

Vektorgeometrie

Operationen

Kreuzprodukt

$$\vec{a} \times \vec{b} = \begin{pmatrix} a_2b_3 - a_3b_2 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{pmatrix}$$

Transponieren

$$\begin{pmatrix} a_1 \\ a_2 \\ \dots \end{pmatrix}^T = (a_1, a_2, \dots)$$

Euklidische Norm (Länge)

$$|\vec{a}| = \sqrt{a_1^2 + a_2^2 + \dots}$$

Normalisierung

$$\hat{a} = \frac{1}{|\vec{a}|} \cdot \vec{a}$$

Skalarprodukt

$$\vec{a} \circ \vec{b} = \sum_i (a_i \cdot b_i) = |\vec{a}| \cdot |\vec{b}| \cdot \cos \alpha$$

Fläche (Parallelogramm)

$$|\vec{a} \times \vec{b}|$$

Fläche (Dreieck)

$$|\vec{a} \times \vec{b}| \cdot \frac{1}{2}$$

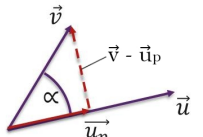
Gleichungssysteme

$$A\vec{x} + \vec{b} \Leftrightarrow \begin{matrix} a_{11}x_1 + a_{12}x_2 = b_1 \\ a_{21}x_1 + a_{22}x_2 = b_2 \end{matrix}$$

$$[A | \vec{b}]: \left[\begin{array}{cc|c} 1 & 4 & 2 \\ 2 & 9 & 5 \end{array} \right] \Rightarrow \left[\begin{array}{cc|c} 1 & 0 & -2 \\ 0 & 1 & 1 \end{array} \right]$$

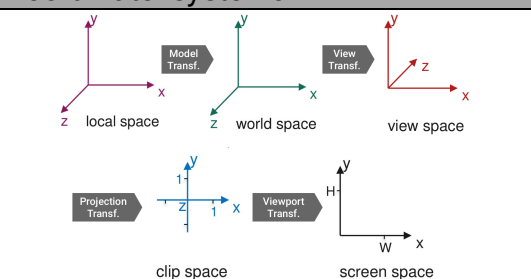
Orthogonale Projektion

$$\vec{u}_p = \left(\frac{\vec{u} \circ \vec{v}}{|\vec{u}|^2} \right) \cdot \vec{u}$$
$$= |\vec{v}| \cdot \cos \alpha \cdot \hat{u}$$



3D Geometrien

Koordinatensysteme



Transformation

$$T(\vec{x}) = \vec{x} + \vec{d}$$

$$S(\vec{x}) = s \cdot \vec{x}$$

$$R_\theta(\vec{x}) = \begin{pmatrix} x \cdot \cos \theta - y \cdot \sin \theta \\ x \cdot \sin \theta + y \cdot \cos \theta \end{pmatrix}$$

$$s \cdot (\vec{d} + \vec{x}) \neq (s \cdot \vec{d}) + \vec{x}$$

$$P' = P - E$$

Homogene Koordinaten

$$\begin{pmatrix} 1 & d_1 \\ 1 & d_2 \\ & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x + d_1 \\ y + d_2 \\ 1 \end{pmatrix}$$

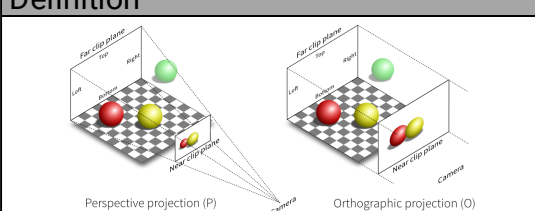
$$\begin{pmatrix} s_1 & & \\ & s_2 & \\ & & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} s_1 \cdot x \\ s_2 \cdot y \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \\ & & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x \cos \theta - y \sin \theta \\ x \sin \theta + y \cos \theta \\ 1 \end{pmatrix}$$

$$M_R \cdot (M_S \cdot \vec{x}) = (M_R \cdot M_S) \cdot \vec{x}$$

Projektionen

Definition



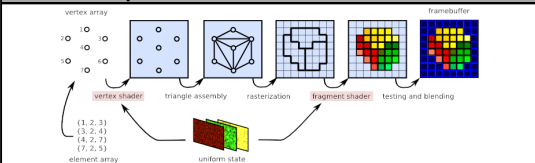
Berechnung

$$c_x = \frac{e_x z - e_z x}{z - e_z}$$
$$c_x = x$$

$$c_y = \frac{e_y z - e_z y}{z - e_z}$$
$$c_y = y$$

GPU-Berechnung

Grafik-Pipeline



GLSL Programmiermodell

- in: Aus vorheriger Stage
- out: An nächste Stage
- uniform: Vom Host an Primitiven

```
in vec3 positionIn;
in vec3 normalIn;
out vec3 normal;

// Transformations to Clip-Space
uniform mat4 model;
uniform mat4 view;
uniform mat4 projection;

void main() {
    gl_Position = vec4(positionIn, 1.0);
    normal = vec4(normalIn, 1.0) * m
}
```

Beleuchtung & Texturen

Oberflächennormale

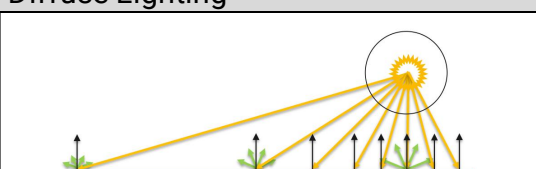
$$N_{V_1} = (V_2 - V_1) \times (V_3 - V_1)$$

Beleuchtungsmodelle

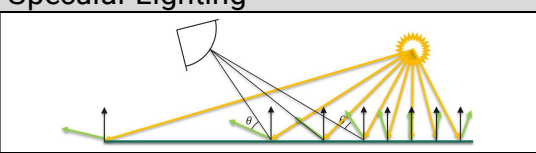
Ambient Lighting



Diffuse Lighting



Specular Lighting



Berechnungen

$$e_\alpha = \cos \alpha \cdot e = \frac{\vec{n} \circ \vec{p}}{|\vec{n}| \cdot |\vec{p}|} \cdot e$$

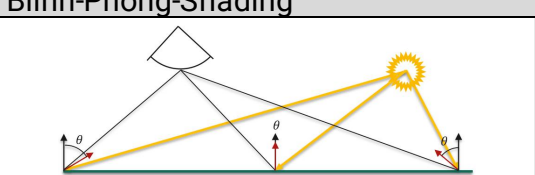
$$e_F = \max(e_\alpha, 0) \cdot F$$

Kombinationsmodelle

Phong-Shading

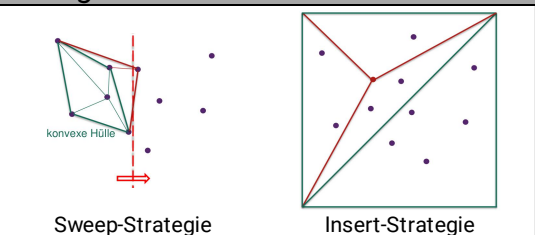
$$C_{\text{Total}} = \frac{1}{3} \cdot (C_{\text{Ambient}} + C_{\text{Diffuse}} + C_{\text{Specular}})$$

Blinn-Phong-Shading



Komplexe Oberflächen

Triangulation



Sweep-Strategie

1. Laufe von links nach rechts.
2. Für jeden Punkt:
 - a. Zeichne eine Linie zu den 2 vorherigen Punkten, für die gilt:
 - Keine Dellen entstehen
 - Keine Überschneidungen
 - b. Verbinde nun alle weiteren Punkte innerhalb dieser Form.
3. Wiederhole, bis zum Ende.

Insert-Strategie

1. Zeichne 2 Anfangsdreiecke um alle Punkte.
2. Für alle Punkte (zufällige Wahl):
 - a. Bestimme umfassende Dreieck.
 - b. Unterteile dieses Dreieck in 3 weitere Dreiecke. D.h. Verbinde alle Eckpunkte mit dem Punkt.
3. Wiederhole, bis zum Ende.
- 4.

Entferne nun alle künstlichen Anfangspunkte und die damit verbundenen Dreiecke.

Approximationen

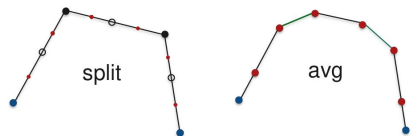
Marching Squares Algorithmus

1. Gitter über die Daten legen.
2. Betrachtungshöhe festlegen.
3. Für alle Quadrate im Gitter:
 - a. Eckpunkte beachten.
 - b. Nach Schema Linien einzeichnen.
4. Wiederhole, bis zum Ende.

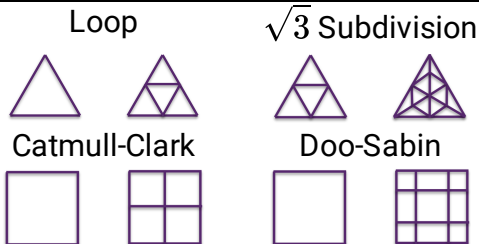
Subdivision Surfaces

Curves: Chaikin's Algorithmus

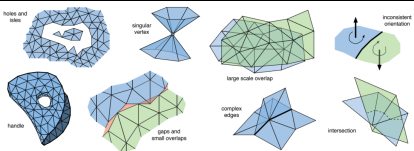
1. Beginne mit einer Kurve
2. Markiere die Anfangspunkte (Blau)
3. Setze in der Mitte von **allen** Strecken einen neuen Punkt (Schwarz ohne Füllung)
4. Setze nun in der Mitte von allen **neuen** Strecken einen Punkt (Rot)
5. Streiche nun alle schwarzen Punkte und verbinde die Roten und Blauen.
6. Wiederhole, solange wie gewünscht.



Surfaces: Algorithmen



Korrektur & Optimierung



Mesh Reduktion / Remeshing

Vertex Clustering

1. Wähle ein Grösse ϵ (Toleranz)
2. Teile den Raum in Quadrate dieser Grösse
3. Berechne pro Quadrat **einen** repräsentativen Eckpunkt (z.B. Mittelpunkt aller Punkte)
4. Lösche die originalen Punkte und ersetze sie durch den neuen Eckpunkt.

Rasterisierung & Sichtbarkeit

Rasterisierung

Aliasing

Zeichne ausschliesslich die Pixel eines Dreiecks, für die gilt:

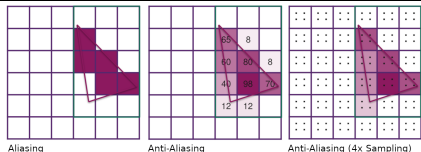
- Das Zentrum liegt **in** dem Dreieck.
- Das Zentrum liegt **auf** der oberen oder linken Seite des Dreiecks.
 - Bei **Eckpunkten** muss das Zentrum auf der oberen **und** linken Seite liegen.
 - Zwei **linke** Seiten sind auch gültig.

Bresenham Linien-Algorithmus

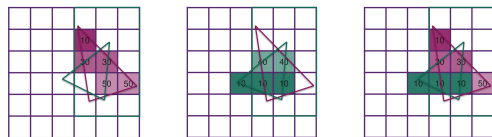
1. Berechne $\Delta x = x_{\text{Ende}} - x_{\text{Start}}$
2. Berechne $\Delta y = y_{\text{Ende}} - y_{\text{Start}}$
3. Berechne $m = \Delta y / \Delta x$
4. Wenn $\Delta x \geq \Delta y$ dann mit $i = 0$:

- a. $x_i = x_{\text{Start}} + i$
- b. $y_i = y_{\text{Start}} + \lfloor m \cdot i + 0.5 \rfloor$
- c. Zeichne den Pixel $P(x_i, y_i)$
- d. $i \leftarrow i + 1$

Anti-Aliasing

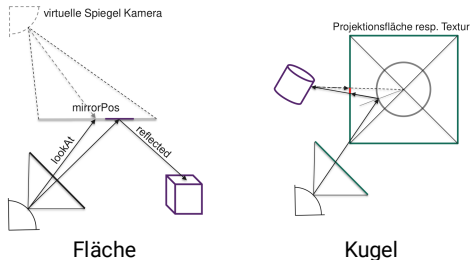


Sichtbarkeit



Spiegelungen & Schatten

Spiegelungen



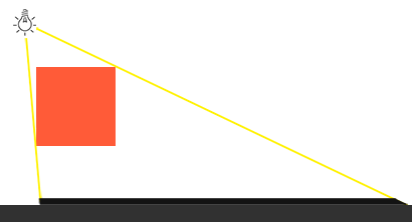
Berechnungen

$$\vec{v} = E - M \quad \vec{d} = \left(\frac{\vec{n} \circ \vec{v}}{|\vec{n}|^2} \right) \cdot \vec{n}$$

$$E' = E - 2 \cdot \vec{d}$$

Schatten

Shadow Mapping



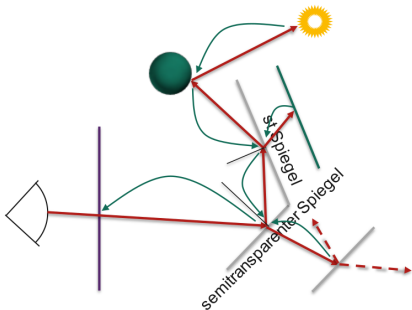
Depth-Map

Visualisierung des Z-Buffers.

- Schwarz: $Z_O = 0$ (Nahe)
- Weiss: $Z_O = \infty$ (Weit weg)

Ray-Tracing

Grundprinzip



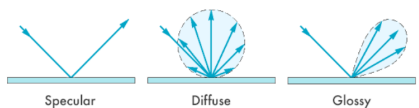
«Whitted Ray Tracing»-Algorithmus

1. Spiegelnd: Reflexionsstrahl
2. Durchsichtig: Lichtstrahl

Diffuse: Strahlen zu Lichtquellen

- Schatten: Kein Betrag
- Belichtet: Betrag vom remittierten Licht berechnen (Phong-Shading)

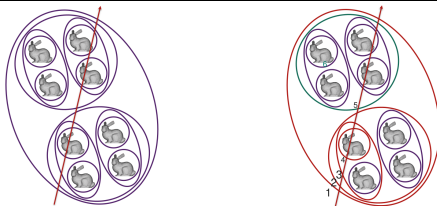
Advanced Ray-Tracing



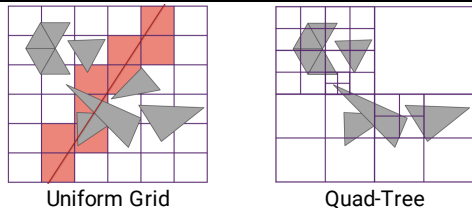
Acceleration Structures

- **Problem:** Objekte in der Szene haben sehr komplexe Oberflächen.
- **Lösung:** Umschliesse alle Objekte in einfache Bounding Volumes.

Elementteild



Raumteild



Animationen

Techniken

- Vom Künstler erstellt (Key Frames)
- Datengetrieben (Motion Capture)
- Prozedural (Simulation & Calculation)

Key Frames & Tweening

Beschreibt ein Verfahren, bei dem wichtige Animationspunkte (Key Frames) **manuel** erstellt und die Übergänge dazwischen interpoliert werden.