Zur Erinnerung: Parametrisierung von Dreiecken

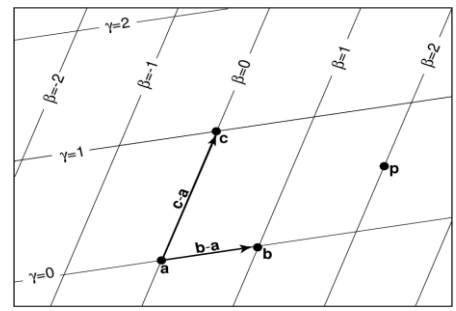
Interpolation über Dreiecke

Parametrisierung eines Dreiecks

$$\mathrm{p} = \mathrm{a} + \beta(\mathrm{b\text{-}a}) + \gamma(\mathrm{c\text{-}a})$$

$$\Leftrightarrow$$

$$\mathrm{p} = (1-\beta-\gamma)\mathrm{a} + \beta\mathrm{b} + \gamma\mathrm{c}$$



$$\Rightarrow$$

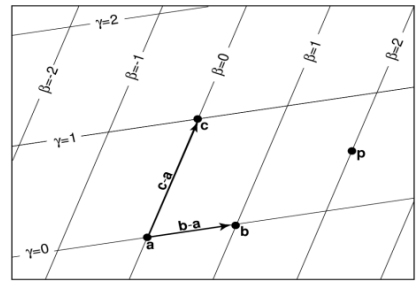
$$\mathrm{p}(\alpha,\beta,\gamma) = \alpha\mathrm{a} + \beta\mathrm{b} + \gamma\mathrm{c} \quad \text{mit} \quad \alpha+\beta+\gamma = 1$$

Baryzentrische Koordinaten

Baryzentrische Koordinaten

p liegt innerhalb des Dreiecks falls gilt:

$$0 < \alpha < 1$$
$$0 < \beta < 1$$
$$0 < \gamma < 1$$

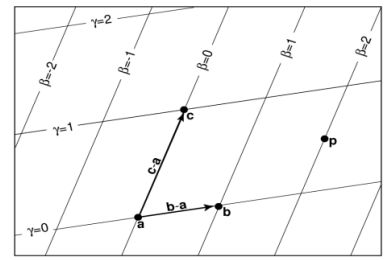


Baryzentrische Koordinaten

Berechne:

$$\mathrm{p} = \mathrm{a} + \beta(\mathrm{b\text{-}a}) + \gamma(\mathrm{c\text{-}a})$$

$$\begin{bmatrix} x_b - x_a & x_c - x_a \\ y_b - y_a & y_c - y_a \end{bmatrix} \begin{bmatrix} \beta \\ \gamma \end{bmatrix} = \begin{bmatrix} x_P - x_a \\ y_P - y_a \end{bmatrix}$$



$$\gamma(x_P, y_P) = \frac{(y_a - y_b)\,x_P + (x_b - x_a)\,y_P + x_a y_b - x_b y_a}{(y_a - y_b)\,x_c + (x_b - x_a)\,y_c + x_a y_b - x_b y_a}$$

$$\beta(x_P, y_P) = \frac{(y_a - y_c)\,x_P + (x_c - x_a)\,y_P + x_a y_c - x_c y_a}{(y_a - y_c)\,x_b + (x_c - x_a)\,y_b + x_a y_c - x_c y_a}$$

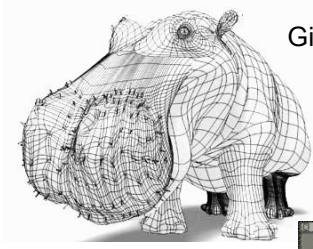
$$\alpha = 1 - \beta - \gamma$$

Texture mapping

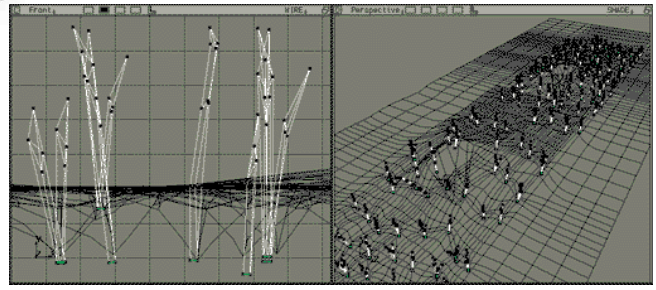
..... mehr Photorealismus

Ein Beispiel kommerzieller Computer Graphik (Kunst) von
Jeremy Birn http://www.3drender.com/jbirn/hippo/hairyhipponose.html

Wie viel Geometrie ist nötig?

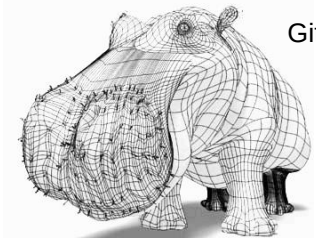


Gittermodell



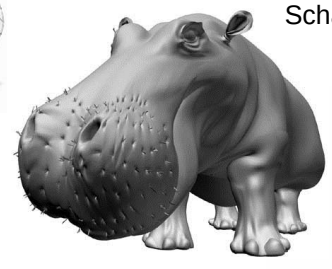
Mit Werkzeugen zur Geometriemodellierung (Splines, NURBS,...) lassen sich detailreiche Modelle erzeugen.

Wie viel Geometrie ist nötig?



Gittermodell

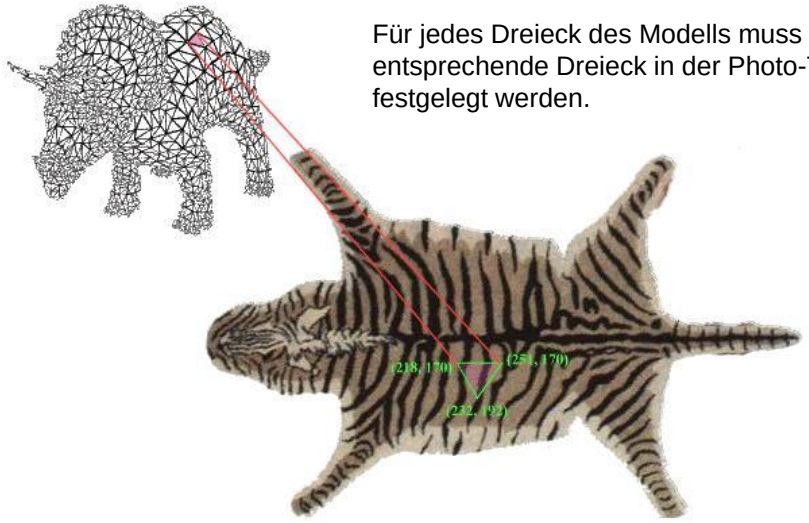
Gittermodell +
Schattierung



Viele Oberflächendetails lassen sich nicht allein durch Geometrie erzeugen!

Gittermodell +
Schattierung +
Texturemapping

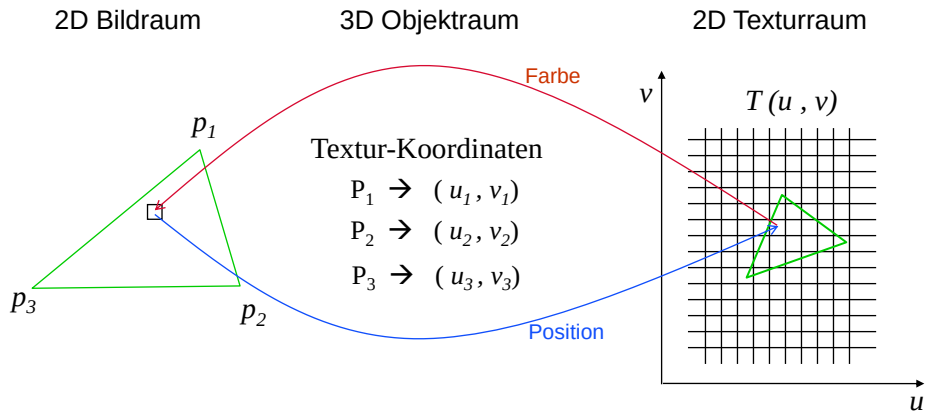
Photo-Texturen



Für jedes Dreieck des Modells muss das entsprechende Dreieck in der Photo-Textur festgelegt werden.

Zum Rastern werden dann die Pixelkoordinaten in der ‚Texture Map' interpoliert.

Abbilden der Textur in den Bildraum

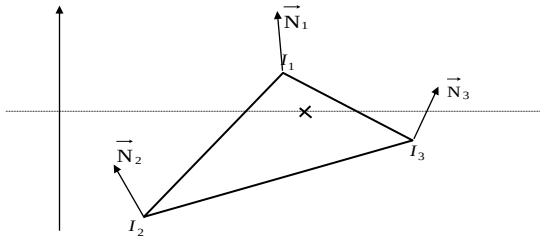


Zu jedem Bildpixel muss die entsprechende Position in den Texturkoordinaten (u,v) gefunden werden, wenn nötig muss zwischen den Texturwerten interpoliert werden.

Wie wurden beim ‚Shading' die Polygone gerastert ??

Wiederholung:
Rasterung von Polygonen mit Beleuchtungsmodell

Die Literatur unterscheidet die Algorithmen bezüglich ihrer Methodik beim Füllen von Polygonen.



Man unterscheidet:

1.) Flat Shading

2.) Gouraud Shading

3.) Phong Shading

im Detail

===>

Wiederholung: Gouraud Shading

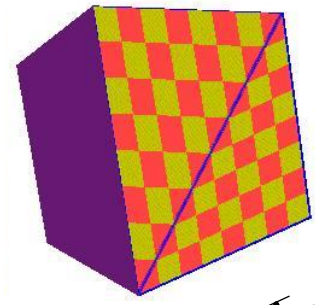
1. Berechne im Objektraum die Normalenvektoren in den Eckpunkten der Polygone;

for jedes Polygon do
  2. berechne im Objektraum das Beleuchtungsmodell an allen Eckpunkten des Polygons;
  3. projiziere das Polygon in die Bildebene;

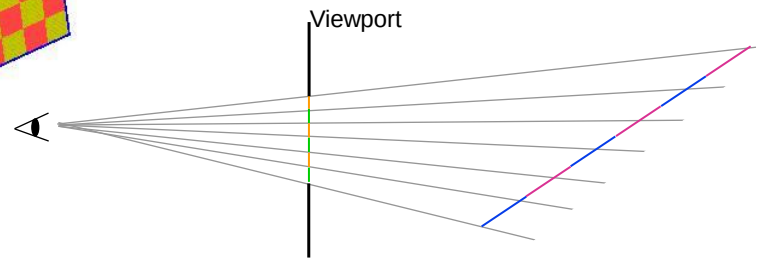
  for alle vom Polygon überdeckten Scanlinien do
    4. berechne den linear interpolierten Farbwert an der linken und rechten Kante des Polygons;

    for jedes Polygonpixel der Scanlinie do
      5. berechne den linear interpolierten Farbwert des Pixels.
    endfor
  endfor
endfor

Füllen der Polygone analog zu Gouraud Shading



Was geht hier schief?

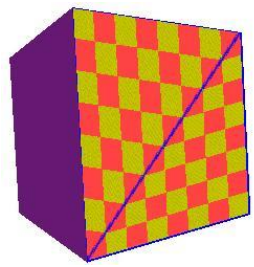


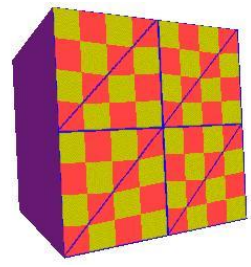
Gleichmäßige Schrittweiten in den Bildkoordinaten entsprechen **nicht** gleichmäßigen Schrittweiten auf dem Objekt !

Bei Gouraud Shading fällt der Fehler nur nicht sehr auf!

Auswege

1. Verkleinert man die Polygone werden auch die Fehler kleiner!





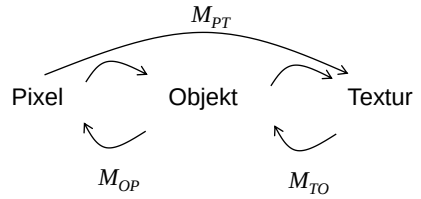
2. Korrekte perspektivische Rückabbildung,

   wird heute von fast allen Systemen verwendet !

zweistufige Abbildung:

Pixel → Objekt → Textur

Korrekte perspektivische Rückabbildung



$$M_{PT} = M_{TP}^{-1} = [M_{OP}\ M_{TO}]^{-1}$$

A)

$$[x,\ y,\ z,\ 1]^T = M_{TO}\ [u,\ v,\ 1]^T$$

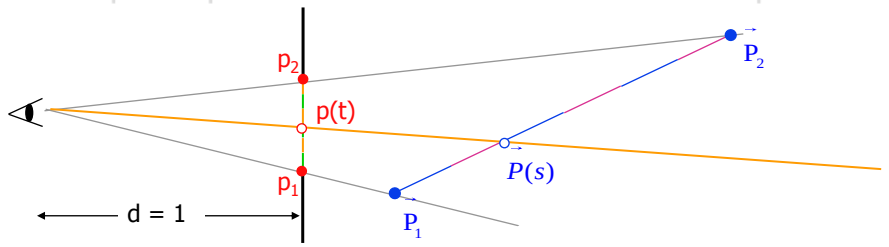
$$[x,\ y,\ z,\ 1]^T = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ 0 & 0 & 1 \end{bmatrix} [u,\ v,\ 1]^T \quad \rightarrow\rightarrow \quad M_{TP} = M_{OP} M_{TO} = \begin{bmatrix} a_{11} & a_{11} & a_{11} & 0 \\ a_{21} & a_{21} & a_{21} & 0 \\ 0 & 0 & 0 & 0 \\ \frac{a_{31}}{z_0} & \frac{a_{32}}{z_0} & \frac{a_{33}}{z_0} & \frac{1}{1} \end{bmatrix}$$

B)

$$M_{OP} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1/z_0 & 1 \end{bmatrix}$$

Durch invertieren von M_{TP} ergibt sich eine lineare Abbildung von den Pixelkoordinaten in die Textur.

Korrekte perspektivische lineare Interpolation



Vergleiche Interpolation im Bildraum

$$p(t) = p_1 + t(p_2 - p_1) = \frac{x_1}{z_1} + t\left(\frac{x_2}{z_2} - \frac{x_1}{z_1}\right)$$

mit Interpolation im Objektraum

$$\vec{P}(s) = \vec{P}_1 + s\left(\vec{P}_2 - \vec{P}_1\right)$$

$$\begin{bmatrix} x(s) \\ z(s) \end{bmatrix} = \begin{bmatrix} x_1 \\ z_1 \end{bmatrix} + s\left(\begin{bmatrix} x_2 \\ z_2 \end{bmatrix} - \begin{bmatrix} x_1 \\ z_1 \end{bmatrix}\right)$$

$$p(t) = \mathrm{P}_{\mathrm{per}}\left(\begin{bmatrix} x(s) \\ z(s) \end{bmatrix}\right) = \frac{x_1 + s(x_2 - x_1)}{z_1 + s(z_2 - z_1)}$$

Korrekte perspektivische Interpolation

Wie suchen die Abbildung von t nach s :

$$\frac{x_1}{z_1} + t\left(\frac{x_2}{z_2} - \frac{x_1}{z_1}\right) = \mathrm{P}_{\mathrm{per}}\begin{pmatrix}\lceil x(s)\rceil\\ \lceil z(s)\rceil\end{pmatrix} = \frac{x_1 + s(x_2 - x_1)}{z_1 + s(z_2 - z_1)}$$

löse nach s als Funktion von t auf:

$$s = \frac{t\, z_1}{z_2 + t(z_1 - z_2)}$$

Leider haben wir nach der Projektion keinen Zugriff auf z.

Von der perspektivischen Abbildung kennen wir jedoch
$w_1 = z_1 / d$ und $w_2 = z_2 / d$

$$s = \frac{t\, w_1}{w_2 + t(w_1 - w_2)} = \frac{t\, 1/w_2}{1/w_1 + t(1/w_2 - 1/w_1)}$$

Interpolation über Dreiecke

Bilineare Interpolation des projizierten Dreiecks ($\mathrm{p}_0, \mathrm{p}_1, \mathrm{p}_2$)

bestimme für jeden Pixel p die Baryzentrischen Bildkoordinaten α, β, γ

$$\mathrm{p}(\beta,\gamma) = (1-\beta-\gamma)\,p_0 + \beta\, p_1 + \gamma\, p_2$$

bestimme aus diesen Baryzentrischen Bildkoordinaten α, β, γ die Baryzentrischen Weltkoordinaten $\alpha_w, \beta_w, \gamma_w$

Vor der Projektion gilt:

$$\mathrm{p}(\beta,\gamma) = (1-\beta-\gamma)\frac{1}{w_0}\begin{bmatrix}x_0\\ y_0\\ z_0\\ w_0\end{bmatrix} + \beta\frac{1}{w_1}\begin{bmatrix}x_1\\ y_1\\ z_1\\ w_1\end{bmatrix} + \gamma\frac{1}{w_2}\begin{bmatrix}x_2\\ y_2\\ z_2\\ w_2\end{bmatrix} = \frac{1}{w}\begin{bmatrix}x\\ y\\ z\\ w\end{bmatrix}$$

Baryzentrischen Weltkoordinaten α_w, β_w, γ_w

1 / w ist linear in Bildschirmkoordinaten

$$\frac{1}{w} = \frac{1}{w_0} + \beta\left(\frac{1}{w_1} - \frac{1}{w_0}\right) + \gamma\left(\frac{1}{w_2} - \frac{1}{w_0}\right)$$

Alle in Weltkoordinaten linearen Variablen ***k*** sind als
k / w linear in Bildschirmkoordinaten.

$$p(\beta,\gamma) = (1-\beta-\gamma)\frac{1}{w_0}\begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ w_0 \end{bmatrix} + \beta\frac{1}{w_1}\begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ w_1 \end{bmatrix} + \gamma\frac{1}{w_2}\begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ w_2 \end{bmatrix} = \frac{1}{w}\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}$$

z.B. für β_{Welt} gilt: $\dfrac{\beta_{Welt}}{w} = \dfrac{0}{w_0} + \beta\left(\dfrac{1}{w_1} - \dfrac{0}{w_0}\right) + \gamma\left(\dfrac{0}{w_2} - \dfrac{0}{w_0}\right)$?

Baryzentrischen Weltkoordinaten α_w, β_w, γ_w

z.B. für β_{Welt} gilt:

$$\frac{\beta_{Welt}}{w} = \frac{\beta}{w_1} \quad (1)$$

$$\frac{1}{w} = \frac{1}{w_0} + \beta\left(\frac{1}{w_1} - \frac{1}{w_0}\right) + \gamma\left(\frac{1}{w_2} - \frac{1}{w_0}\right) \quad (2)$$

(1) durch (2)

$$\beta_{Welt} = \frac{\dfrac{\beta}{w_1}}{\dfrac{1}{w_0} + \beta\left(\dfrac{1}{w_1} - \dfrac{1}{w_0}\right) + \gamma\left(\dfrac{1}{w_2} - \dfrac{1}{w_0}\right)}$$

$$\beta_{Welt} = \frac{w_0 w_2 \beta}{w_1 w_2 + w_2\beta(w_0 - w_1) + w_1\gamma(w_0 - w_2)}$$

$$\gamma_{Welt} = \frac{w_0 w_1 \gamma}{w_1 w_2 + w_2\beta(w_0 - w_1) + w_1\gamma(w_0 - w_2)}$$

Perspektivisches Rastern eines Dreiecks

Compute bounds for $x = x_i / w_i$ and $y = y_i / w_i$

for all x do

  for all y do

    compute (α,β,γ) for (x, y)

    if (α in $[0,1]$, β in $[0,1]$, γ in $[0,1]$) then

$$d = w_1 w_2 + w_2 \beta (w_0 - w_1) + w_1 \gamma (w_0 - w_2)$$

$$\beta_{welt} = w_0 w_2 \beta / d$$

$$\gamma_{welt} = w_0 w_1 \gamma / d$$

$$\alpha_{welt} = 1 - \beta_{welt} - \gamma_{welt}$$

$$u = \alpha_{welt} u_o + \beta_{welt} u_1 + \gamma_{welt} u_2$$

$$v = \alpha_{welt} v_o + \beta_{welt} v_1 + \gamma_{welt} v_2$$

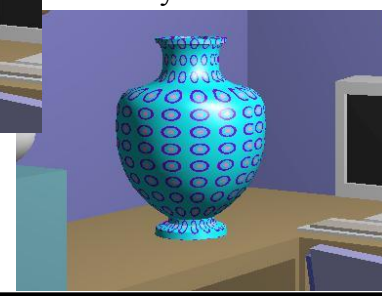
drawpixel(x, y) with color texture(u, v)

Verschiedene Projektionen einer Textur auf ein Objekt

Ebene

Kugel

Zylinder



Texturierung komplexer Objektoberflächen

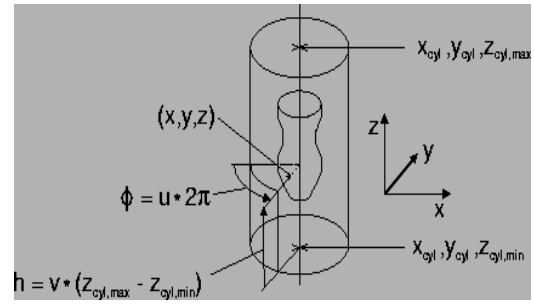
Idee: 1. Umhülle das Objekt mit einer parametrisierbaren Oberfläche.

2. Projiziere das Objekt auf diese Fläche

1. Zylinder

$$u = \frac{\pi + \operatorname{atan2}\left(y - y_{zyl}, x - x_{zyl}\right)}{2\pi}$$

$$v = \frac{z - z_{zyl,\min}}{z_{zyl,\max} - z_{zyl,\min}}$$

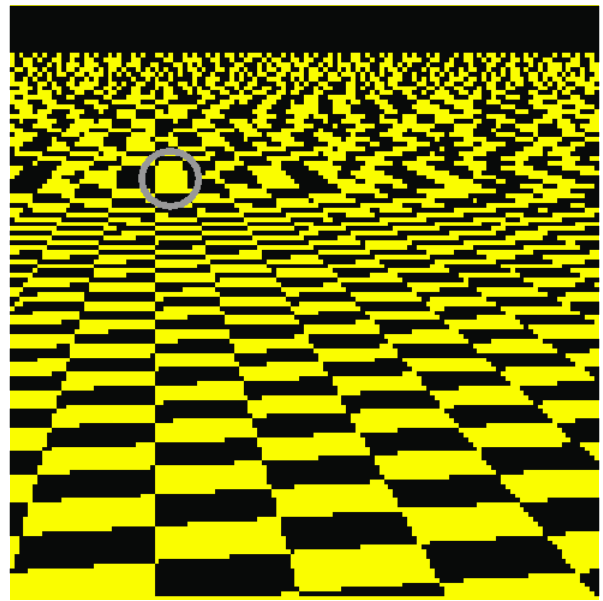


2. Kugel

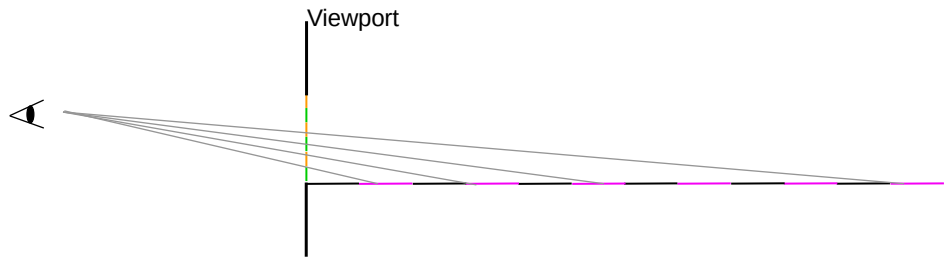
$$u = \frac{\pi + \arctan 2(y - y_{mittel}, x - x_{mittelb})}{2\pi}$$

$$v = \frac{\arctan 2\left(\sqrt{(x - x_{mittel})^2 + (y - y_{mittel})^2}, z - z_{zyl,\min}\right)}{\pi}$$

Schachbrettmuster: was geht hier schief?



Aliasing Effekte

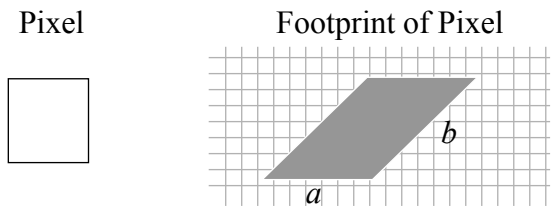


Wenn mehrere Texturwerte (Texel) auf einen Pixel abgebildet werden, hängt es oft vom Zufall ab welcher Wert dem Pixel zugeordnet wird!

Die Projektion eines einzelnen Pixels (Quadrats) auf die Texturebene ergibt ein Parallelogramm (Footprint), das von folgenden Vektoren aufgespannt wird:

$$\vec{r}_1 = \begin{bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial v}{\partial x} \end{bmatrix} \text{ und } \vec{r}_2 = \begin{bmatrix} \frac{\partial u}{\partial y} \\ \frac{\partial v}{\partial y} \end{bmatrix}$$

Vom „footprint“ zum Pixelwert



$$a = \sqrt{\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2}$$

$$b = \sqrt{\left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2}$$

Idealerweise würde die Projektion eines jeden Pixels in die Textur (footprint) bestimmt werden und dann alle Texel innerhalb des Footprints gemittelt.

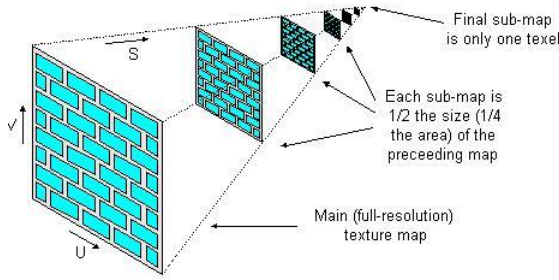
Dies ist sehr viel Rechenaufwand!!

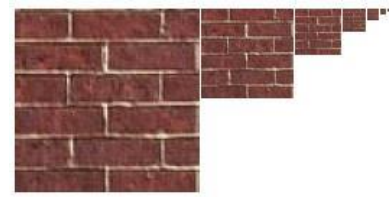
Im Allgemeinen wird die Mittelung der Texel vorberechnet und in Texturkarten (Mip-Maps) niederer Auflösung gespeichert.

Die meisten Texturierungsverfahren unterscheiden sich nur in der Approximation des Footprintinhalt.

Mip-Mapping

Beim Mip-mapping-Verfahren, wird jede Textur in eine Auflösungspyramide überführt, dabei werden immer 4 Pixel durch Mitteln auf ein Pixel der nächst niederen Auflösungsstufe reduziert.





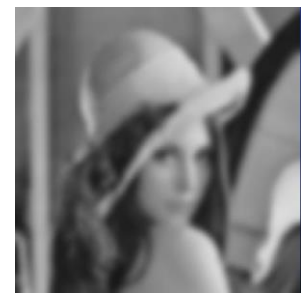
Der Speicherbedarf erhöht sich um ein Drittel!

Die Mittelung von 4 Pixel zu einem wird auch Box-Filterung genannt.

Box-Filterung ist nicht die Regel!

Zur Bildverkleinerung muss das Abtasttheorem herangezogen/optimiert werden (Shannon).

siehe Lehrbuch zur Bildverarbeitung!

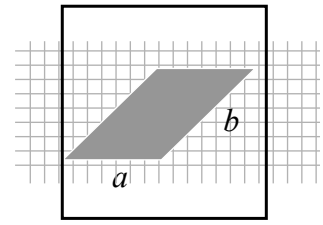


Approximation des Footprintinhalts

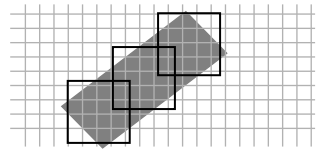
Klassisches Mip-Mapping: Der Footprint wird auf ein Texel (quadratisch) der Mip-Map der Kantenlänge 2^d gerundet.

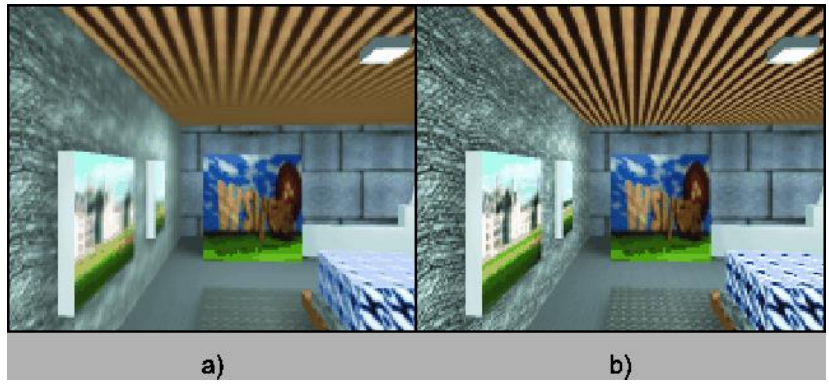
$$d = \log_2(\max(a,b))$$

Nachteil: Es werden viel zu viele Texel zusammengefasst.



Footprint Assembly: Die kürzeste Kantenlänge des Footprint bestimmt die Mip-Map-Stufe. Dann wird der Footprint durch mehrere Mip-Map-Texel überdeckt.





Standard MIP-Mapping

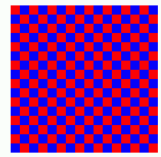
Foot-Print-Assembliy

Beleuchtungsmodelle und Texturemapping

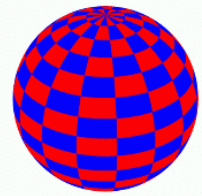
$$I_{total} = k_a I_{ambient} + \sum_{i=1}^{lights} I_i \left(k_d \left(\hat{N} \cdot \hat{L} \right) + k_s \left(\hat{V} \cdot \hat{R} \right)^{n_{shiney}} \right)$$

Phong's Illumination Model

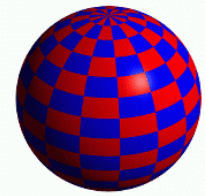
Constant Diffuse Color



Diffuse Texture Color



Texture used as Label



Texture used as Diffuse Color

Bump-, Normal- &Displacement-Mapping

Ziel: Auf eine gegebene grobe Geometrie sollen mehr geometrische Details durch ein Verfahren ähnlich zum „textrue mapping“ abgebildet werden!

Es existieren verschiedene Verfahren, leider ist die Literatur in den Bezeichnungen nicht immer einheitlich.

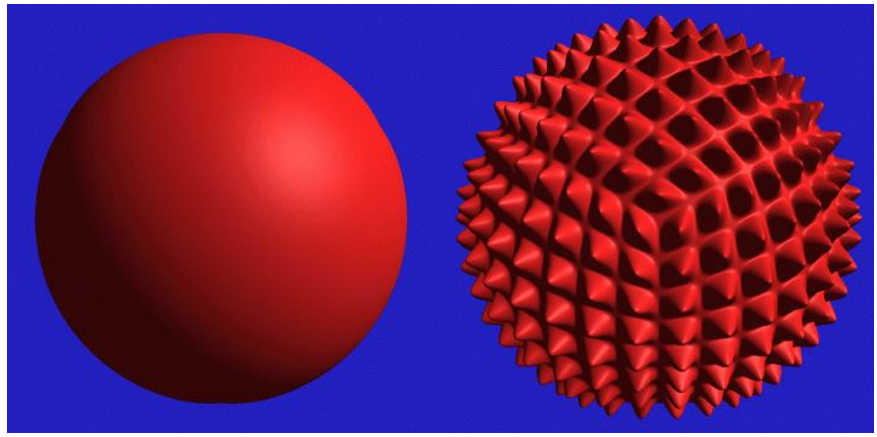
Zwei unterschiedliche Vorgehensweisen:

A) Es werden tatsächlich zusätzliche Vertices auf die grobe Geometrie abgebildet: „Displacementmapping“, „Subdevisiontechnique“: (i.a. sehr aufwändig)

B) Durch eine Manipulation der Normalenwerte kann über das Beleuchtungsmodell eine komplexere Geometrie simuliert werden. „Bumpmapping“, „Normalmaps“

Displacement Mapping

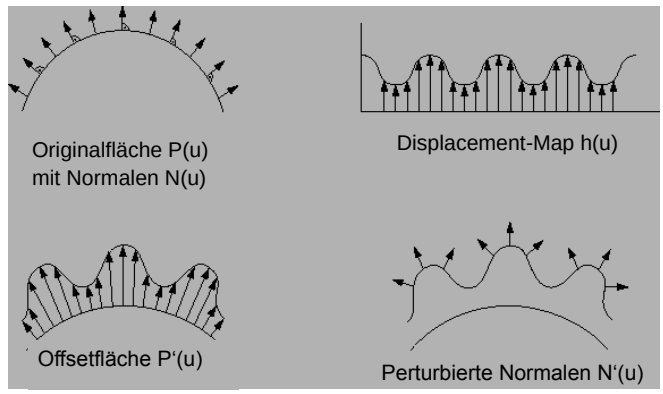
Hier wird eine Karte benutzt um die Oberflächenpunkte wirklich zu verschieben.



Die Geometrie muss vor der Prüfung auf Sichtbarkeit verändert werden.

Bump Mapping

Idee: Für das Beleuchtungsmodell soll die Oberflächennormale moduliert werden, was eigentlich nur durch eine Veränderung der Geometrie möglich ist.



Wie lässt sich die Modulation der Normalen auf **beliebige** Objekte abbilden ?

$$P'(u,v) = P(u,v) + h(u,v)N \qquad \text{und} \qquad N' = P_u' \times P_v'$$

Bump Mapping (2)

Idee: Im Beleuchtungsmodell sollen die Oberflächennormalen moduliert werden, was einer Veränderung der Geometrie entspricht!

$$P'_u = P_u + h_u n + h n_u \qquad P'_v = P_v + h_v n + h n_v \qquad \text{mit} \quad n = \frac{N}{|N|}$$

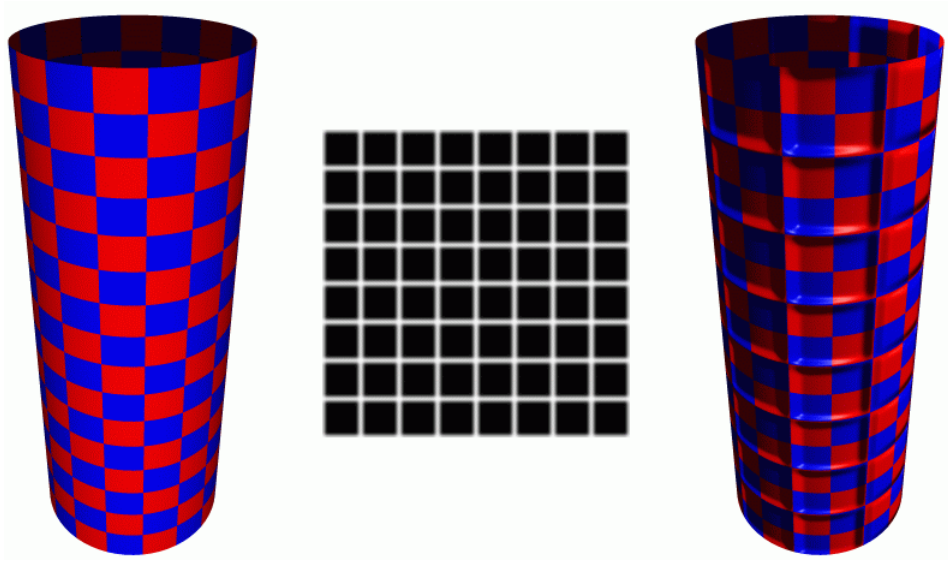
$$N' = P_u \times P_v + h_u (n \times P_v) + h_v (P_u \times n) + h_u h_v (n \times n)$$

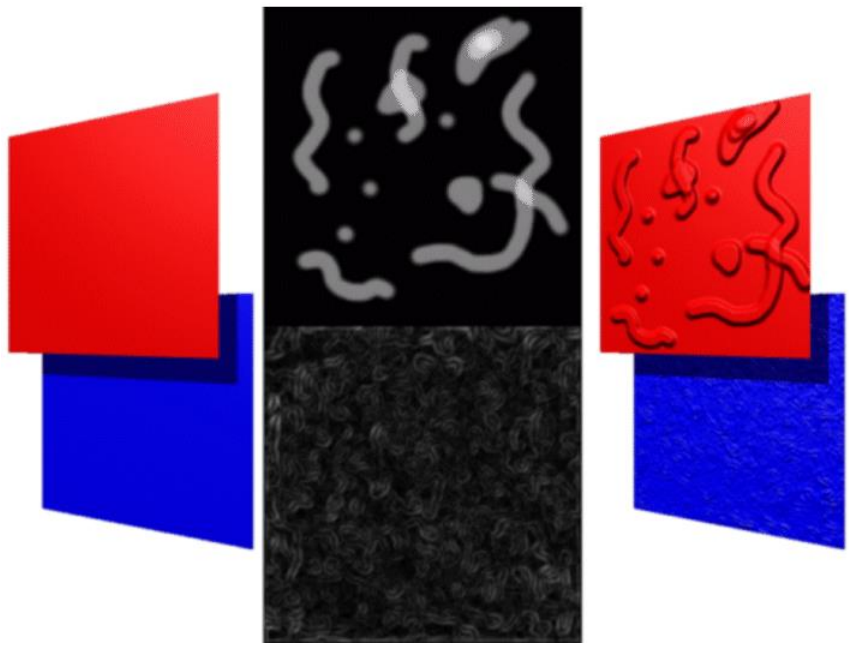
$$N' = N + h_u (n \times P_v) - h_v (n \times P_u) + 0$$

und normiere N´

$\Rightarrow$ Es reicht aus h_u, h_v in einer Bump-Map (entsprechend einerTexturkarte) abzuspeichern!

More Bump Mapping



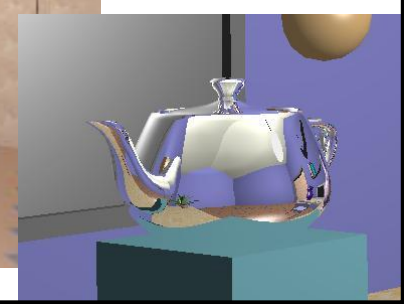


Reflektion- & Environment-Mapping

Environment Mapping Beispiel

Terminator II

Reflektion- & Environment-Mapping

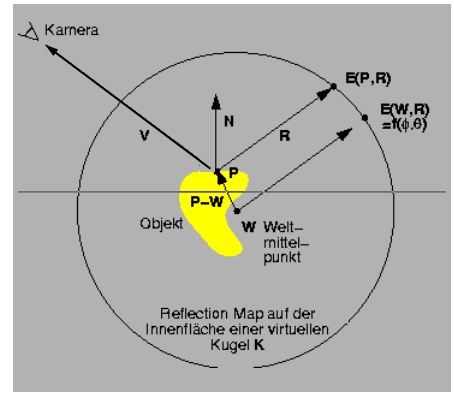


Reflection-Mapping

Die Helligkeitsverteilung der Umgebung soll auf ein Objekt abgebildet werden.

Zum Aufhellen oder Abschatten eines Objekts oder zur Simulation spiegelnder Oberflächen.

1. Möglichkeit: Ray Tracing ! Sehr aufwändig und muss für jeden Blickpunkt neu berechnet werden.
2. Reflection Map Die gesamte Szene wird durch eine einzige virtuelle Kugel ersetzt, und auf dieser als Texture-map mit Φ und Θ parametrisiert (Kugelradius >> Objekt). Die Richtung des auf der Oberfläche reflektierten Strahls ergibt die Position in der Reflectionmap.



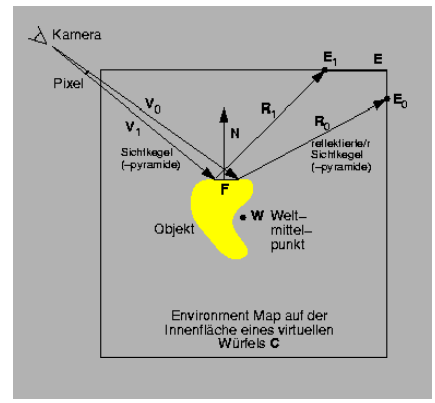
$$\vec{R} + \vec{V} = 2\left\langle \vec{V}\vec{N} \right\rangle \vec{N}$$

$$\Rightarrow \quad \vec{R} = \vec{V} - 2\left\langle \vec{V}\vec{N} \right\rangle \vec{N}$$

$$L_{out} = L_{amb} + L_{diff} + L_{spec} + L_{refl.map}(\Phi, \Theta)$$

Environment-Mapping

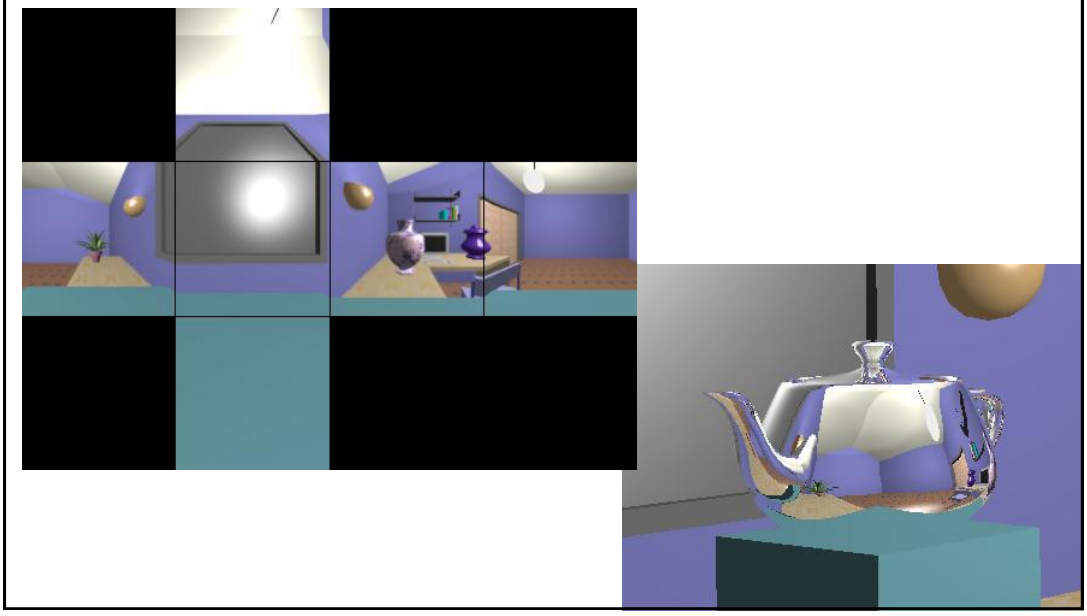
Anstatt auf eine Kugel abzubilden, wird häufig auf einen Würfel projiziert, dies hat den Vorteil, dass leichter über einen Bereich zu interpolieren ist. Somit kann über den Sichtbereich eines Pixels besser integriert werden.



Reflection- und Environmentmapping sind immer nur ein schwacher Ersatz für Raytracing.

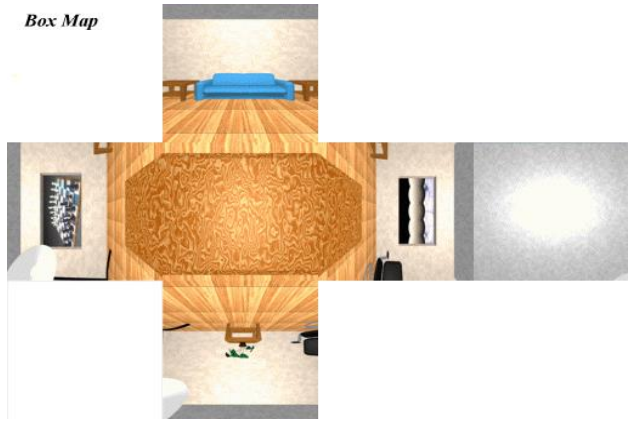
1. Es wird immer nur eine Reflektion berücksichtigt.
2. Annahmen der planaren Umgebung führt zu Verzerrungen.

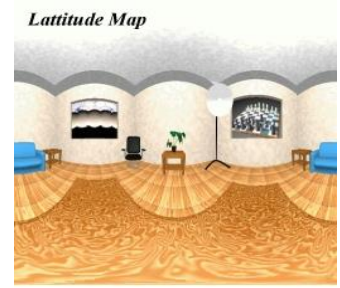
Environment-Mapping



Environment Maps

Welche graphische Repräsentation ?





3D Texturen

3D Texturen

1. Schritt: Abbilden der 3D Objektoberfläche in 3D Texturkoordinaten.

$$(x,y,z) \xrightarrow{F_{inv\_map}} (u,v,w)$$

2. Schritt: Berechnung der eigentlichen Farbwerte.

$$(u,v,w) \xrightarrow{C_{Textur}} (r,g,b)$$

Die Objektgeometrie wird aus dem dreidimensionalen Texturvolumen „herausgeschnitten“.

$C_{3D-Textur}$ ist im Allgemeinen eine funktionale Beschreibung und keine gemessene diskrete Funktion.

Holz als Beispiel

$F_{inv\_map} =$ *Identität*

Holz bestehe aus Zylindern mit Radius R entlang der Längsachse w.

$$R_1 = \sqrt{u^2 + v^2}$$

Da Holz nicht homogen wächst, wird der Radius periodisch moduliert .

$$R_2 = R_1 + k\sin(\pi R_1)$$

Kreisrunde Zylinder sind unnatürlich: $R_3 = R_2 + a\sin(\Theta)$

Ein Jahresring habe die Dicke $d = 0.3$ $\rightarrow$ $d = (R_3 \bmod 0.3)/0.3$

Farbe variiert nicht linear zwischen hell und dunkel:

$$(r,g,b) = \sqrt{d}\,(r_1,g_1,b_1) + \left(1-\sqrt{d}\right)(r_2,g_2,b_2)$$

3D Texturen: gibt es ein Prinzip?

1. Schritt: Moduliere Oberfläche mit einer Dichtefunktion $D(x,y,z)$.

2. Schritt: Moduliere $D(x,y,z)$ mit verschiedenen Stör- oder Rauschfunktionen.

$$H(D(X),X) = f_n(..., f_2(f_1(f_0(D(X)))))$$

Diese Vorgehensweise wird auch mit Hypertextur bezeichnet. (*Perlin 85)*

Wie simuliert man das Rauschen?
Es soll einerseits den Zufall simulieren und dennoch stetig sein C^0, C^1...

An den diskreten Gitterpunkten des Volumens werden Zufallszahlen ermittelt. Zwischen diesen wird dann trilinear interpoliert.