (sonst liegt entweder der Projektionsstrahl (Parllelprojektion) oder der Beobachterstandpunkt in der Bildebene (Zentralprojektion)). Damit existiert die inverse Matrix immer und ist berechenbar.

3.3. Sichtbarkeitstests

3.3.1. Methode des gleitenden Horizonts

Wie oben dargestellt, kann man die Horizonte durch beliebige (polynomiale, ...) Interpolation beschreiben. Wählt man speziell lineare Interpolation zwischen bekannten Horizontpunkten, so läßt sich das zugehörige Verfahren wieder recht einfach darstellen.



Es gibt jetzt zwei Möglichkeiten, die Schnittpunkte der Rasterlinie mit den Horizonten zu bestimmen:

1. "richtiges Ausrechnen"
2. eine neue Linienziehprozedur schreiben, die beim Setzen jedes Punktes einer Linie dessen Sichtbarkeit testet.

Die zweite Variante ist wesentlich einfacher zu handhaben und wird deshalb hier verwendet. Es ist jetzt sogar ausreichend, die Horizonte nicht mehr als stetige Funktionen, sondern als eine Menge diskreter Punkte zu betrachten. Günstig ist es, ein Feld H_i mit i=0..maxx_B und H_i aus N für einen Horizont zu verwenden. Kompakt formuliert heißt das:

0. setzte $HO_i:=-1$
$HU_i:=maxx_B$

Für jede Linie (Strecke) von $A(x_A,y_A)$ *nach* $B(x_B,y_B)$

1. setze $\underline{a}:=\underline{x}_B-\underline{x}_A$, $\underline{x}:=\underline{x}_A$, $t:=0$
2. $x_2:=a_x*t+x_x$, $y_2:=a_y*t+x_y$
3. $y_2>HO[x_2]$ *--> setze Punkt* (x_2,y_2) *und* $HO[x_2]:=y_2$
$y_2<HU[x_2]$ *--> setze Punkt* (x_2,y_2) *und* $HU[x_2]:=y_2$

4. *t:=t+1/Abstand(A,B)*
5. *falls t<1 gehe zu 2.*

3.3. 2. Sehstrahl-Algorithmus

Wie beschrieben, wird gefragt, ob der Sehstrahl zwischen dem zu testenden Punkt der Fläche und dem Rand noch einen weiteren gemeinsamen Punkt mit der Fläche hat. An dieser Stelle wird der Zwischenwertsatz folgendermaßen ausgenutzt:

0. *Sei $\underline{x}_B$ - Beobachterpunkt, $\underline{x}_P$ - Punkt P der Fläche setze $\underline{a}:=\underline{x}_P-\underline{x}_B$, $\underline{x}:=\underline{x}_P$*
1. *laufe auf dem Sehstrahl vom Punkt P um dt auf den Beobachter zu und bestimme, auf welcher Seite der Fläche sich der aktuelle Standort befindet:*

   $s_0:=(\underline{x}_P-\underline{a}*dt)_z-f((\underline{x}_P-\underline{a}*dt)_x,(\underline{x}_P-\underline{a}*dt)_y)$

   setze t:=dt

2. *laufen weiter auf den Rand (bzw. Beobachter) zu (mit Schrittweite dt):*

   $s:=(\underline{x}_P-\underline{a}*t)_z-f((\underline{x}_P-\underline{a}*t)_x,(\underline{x}_P-\underline{a}*t)_y)$

3. *der aktuelle Punkt wechselt auf die andere Seite ($s*s_0\leq 0$) der Fläche --> > Punkt P unsichtbar, Abbruch*
4. *wiederhole (t:=t+dt) ab 2., bis der Rand erreicht ist ($t>t_{max}$)*
5. *Punkt P ist sichtbar, falls der Rand erreicht wurde*

3.3.3. Sortier-Algorithmus

Zunächst werden hier die Raumtiefen für alle charackteristischen Punkte der Oberflächenelemente berechnet und in einer linearen Liste abgelegt (siehe auch 2.3.3.). Danach werden die Elemente mit der größten Raumtiefe beginnend nach unten sortiert. Zum Sortieren wird das rekursive QUICK-SORT Verfahren benutzt, da es bei den auftretenden Elementzahlen in der Größenordnung von 1500 bis 35000 mit Abstand das schnellste Sortierverfahren ist. Damit ergibt sich:

1. *berechne alle rt_i:=Raumtiefe(Element i) und setze in_i:=i (Indexfeld)*

Sortierverfahren:

Die Liste wird so umsortiert, daß alle Zahlen im Linken Teil größer als die Zahlen im rechten Teil sind (im rechten Teil stehen dann auch die Zahlen, die kleiner-gleich sind).
Das Verfahren wird mit dem linken und rechten Teil wiederholt, bis keine Umsortierung mehr nötig ist.

qsort(LINKS,RECHTS) (sortiert von LINKS bis RECHTS)

2. *i:=LINKS, j:=RECHTS*
3. *i<=j --> > Ende*
4. *setze x:=rt[(LINKS+RECHTS) div 2]*
5. *falls rt_i>x dann i:=i+1 und wiederhole ab 5.*
6. *falls rt_j<x dann j:=j-1 und wiederhole ab 6.*
7. *falls i<j dann:*
   vertausche rt_i mit rt_j und in_i mit in_j
   i:=i+1 und j:=j-1
8. *falls i>j wiederhole ab 5.*
9. *falls (j-LINKS)<(RECHTS-i) dann*
   qsort(LINKS,j) danach qsort(i,RECHTS), sonst
   qsort(i,RECHTS) danach qsort(LINKS,j)

10. *zeichne alle Flächenelemente nach der in in_i enthaltenen Reihenfolge*

Für dieses Verfahren ist es relativ wichtig, über den zur Verfügung stehenden Speicherplatz Bescheid zu wissen. Folgende Abschätzung hilft dabei:

Sei Rx die Anzahl der Rasterlinien in x Richtung
Ry -"- y Richtung, dann gilt

Hauptspeicherbedarf: Rx*Ry*4*10 Byte (10*10 -> 4000 Byte
90*90 -> 316.4 kByte)

Stacktiefe: $2*\log_2(Rx*Ry*4)*34$ Byte (90*90 -> 1019 Byte)

(Die Zahl 34 Byte resultiert aus den lokalen Variablen der Prozedur QSORT:
i, j, LINKS, RECHTS (je 4 Byte), einem Zeiger p und einer Rückkehradresse (4 Byte) sowie x (10 Byte).)
Man erkennt daran, daß auf MS-DOS Rechnern 90*90 Rasterlinien (bzw. allgemein 32400 Oberflächenelemente) die obere Grenze der Auflösbarkeit darstellen dürfte. Eine für professionelle Zwecke verwendbare Auflösung $\geq$ 200*200 (bzw. 160'000 bis 200'000 Elemente) würde damit mindestens 1.5 MByte Hauptspeicher erfordern.

3.4. Analglyphenmodell

Die unbedingten Vorraussetzungen für dieses Modell sind ein Farbmonitor und mindestens eine EGA-Karte (der VGA-Karte sollte in jedem Fall der Vorzug gegeben werden).
Zunächst muß man sich für seine Stereo-Brille eine geeignetes Farbpaar suchen. Folgende Farbpaare haben sich als günstig erwiesen:

	Hintergrund	links	beide	rechts
	0	1	5	4
*	5	6	0	1
*	7	6	8	3
	10	11	15	14
	13	12	8	9
*	14	13	9	10
*	15	13	9	11

(* bezeichnet für die beiliegende Brille besonders gut geeignete Stereo-Paare).

Das Erzeugen eines Bildes verläuft jetzt analog zur einfachen Darstellung, d.h.:

1. *Bildbereich bestimmen*
2. *Rasterlinie ablaufen, auf Sichtbarkeit testen*
3. *Abstand vom Beobachter bestimmen -> dx*
4. *Punkt (x'-dx/2,y') mit Farbe LINKS und*
   Punkt (x'+dx/2,y') mit Farbe RECHTS zeichnen
   (für eine bessere Variante sollte noch getestet werden, ob auf (x'±dx/2,y') schon ein Punkt des jeweils anderen Teilbildes gezeichnet wurde, wenn ja, dann ist der Punkt mit der Farbe für beide zu zeichnen)

3.5. Höhenlinienmodell

An dieser Stelle soll nur eine kompakte Formulierung zum Verfolgen der Höhenlinien in einer Richtung angegeben werden.

1. *laufe auf Linien der Art y:=ymin+k*hy-hy/2 die gesamte Fläche ab*
2. *berechne z1:=f(x,y) und z2:=f(x+step,y)*
3. *für alle Höhenstufen hindex=1..nh Höhenlinie der Stufe HINDEX zwischen z1 und z2 suchen:*

   *falls (z1-zmin-HINDEX*hz)*(z2-zmin-HINDEX*hz)<=0*

4. *Höhenlinie der Stufe HINDEX zwischen z1 und z2 gefunden, verfolgen in einer Richtung*
5. *setze:*
   *xv[1]:=x+(HINDEX*hz+zmin-z1)*step/(z2-z1)*
   xv[2]:=y
   *xv[3]:=HINDEX*hz+zmin*
6. *Schnittpunkt Kreis k(xv,step) suchen, dazu setze: u:=0*
7. *k1:=f(xv[1]+step*cos(u), xv[2]+step*sin(u))-zmin-HINDEX*hz*
   *k2:=f(xv[1]+step*cos(u-step), xv[2]+step*sin(u-step))-zmin-HINDEX*hz*
   *falls k1*k2≥0, dann u:=u-step*
   *wiederhole ab 7., solange k1*k2≥0*

8. *Schnittpunkt mit u bestimmt, interpolieren*

   *z_1:=f(xv[1]+step*cos(u),xv[2]+step*sin(u))*
   *z_2:=f(xv[1]+step*cos(u-step), xv[2]+step*sin(u-step))*
   falls |z_2-z_1|≥1e-2000,
   dann u:=step(hindex*hz+zmin-z_1)/(z_2-z_1)+u*

9. *Schnittpunkt:*

   *x'[1]:=xv[1]+cos(u)*step*
   *x'[2]:=xv[2]+sin(u)*step*

10. *Sichtbarkeitstest, Projektion und zeichnen*

11. *setzte xv:=x'*

12. *wiederholen ab 6., bis Richtungswechsel oder Rasterlinie erreicht oder Rand erreicht*

3.6. Die Unit "VISUAL"

Zur Anwendung kam die Programmiersprache PASCAL unter Verwendung des UNIT-Konzepts, das heißt:
Der Anwender hat ein Werkzeug in der Hand, mit dem er schnell und einfach Funktionen der Art z=f(x,y) betrachten kann. Er muß sich lediglich ein kleines Programm schreiben, in welchem er sein Funktion definiert und das zwei Schalter setzt (siehe auch 3.6.3.). Die Unit "VISUAL" hat nach außen nur eine sichtbare Prozedur und einige Datentypen.

Schalter: {$F+} - erlaubt Funktionsparameter
{$M 60000,0,655359} - stellt Stack und Heap bereit

3.6.1. Datentypen

STYLE_TYPE:=(RASTERED,HIGHLINED,FILLED)

Dieser Datentyp gibt die Art der Oberfläche an, dabei bedeutet:

RASTERED:	Rasterlinien
HIGHLINED:	Höhenlinien
FILLED:	Flächenfüllmodell

PLOT_TYPE=(NORMAL,COLORED,STEREO,SHADED)

Gibt die Art des Plots an:

NORMAL:	monochrome Darstellung
COLORED:	Höhenstufen werden farbig dargestellt (Landkartenmodell)
STEREO:	Analglyphenbild
SHADED:	Schattenwurf (Raytrace-Modell)
COLOR_SHADED:	Schattenwurf und Farbdarstellung

VISIBL_TYPE:=(HIDDEN,VECTORIAL,TEST_OFF)

Bezeichnet die Art des Sichtbarkeitstest:

HIDDEN:	Methode des gleitenden Horizonts
VECTORIAL:	Sehstrahl-Algorithmus
TEST_OFF:	Test ausgeschaltet

Der Sortier-Algorithmus ist hier nicht mit aufgeführt, da er nur in Verbindung mit dem Flächenfüllmodell und dort immer aktiv ist.

PROJEKTION_TYPE:=(PARALLEL,CENTRAL)

PARALLEL:	Parallelprojektion
CENTRAL:	Zentralprojektion

Für die Datenübergabe an die Plot-Prozedure ist weiterhin der Typ

```
FCT_3D=FUNCTION(X,Y:EXTENDED):EXTENDED sowie die Typen

P_LIGHT_LIST=^LIGHT_LIST

LIGHT_LIST=RECORD
             X,Y,Z:EXTENDED
             NEXT:P_LIGHT_LIST
           END
```

erforderlich.

3.6.2. Plot-Prozedur

```
Syntax:  PLOT_3D ( F:FCT_3D,XMIN,XMAX,YMIN,YMAX,STEP,RLX,RLY,
                   SURFACE_STYLE,PLOT_STYLE,VISIBL_STYLE,
```

X0X,X0Y,X0Z,U,V,PROJECTION_TYPE,PX,PY,PZ,P_LIGHT)

Dabei bedeuten:

F: zu zeichnende Funktion (:FCT_3D)

XMIN,XMAX,YMIN,YMAX: Definitionsbereich (:EXTENDED)
STEP: Schrittweite auf der Rasterlinie (muß auch bei Flächenfüllen mit angegeben werden, kann dort aber gleich 0 gesetzt werden)
RLX,RLY: Anzahl Rasterlinien in x bzw. y Richtung (:INTEGER)

SURFACE_STYLE: Oberflächentyp (:STYLE_TYPE)
PLOT_STYLE: Zeichenstil (:PLOT_TYPE)
VISIBL_STYLE: Sichtbarkeitstest (:VISIBL_TYPE)

X0X,X0Y,X0Z: Koordinaten des Ursprungs des lokalen Koordinatensystems (:EXTENDED)
U,V: Bestimmungswinkel der Projektionsebene (:EXTENDED)

PROJECTION_TYPE: Projektionsart (:PROJEKTION_TYPE)
PX,PY,PZ: Falls PROJECTION_TYPE=CENTRAL, dann Koordinaten des Beobachterstandortes,
Falls PROJECTION_TYPE=PARALLEL, dann Richtung des Projektionsvektors (:EXTENDED)

P_LIGHT: Zeiger auf eine Liste der Lichtquellen (P_LIGHT_LIST)

Die meisten Teiloperationen der PLOT_3D Prozedur können durch drücken der Taste <Esc> abgebrochen werden.

3.6.3. Aufruf

```
program test_3d_grafik;

{$F+,$M 60000,0,655359}

uses graph,visual;

function f(x,y:extended):exended;
begin
  f:=x*y;
end;

var   d,m  :integer;
      ercode:word;

begin
```

```
  { Grafikmodus initialisieren }
  d:=0;
  initgraph(d,m,'');
  ercode:=plot_3d(f,-10,10,-10,10,0.1,30,40,rastered,colored,
                  vectorial,0,0,0,45,45,central,100,-100,100,nil);
  outtextxy(0,0,error[ercode]);
  readkey;
  restorecrtmode;
end.
```

Dieses Programm stellt die Funktion f(x,y)=x*y über dem Intervall (-10,10)x(-10,10) mittels Zentralprojektion auf der Ebene durch (0,0,0) und dem Normalenvektor (1,-1,1) dar. Das Projektionszentrum (=Beobachterstandpunkt) liegt in (100,-100,100). Es werden 30*40 entsprechend der getroffenen Höhenstufen eingefärbte Rasterlinien dargestellt. Die Schrittweite auf einer Rasterlinie beträgt 0,1. Die Sichtbarkeit eines Punktes wird mit dem Sehstrahl-Algorithmus getestet. Zum Schluß wird eine Fehler-/Erfolgsmeldung ausgegeben. Der Zeiger auf die Lichtliste muß NIL gesetzt werden, wenn kein Beleuchtungsmodell verwendet wird.

3.6.4. Fehlerbehandlung

PLOT_3D liefert einen Fehlercode zurück, der aber ab Turbo-Pascal V 5.5 nicht mehr abgefragt werden braucht. Will man aber Informationen über den Fehlerstatus der letzten ausgeführten Operation haben, so signalisiert der rückgegebene Wert folgende Zustände:

(Mit ERROR[Fehlercode] folgt:)

0: 'no error'
1: 'general fault'
2: 'observer position in definition range'
3: 'observer position in screen'
4: 'projection line parallel to screen'
5: 'fill style does not allow stereo'
6: 'not enough memory'
7: 'aborted'

Fehlerbeseitigung:

1: alle Eingabedaten überprüfen, Fehler nicht direkt lokalisierbar
2: anderen Beobachterstandort wählen
3: anderen Beobachterstandort oder andere Bildebene wählen
4: anderen Projektionsstrahl oder andere Bildebene wählen
5: entweder Flächenfüllmodell oder Rasterlinien-/Höhenlinien-Modell wählen
6: geringere Anzahl Rasterlinien (bzw. Oberflächenelemente) wählen oder mehr Hauptspeicher zur Verfügung stellen
7: weißt nur auf einen Abbruch durch den Benutzer hin

4. Beispiel

Zur Präsentation ist ein kleines Programm verfügbar, das den Sinn des Unit-Konzepts demonstriert und erschöpfend alle Möglichkeiten der PLOT_3D Prozedur aufzeigt. Es wurden folgende, hier nur in PASCAL-Notation angeführte Funktionen implementiert:

Wenn nicht anders erklärt, so gilt r:=sqrt(x^2+y^2).

F1: *f1:=sin(x)+cos(y);*

F2: *f2:=20*exp(-r/10);*

F3: *f3:=(sin(x)+cos(y))*5*exp(-r/10);*

F4: *if r<=100 then f4:=sqrt(100-r) else f4:=0;*

F5: *f5:=(sin(x)*cos(y))*8*exp(-r/4);*

F6: *f6:=x*x/5-y*y/10;*

F7:
```
if r<=5 then f7:=sqrt(25-r*r) else
if x<-5 then f7:=sqr((x+5))/120*sqr(y) else
f7:=0;
```

F8:
```
h1:=-(x+10)*3/4+5+sin(x)+1;
h2:=h1-2;
if (y<=h1) and (y>=h2) then f8:=-sqrt(1-sqr(y-h2-1));
if (y>h1) and (x<=0) then f8:=x*x/100*(y-h1);
if (y>(x-10)*(x-10)*4/5-10)
then f8:=-sqrt(x+sqrt((y+10)*5/4)-10)*1.5;
h1:=sqrt(sqr(x+5)+sqr(y+5));
if h1<=2 then f8:=2-h1*h1/2;
```

F9: *f9:=exp(-r/5)*10*cos(r);*

Das Programm ist über eine Menü sehr einfach zu bedienen. Etwas räumliches Vorstellungsvermögen ist allerdings von Nöten, damit man z.B. weiß, wie die Bildebene liegt, wie der Projektionsstrahl verläuft ...

Funktion wählen:	Funktion auswählen
Definitionsbereich:	Definitionsbereich bestimmen
Projektionsparameter:	Lage der Bildebene, Anzahl Rasterlinien, Auflösung (Schrittweite)
Projektionsart:	Zentral bzw. Parallelprojektion & Parameter
Darstellungsart:	Oberflächentyp, Farbmodell und Sichtbarkeit
Funktion plotten:	Funktion mit eingestellten Daten zeichnen

Als Standardvorgaben sind gewählt:

Funktion 1, Definitionsbereich (-10,10) x (-10,10), 20*20 Rasterlinien, Schrittweite 0.1, Rasterlinienmodell, monochrom Darstellung, Parallelprojektion, Methode des gleitenden Horizonts, Bildebene liegt in (0,0,0) mit Normalenvektor (1,-1,1) (d.h. u=45^o und v=45^o), Projektionsstrahl hat die Richtung (-1,1,-1) (u_p=-45^o sowie v_p=45^o).