

Dreiecksnetze

Einführung in die Computergrafik
(für Augmented Reality)

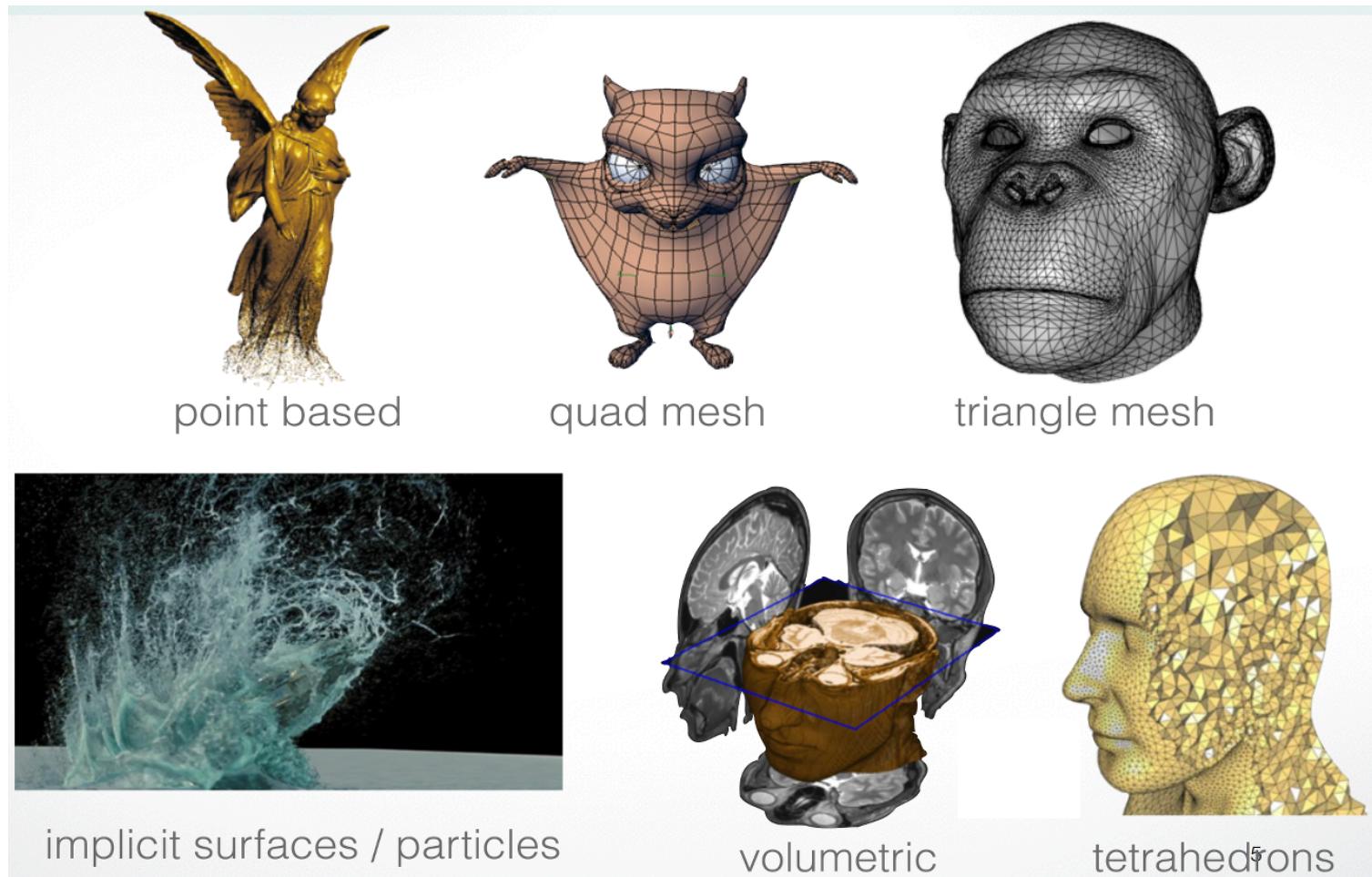
Wiederholung

- Organisation
- Grundlagen: Vektoren
- 3D-Raum
- Dreiecksnetze
- OpenGL ES
- Rendering in OpenGL
- Szenengraph
- Augmented Reality

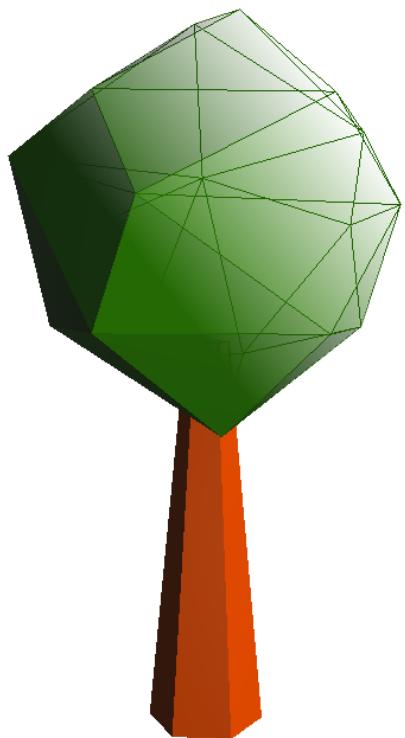
Agenda

- Dreiecksnetze
- Normalen
- Lokale Beleuchtungsmodelle
- Texturen
- Nachbarschaftsdatenstrukturen
- Glättung
- Grundlagen: Matrizen

Oberflächenrepräsentationen



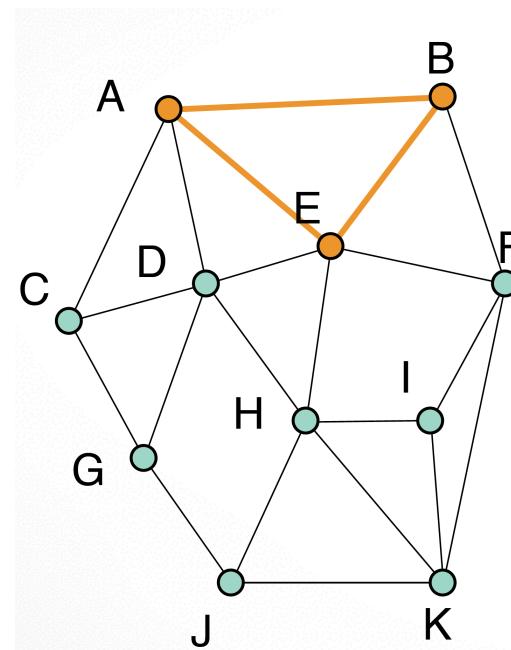
Quelle: Mario Botsch, Leif Kobbelt, Mark Pauly, Pierre Alliez, Bruno Levi: Polygon Mesh Processing, A. K. Peters, 2010



Dreiecksnetze

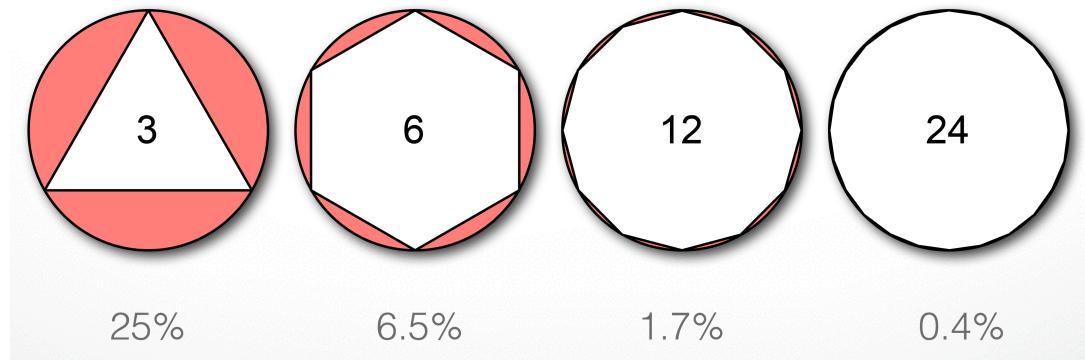
Polygonale Netze

- Struktur: Graph
 - Graph $\{V, E\}$
 - Knoten: Vertices $V = \{A, B, C, D, \dots\}$
 - Kanten $E = \{(AB), (AE), \dots\}$
 - Facetten $F = \{(AEB), (EFB), \dots\}$
- Grad oder Valenz eines Vertex
 - Incidente Kanten
 - Beispiele
 - $\text{Grad}(A) = 4$
 - $\text{Grad}(E) = 5$

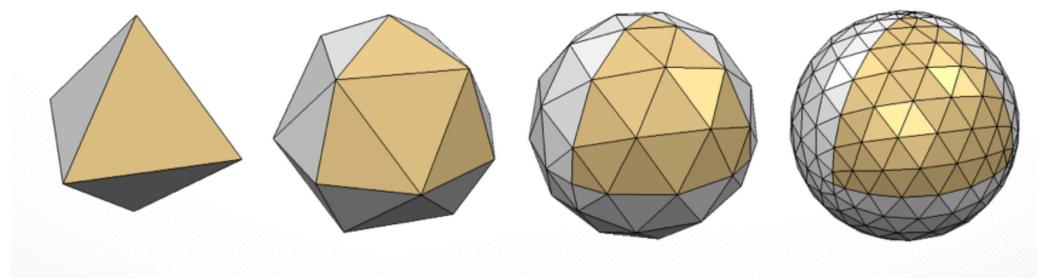


Warum Polygone

- Gute Annäherungseigenschaften ($O(h^2)$)



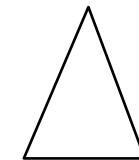
- Fehler invers proportional zu Anzahl der Facetten (hier: Dreiecke)



Source: Mario Botsch, Michael Sernat, Leif Kobbelt: Phong Splatting, Symposium on Point-Based Graphics 2004, 25-32

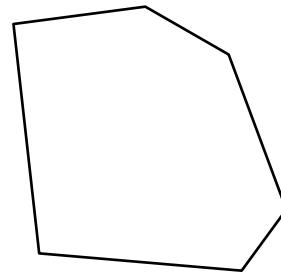
Repräsentation

- definiert durch Vertices V und Facetten F
- jeder Vertex $v \in V$ hat Position $v.p \in R^3$
- jede Facette $f \in F$ besteht aus einer Liste von Vertices
- alle Vertices einer Facette müssen planar sein
- meist: Dreiecke oder Quads
 - Mindestanforderung: konvexas Polygon

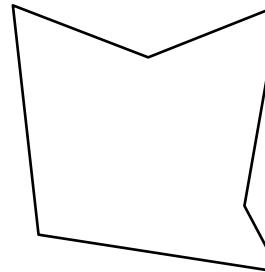


Dreieck

Quad



konvex



konkav

Datenstruktur

Triangles		
x_{11} y_{11} z_{11}	x_{12} y_{12} z_{12}	x_{13} y_{13} z_{13}
x_{21} y_{21} z_{21}	x_{22} y_{22} z_{22}	x_{23} y_{23} z_{23}
...
...
...
x_{F1} y_{F1} z_{F1}	x_{F2} y_{F2} z_{F2}	x_{F3} y_{F3} z_{F3}

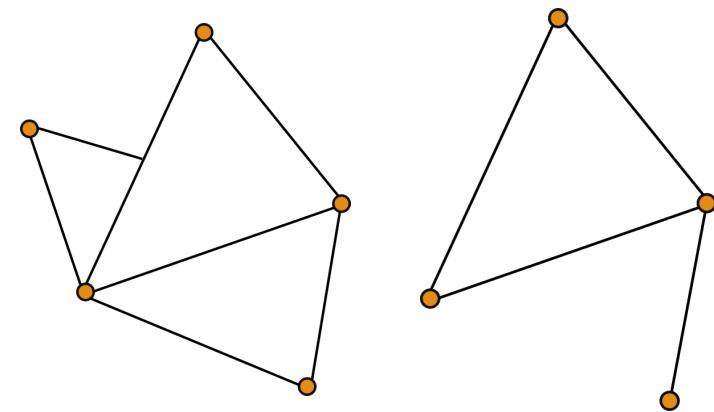
Facettenliste

Vertices	Triangles
x_1 y_1 z_1	i_{11} i_{12} i_{13}
...	...
x_v y_v z_v	...
...	...
...	...
...	...
i_{F1} i_{F2} i_{F3}	

Indizierte Facettenliste
(beispielsweise im Wavefront OBJ
Datenformat)

Wohlgeformtheit

- wohlgeformte polygonale Netze:
 - 2 Facetten an jeder Kante
 - Rand: nur eine Facette
 - 2 Vertices an jeder Facette
 - jede Facette durch 3 Kanten umrandet
 - Kanten/Facetten erzeugen Schirm um gemeinsamen Vertex



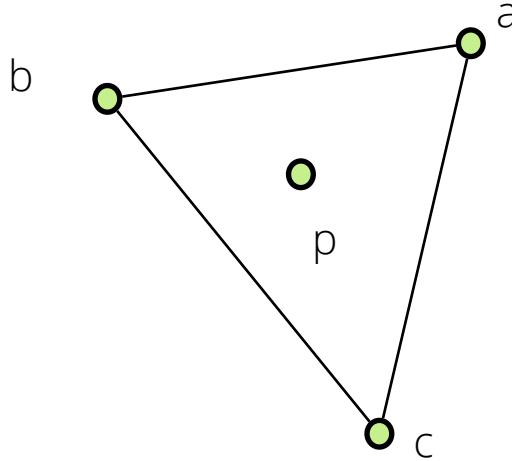
nicht wohlgeformt

Dreiecksnetze

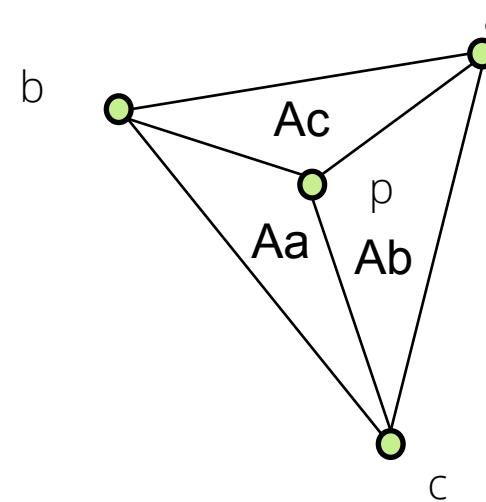
- meist: Dreiecke
 - jeder Polygon kann trianguliert werden (siehe Ear-Cutting-Algorithmus, [1])
 - Facetten sind automatisch planar
- durchschnittliche Valenz in Dreiecksnetz: 6
- üblicherweise zusätzliche Vertex-Attribute
 - Farbe
 - ...

Baryzentrische Koordinaten

- siehe [2] oder [Weisstein, Eric W. "Barycentric Coordinates." From MathWorld--A Wolfram Web Resource.](http://mathworld.wolfram.com/BarycentricCoordinates.html) <http://mathworld.wolfram.com/BarycentricCoordinates.html>



$$\mathbf{p} = \alpha \mathbf{a} + \beta \mathbf{b} + \gamma \mathbf{c}$$



Punkt innerhalb des Dreiecks:
 $\alpha + \beta + \gamma = 1$
 $0 \leq \alpha, \beta, \gamma \leq 1$

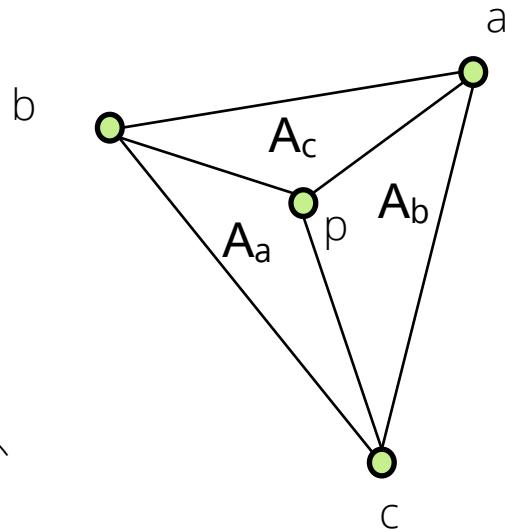
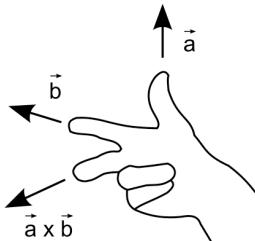
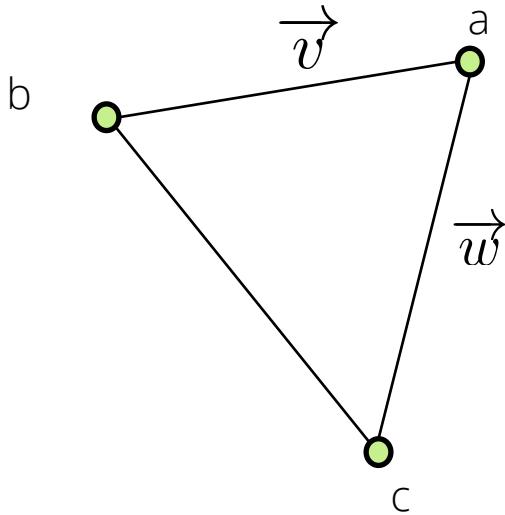
$$\alpha = \frac{A_a}{A}$$

$$\beta = \frac{A_b}{A}$$

$$\gamma = \frac{A_c}{A}$$

Fläche

- berechne A_a , A_b und A_c
- Kreuzprodukt
- Reihenfolge relevant

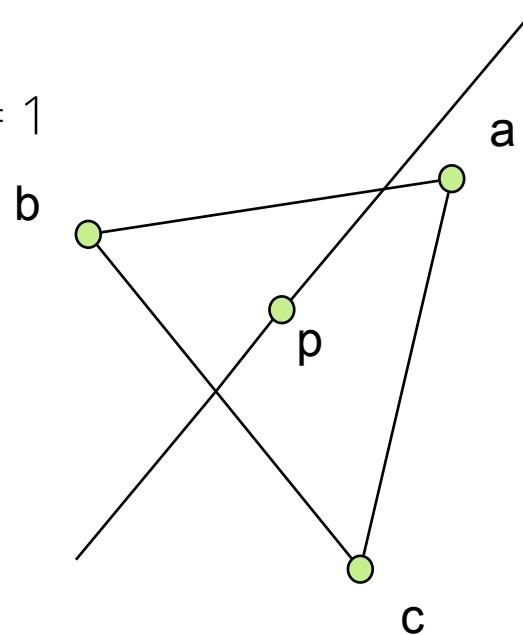


$$|\vec{v} \times \vec{w}| = A_{\text{Parallelogramm}} = 2A_{\text{Dreieck}}$$

Bildquelle (rechtshändiges Koordinatensystem): Ladyt
in der Wikipedia auf Deutsch - Eigenes Werk,
Gemeinfrei, <https://commons.wikimedia.org/w/index.php?curid=17618148>

Anwendungsbeispiel

- Strahl-Dreiecks-Schnitt
 - Ebene aus (a, b, c) , Normalform (siehe Skript)
 - Schnitt: $p = \text{Ebene} \cap \text{Strahl}$
 - Berechnet Baryzentrische Koordinaten von p
 - p in Dreieck a, b, c :
 - if $0 \leq \alpha, \beta, \gamma \leq 1$ and $\alpha + \beta + \gamma = 1$



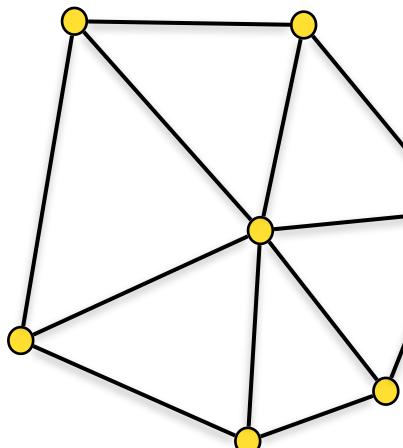
Polygonale Netze

- Tesselierung parametrischer Oberflächen
- Tesselierung von CAD-Modellen, z.B. Modellierungswerkzeuge wie AutoCAD, Maya, 3D Studio Max, Cinema 4D, Blender, ...
- Modellierung im „low-poly-look“
- Tesselierung impliziter Oberflächen
- Triangulierung von Punktwolken

Geschlossene Netze

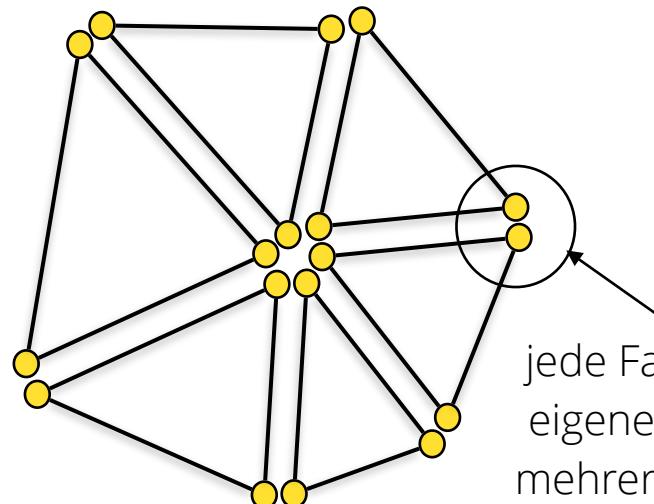
- Polygonale Netze können unterschiedliche Topologie haben

geschlossenes Dreiecksnetz



mehrere Facetten
teilen sich einen
Vertex

Dreieckssuppe

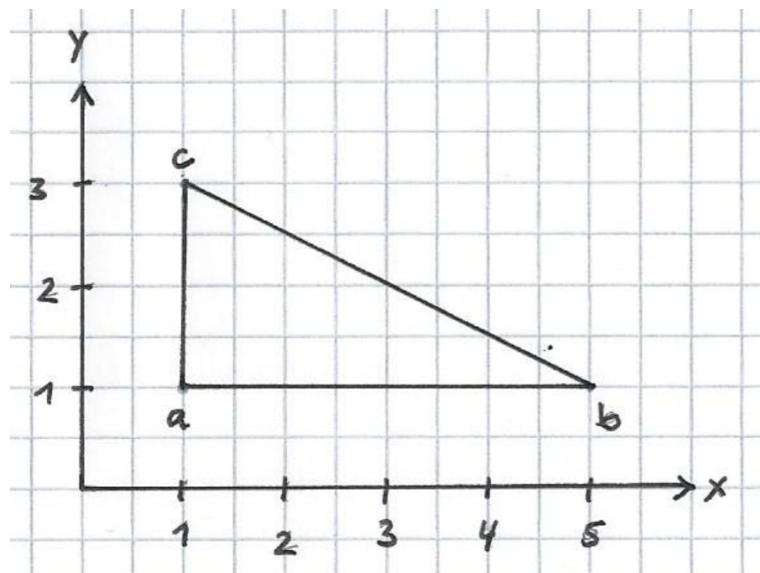


jede Facette hat
eigene Vertices,
mehrere Vertices
haben gleiche
Position

- Hinweis: gleiches visuelles Ergebnis nach dem Rendering

Übung: Fläche

- Berechnen Sie die Fläche des Dreiecks



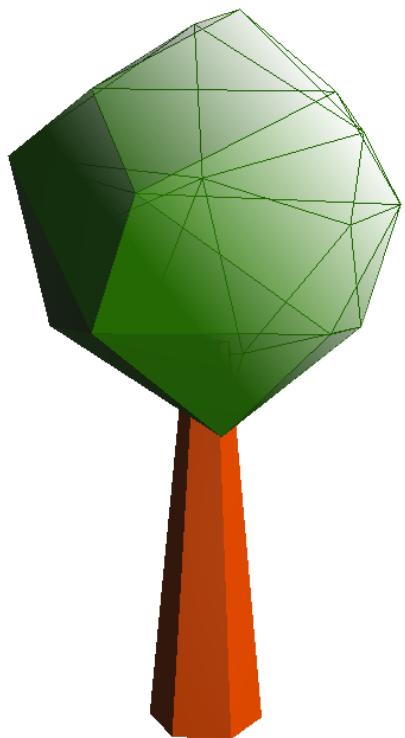
Erinnerung:
Kreuzprodukt

$$a = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$$

$$b = \begin{pmatrix} 5 \\ 1 \\ 0 \end{pmatrix}$$

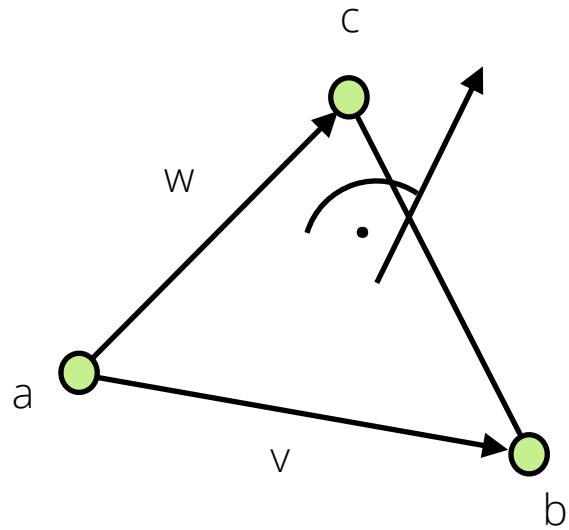
$$c = \begin{pmatrix} 1 \\ 3 \\ 0 \end{pmatrix}$$

$$\vec{v} \times \vec{w} = \begin{pmatrix} v_y w_z - v_z w_y \\ v_z w_x - v_x w_z \\ v_x w_y - v_y w_x \end{pmatrix}$$



Normalen

Dreieck



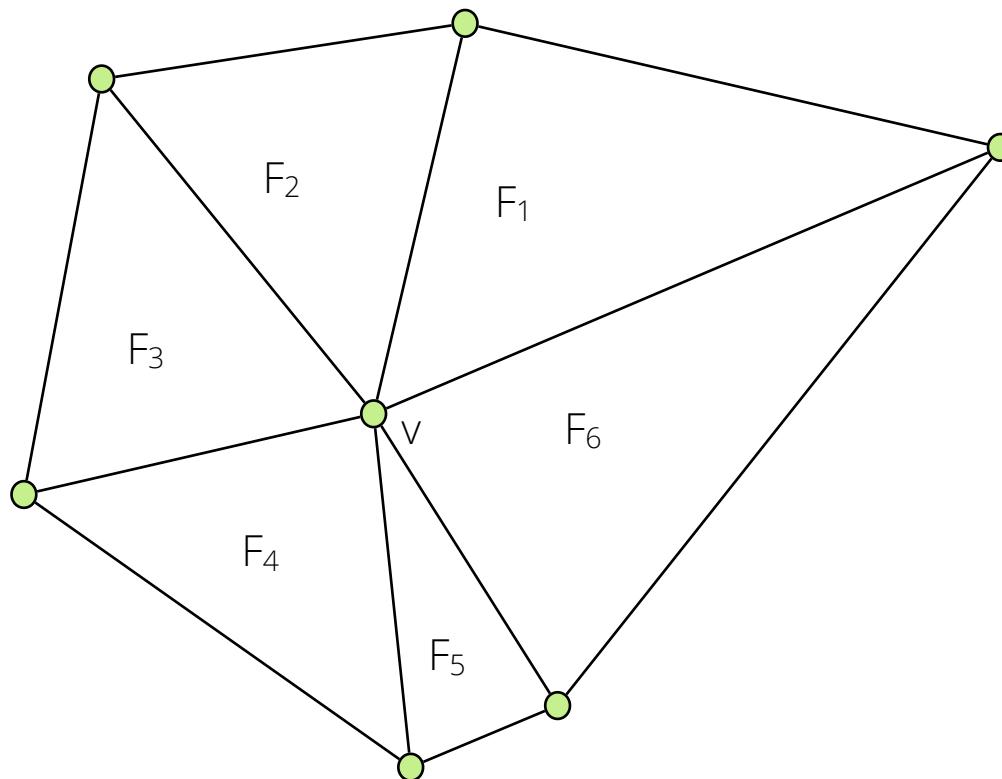
$$\text{Normale } n = \frac{v \times w}{\|v \times w\|} = \frac{(b - a) \times (c - a)}{\|(b - a) \times (c - a)\|}$$

Vertex-Normale

- gemittelt aus inzidenten Facetten
- verschiedene Möglichkeiten
 - keine Gewichtung
 - Gewichtung durch Winkel
 - Gewichtung durch Fläche

Keine Gewichtung

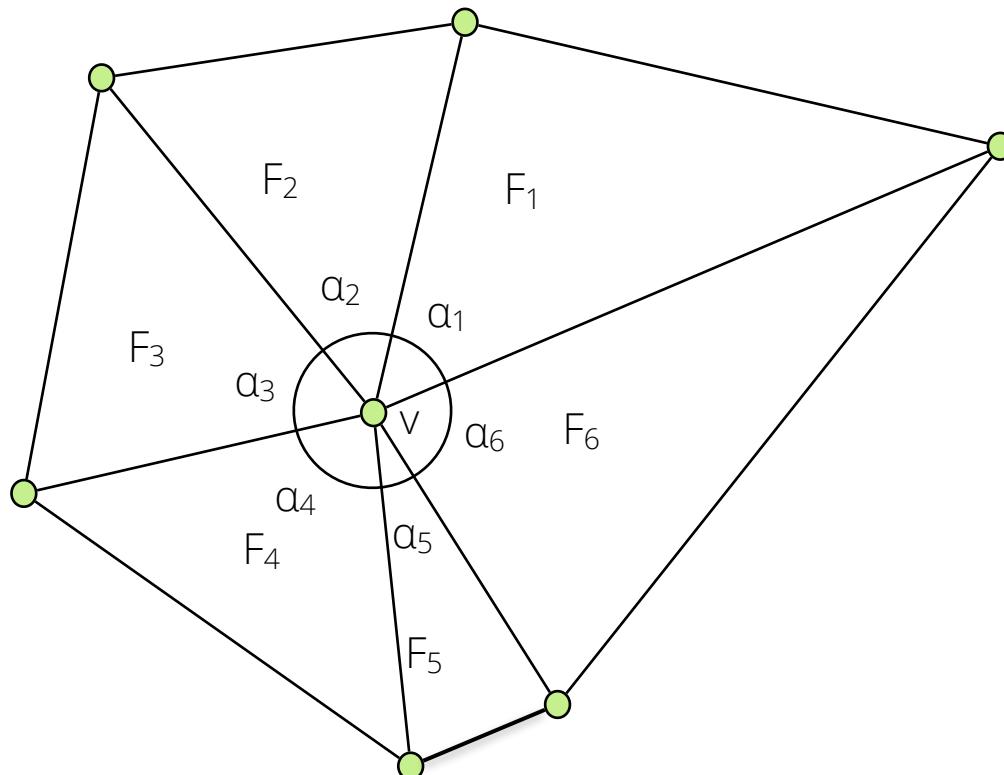
- Normalen der inzidenten Facette i: \vec{n}_i
- $\text{Grad}(v) = N$



$$\vec{n} \propto \sum_{i=1}^N \vec{n}_i$$

Gewichtung durch Winkel

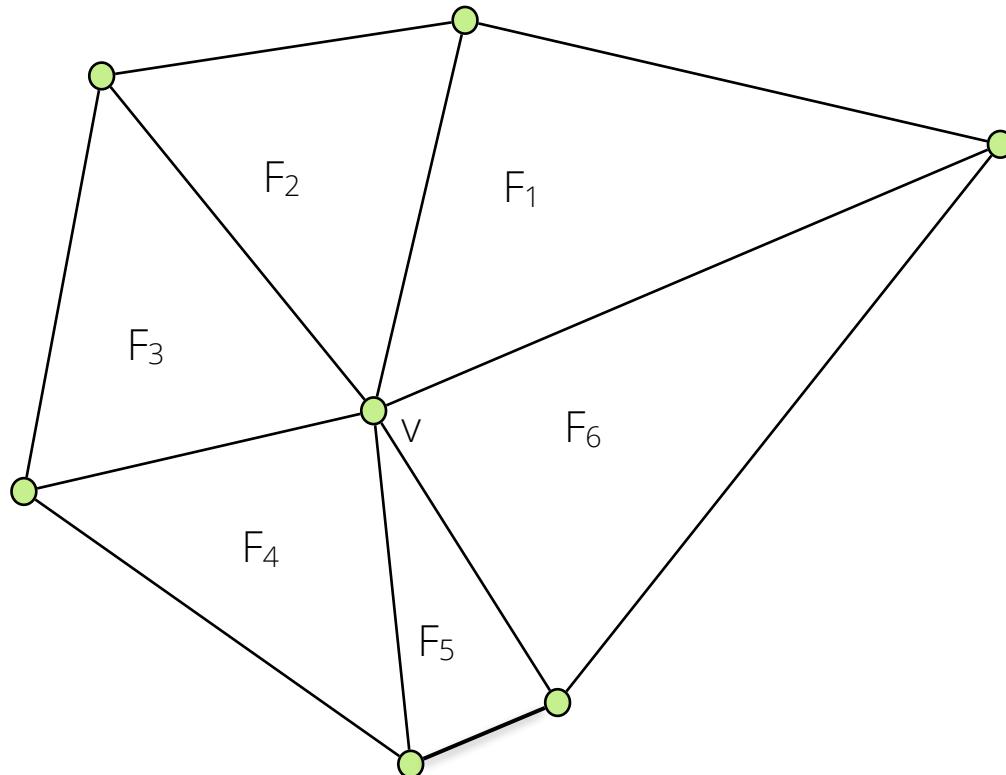
- Normalen der inzidenten Facette i: \vec{n}_i
- $\text{Grad}(v) = N$
- Winkel an der Facette: a_i



$$\vec{n} \propto \sum_{i=1}^N \alpha_i \vec{n}_i$$

Gewichtung durch Fläche

- Normalen der inzidenten Facette i: \vec{n}_i
- $\text{Grad}(v) = N :$
- Fläche der Facette i: A_i

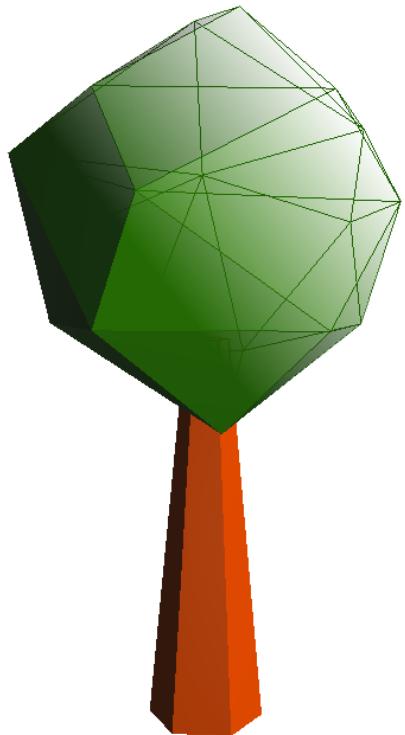


$$\vec{n} \propto \sum_{i=1}^N A_i \vec{n}_i$$

Übung: Vertex-Normale

- Berechnen Sie die Vertex-Normale n
- aus den Normalen der 5 inzidenten Facetten
- keine Gewichtung
- Normalen der inzidenten Facetten:

$$\begin{pmatrix} 0.5 \\ 1 \\ 2 \end{pmatrix}, \begin{pmatrix} 1.5 \\ -0.5 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0.5 \\ 2.5 \end{pmatrix}, \begin{pmatrix} 1 \\ -2.5 \\ 1.5 \end{pmatrix}, \begin{pmatrix} 2 \\ 1.5 \\ 1 \end{pmatrix}$$

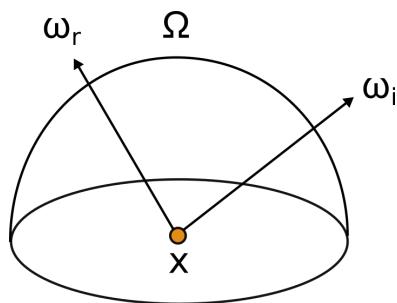


Lokale Beleuchtungsmodelle (Phong)

Einführung

- Lichttransport zwischen Oberflächen
- komplexes Verhalten: Scattering & Reflexion
- Beschreibung durch die Rendering-Gleichung

- siehe [5]



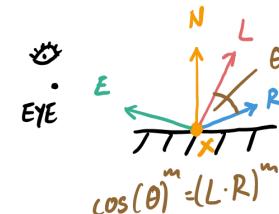
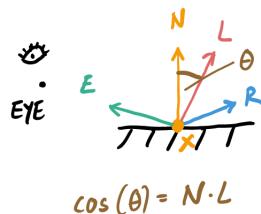
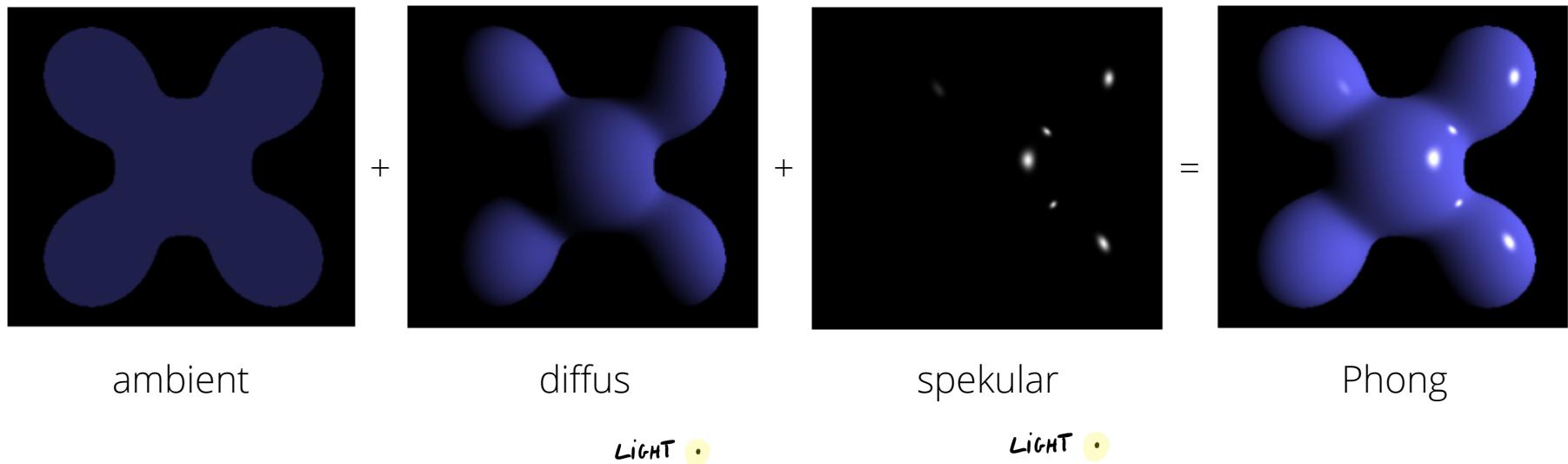
Subsurface Scattering
(Quelle: John Halpin, Flickr, CC-BY,
<https://flic.kr/p/AypFGZ>)

$$L_r(x, \omega_r) = L_e(x, \omega_r) + \int_{\Omega_i} \rho(x, \omega_r, \omega_i) L_i(x, \omega_i) \cos(\theta_i) d\omega_i$$

- Berechnung (noch) nicht für interaktive Anwendungen geeignet

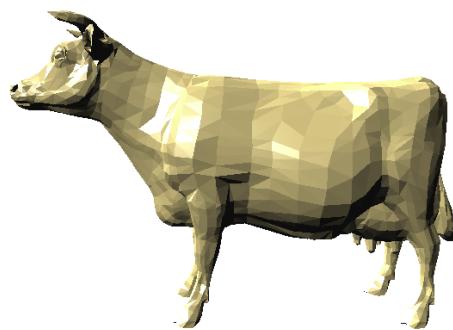
Phong-Beleuchtungsmodell

- für n Lichtquellen: $L_{Phong} = L_{amb} + \sum_{j=1}^n (L_{diff} + L_{spec})$
- siehe [6]



Quelle: Brad Smith - Eigenes Werk, CC BY-SA 3.0, <https://commons.wikimedia.org/w/index.php?curid=1030364>

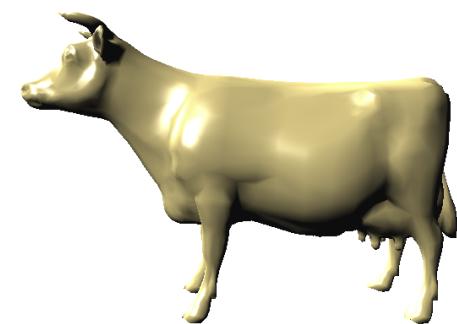
Shading



Flat Shading



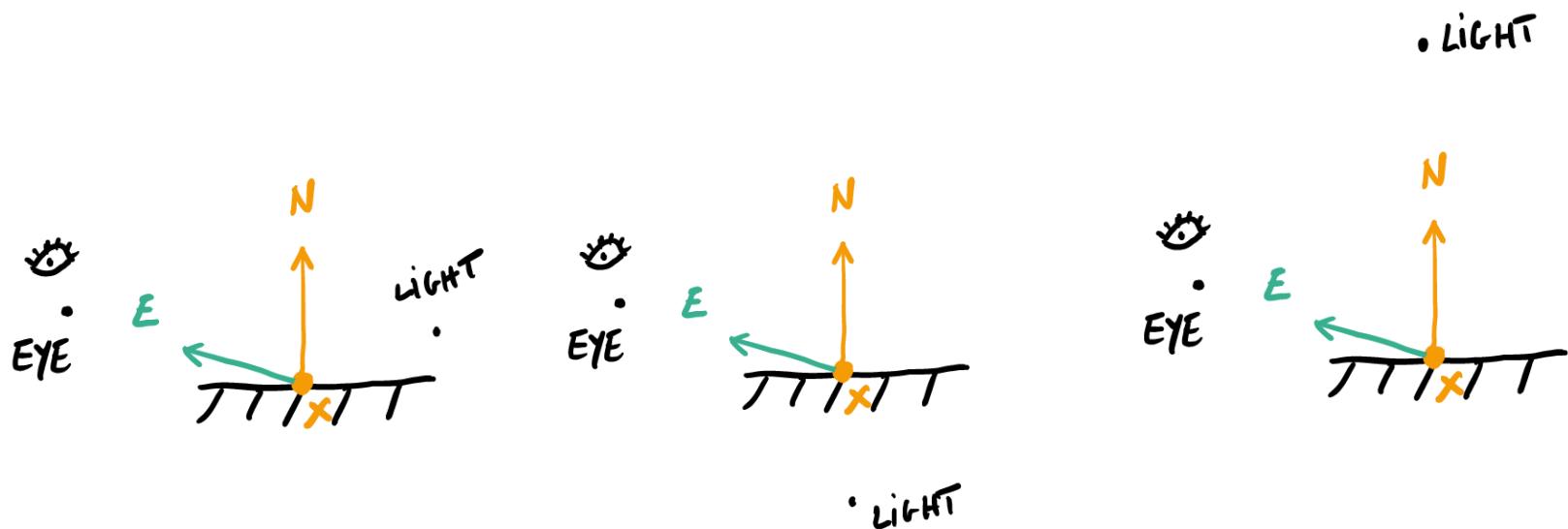
Gouraud shading

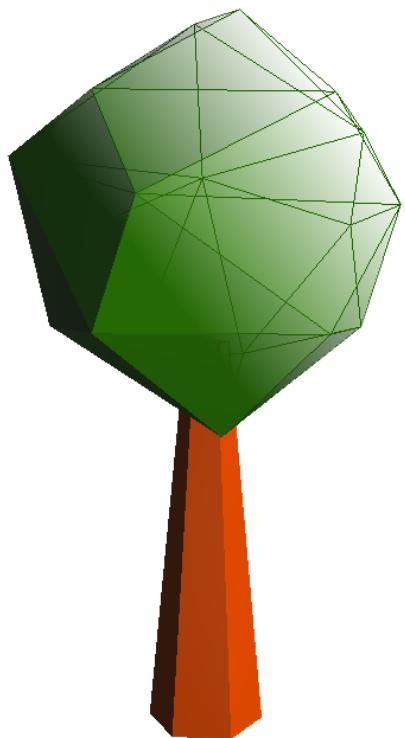


Phong shading

Übung: Beleuchtung

- Welche der Komponenten im Phong Beleuchtungsmodell hat jeweils den größten Einfluss bei den drei Beispielen?

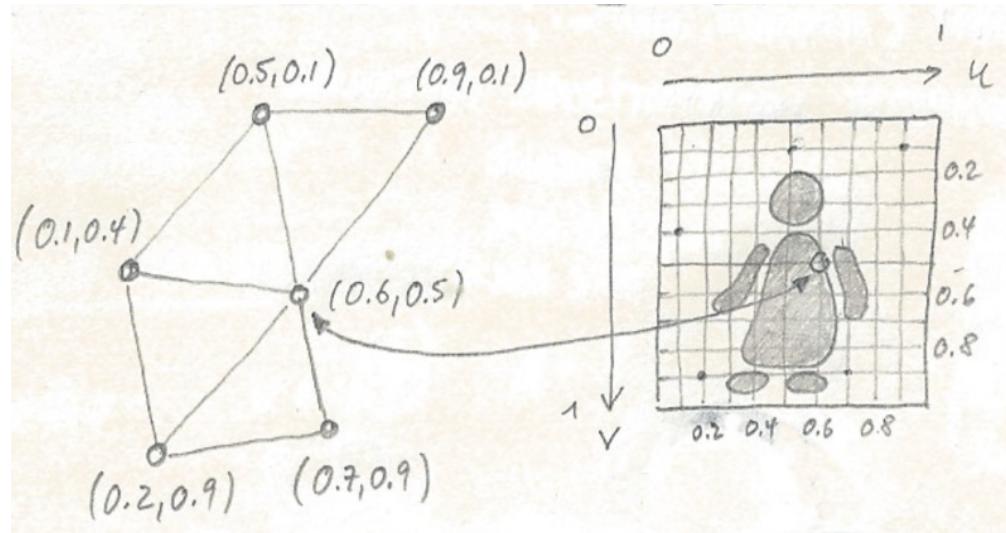




Texturen

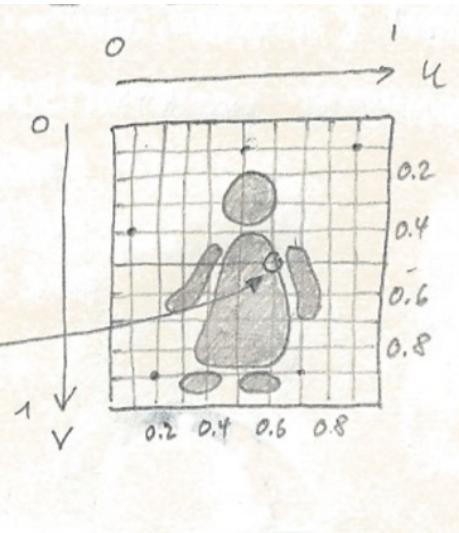
Objektraum vs. Texturraum

Objektraum

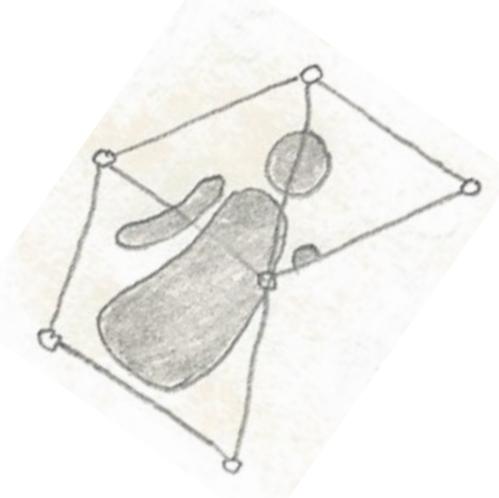


Dreiecksnetz mit
Texturkoordinaten (u, v) für
jeden Vertex

Texturraum



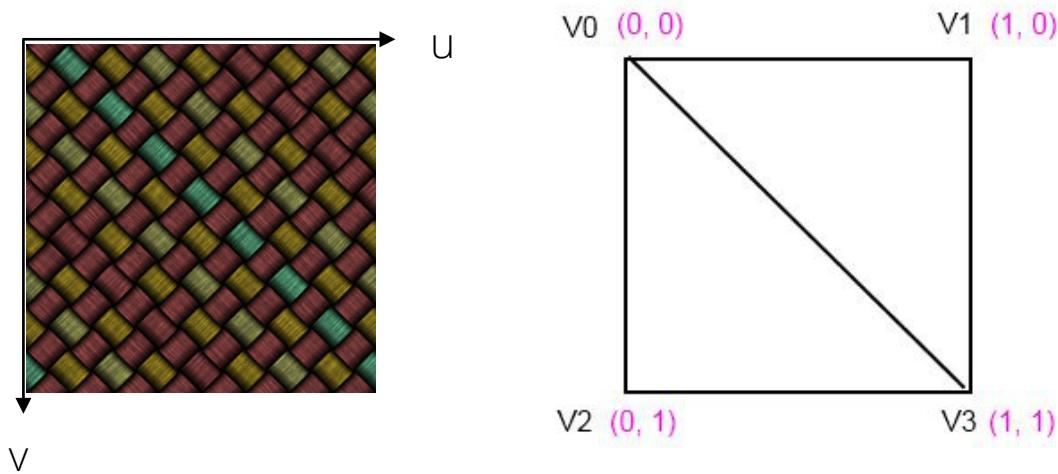
Textur auf Oberfläche
gemapped



Hinweis: Texturkoordinaten werden an die Ecken der
Polygone gebunden (nicht an die Vertices)

Texturkoordinaten

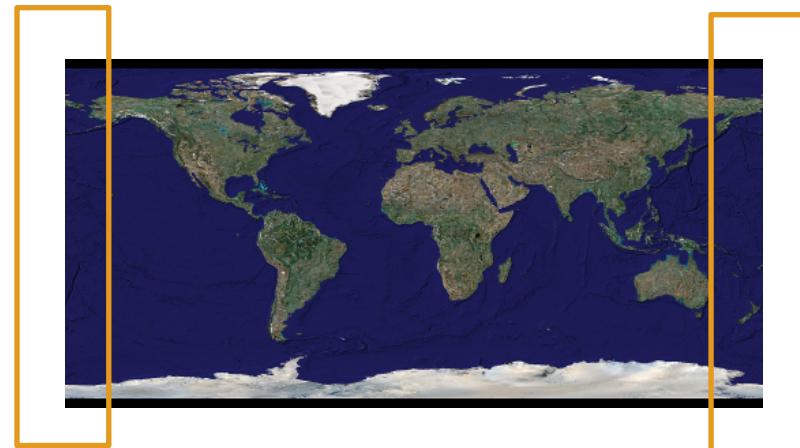
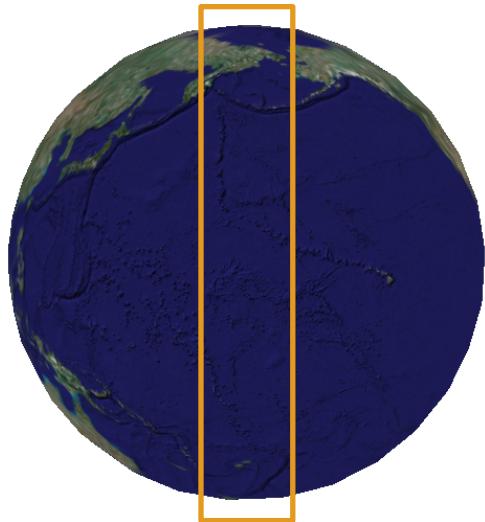
- in OpenGL ES:



Quelle: Patrick Hoesly, Flickr, CC-BY, <https://flic.kr/p/7X6nmx>

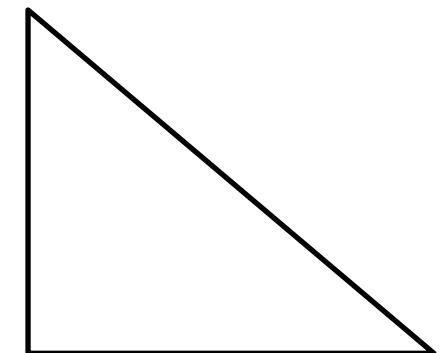
Texturkoordinaten

- eine Texturkoordinate für jeden Vertex?
 - Was ist mit Nähten?



(t_i^x, t_i^y)

- Texturkoordinaten werden mit Facetten assoziiert
 - gleiche Representation wie Vertices
 - Liste der Texturkoordinaten
 - Texturkoordinaten-Indizes in den Facettenecken
 - z.B. 3 für jedes Dreieck



(t_i^x, t_i^y)

(t_k^x, t_k^y)

Texturatlas

- Zusammenfügung mehrerer Texturen in einem Bild
- Teilbilder/-texturen werden durch Texturkoordinaten angesprochen



Texturatlas

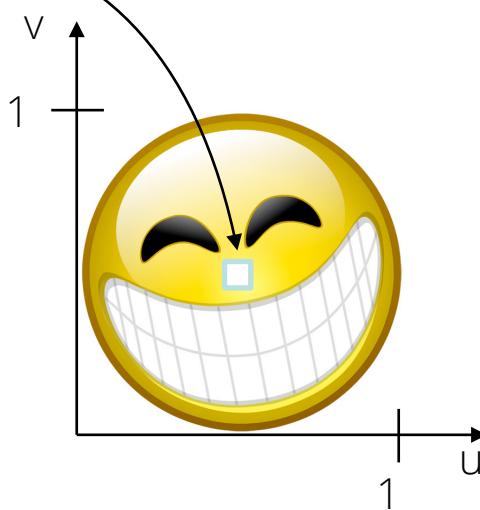
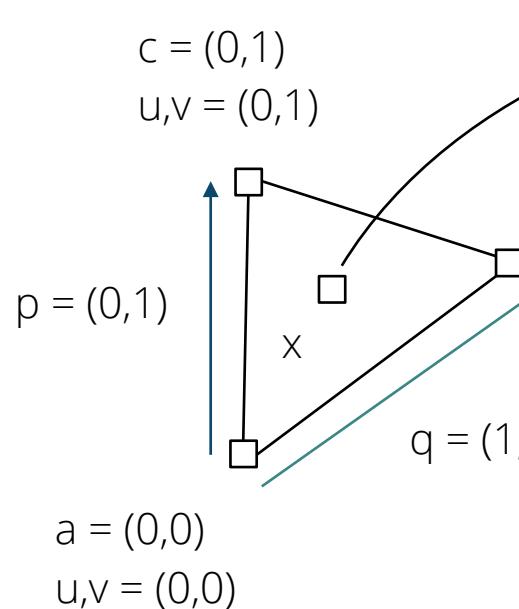
Quelle: http://wiki.simigon.com/wiki/index.php?title=Terrain_texture_atlas,
abgerufen am 8.11.13

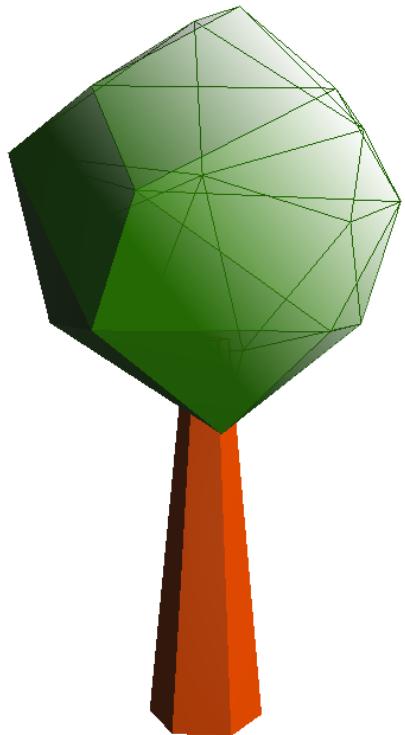
Übung: Texturkoordinaten

- Gegeben: Dreieck a, b, c mit Texturkoordinaten
- Aufgabe: Finde Texturkoordinaten an Position x

$$x = (0.25, 0.5)$$

$$u, v = ?$$

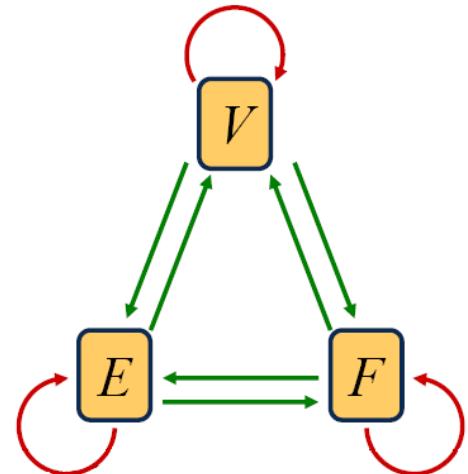




Nachbarschaftsdatenstrukturen

Nachbarschaftsdatenstrukturen

- Idee: günstige Operationen ($O(1)$) für den Zugriff auf benachbarte Entitäten
- Relationen:
 - inzident:
$$\begin{array}{ll} v_i \rightarrow \{f_{i,j}\}_j & v_i \rightarrow \{e_{i,j}\}_j \\ e_i \rightarrow (v_{i,1}, v_{i,2}) & e_i \rightarrow (f_{i,1}, f_{i,2}) \\ f_i \rightarrow \{e_{i,j}\}_j & f_i \rightarrow \{v_{i,j}\}_j \end{array}$$
 - adjacent:
$$\begin{array}{ll} v_i \rightarrow \{v_{i,j}\}_j & f_i \rightarrow \{f_{i,j}\}_j \\ e_i \xrightarrow{v} \{e_{i,j}\}_j & e_i \xrightarrow{f} \{e_{i,j}\}_j \end{array}$$
- nur für wohlgeformte Dreiecksnetze definiert
- alle Relationen explizit repräsentieren:
 - specheraufwändig(er)
 - aufwändig(er) bei Aktualisierung

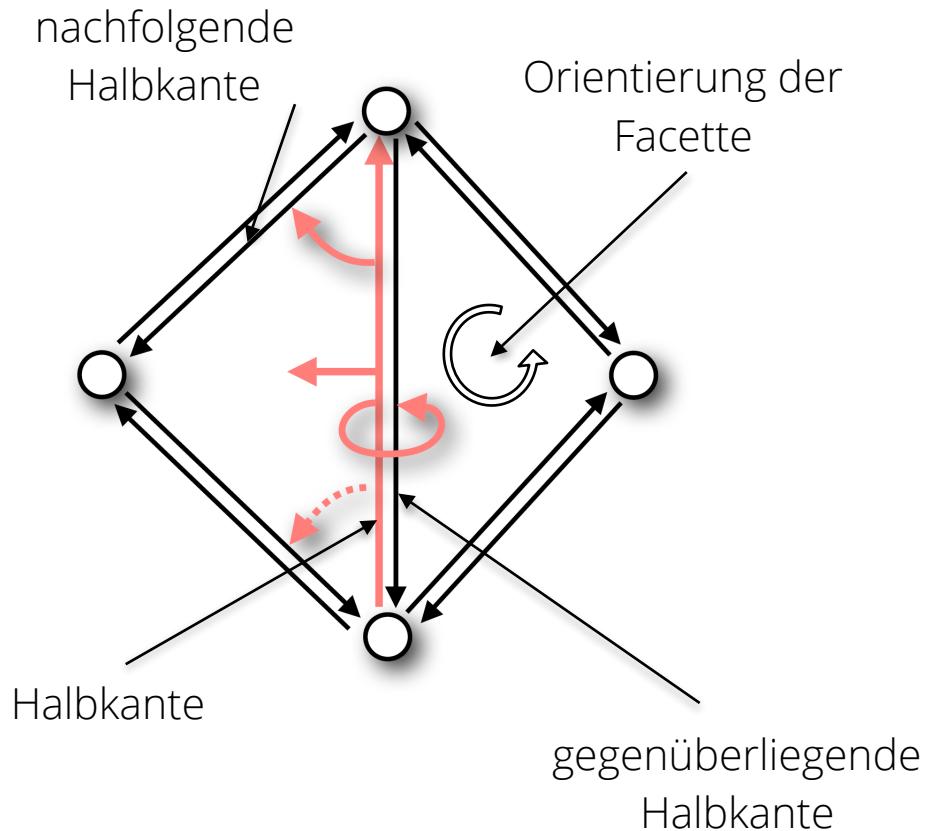


Speicher vs. Performance

- so wenige Relationen wie möglich speichern
- günstige Berechnung der übrigen Relationen

Halbkantendatenstruktur

- Vertex
 - Position
 - ausgehende Halbkante (eine beliebige)
- Halbkante
 - Start-Vertex
 - gegenüberliegende Halbkante
 - nachfolgende Halbkante
 - Facette
- Facette
 - Halbkante (eine beliebige)
- siehe: [3]

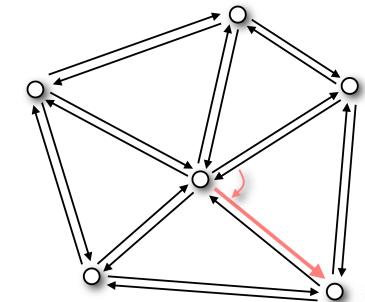
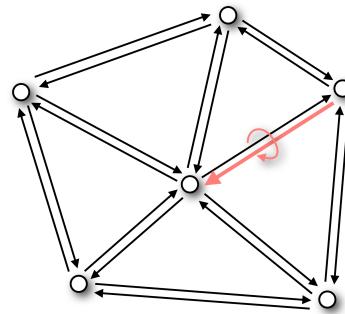
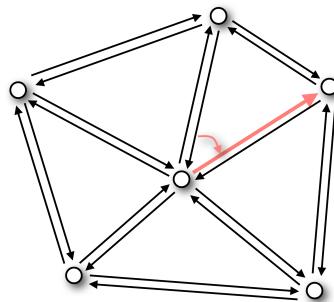
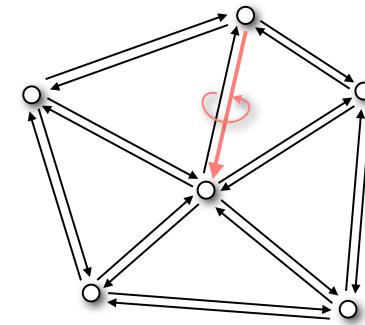
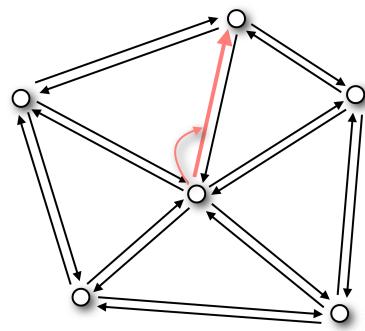
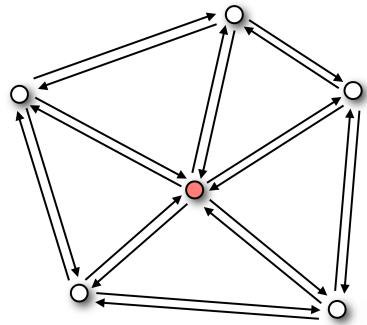


Hinweise

- Datenstruktur verwendet Referenzen, nicht Indizes
- Listen für Vertices, Halbkanten, Facetten
- nur eine Halbkante pro Vertex (zufällig ausgewählt)
- nur eine Halbkante pro Facette (zufällig ausgewählt)

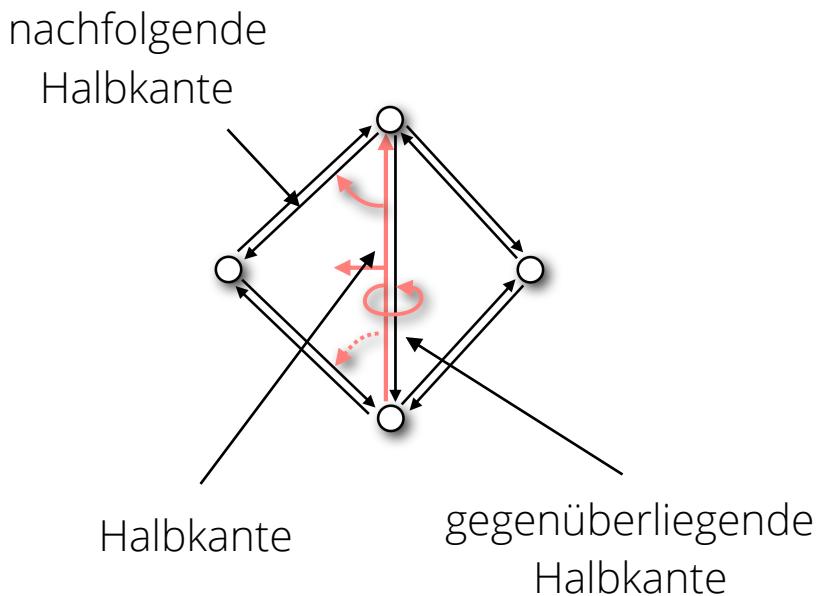
Beispiel: inzidente Halbkanten

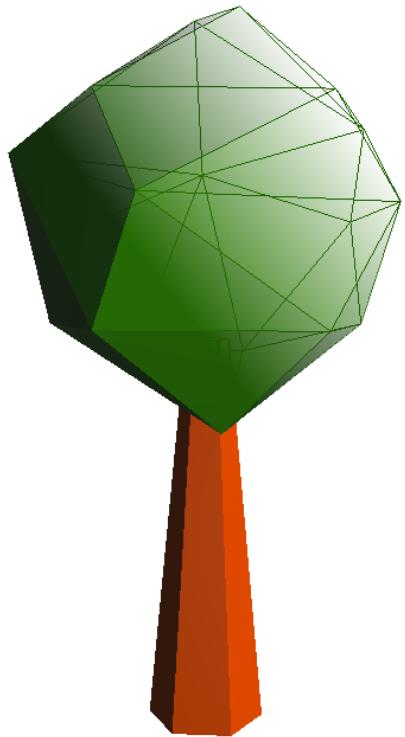
- Finde alle inzidenten Halbkanten



Übung: adjacente Vertices

- Schreiben Sie einen Algorithmus in Pseudocode, der in einer Halbkanten-Datenstruktur alle adjazenten Vertices zu einem Vertex v findet.
- Vertex
 - Position
 - ausgehende Halbkante (eine beliebige)
- Halbkante
 - Start-Vertex
 - gegenüberliegende Halbkante
 - nachfolgende Halbkante
 - Facette
- Facette
 - Halbkante (eine beliebige)

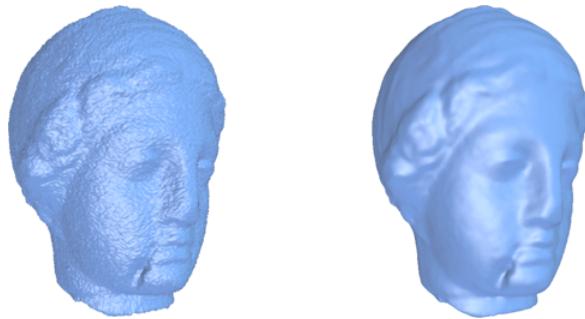




Glättung

Glättung

- viele Datensätze sind verrauscht
 - z.B. wegen Sensorrauschen bei der Aufnahme
- Entrauschen/Glättung notwendig
- Idee
 - Projektion auf glatten Oberflächenausschnitt
 - z.B. Ebene oder Kugel



verrauschter Datensatz (links),
geglätteter Datensatz (rechts)

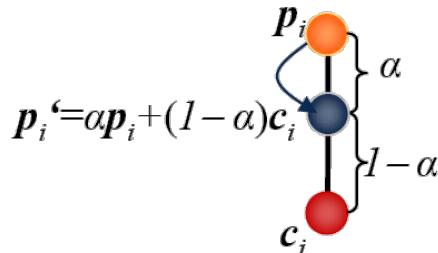
Quelle: B. Mederos, L. Velho, L. H. de Figueiredo:
Smooth surface reconstruction from noisy clouds, in:
J. Braz. Comp. Soc. vol.9 no.3 Campinas Apr. 2004

Laplace (auch Gaussian) Glättung

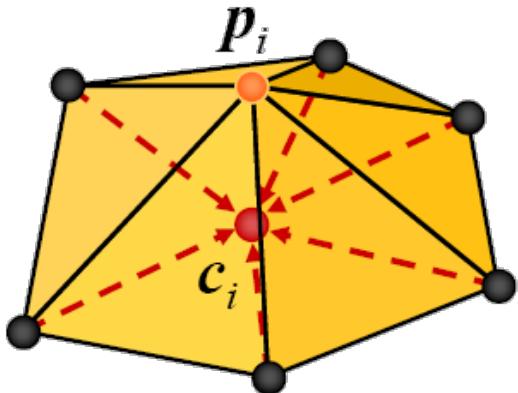
- Idee: bewege alle Vertices zum Schwerpunkt der Nachbarn
- Algorithmus
 - für jeden Vertex p_i
 - berechne Schwerpunkt der c_i der Nachbarn

$$c_i = \frac{1}{|N(p_i)|} \sum_{p_j \in N(p_i)} p_j$$

- bewege p_i in Richtung von c_i (e.g. 50%: $\alpha = 0.5$)



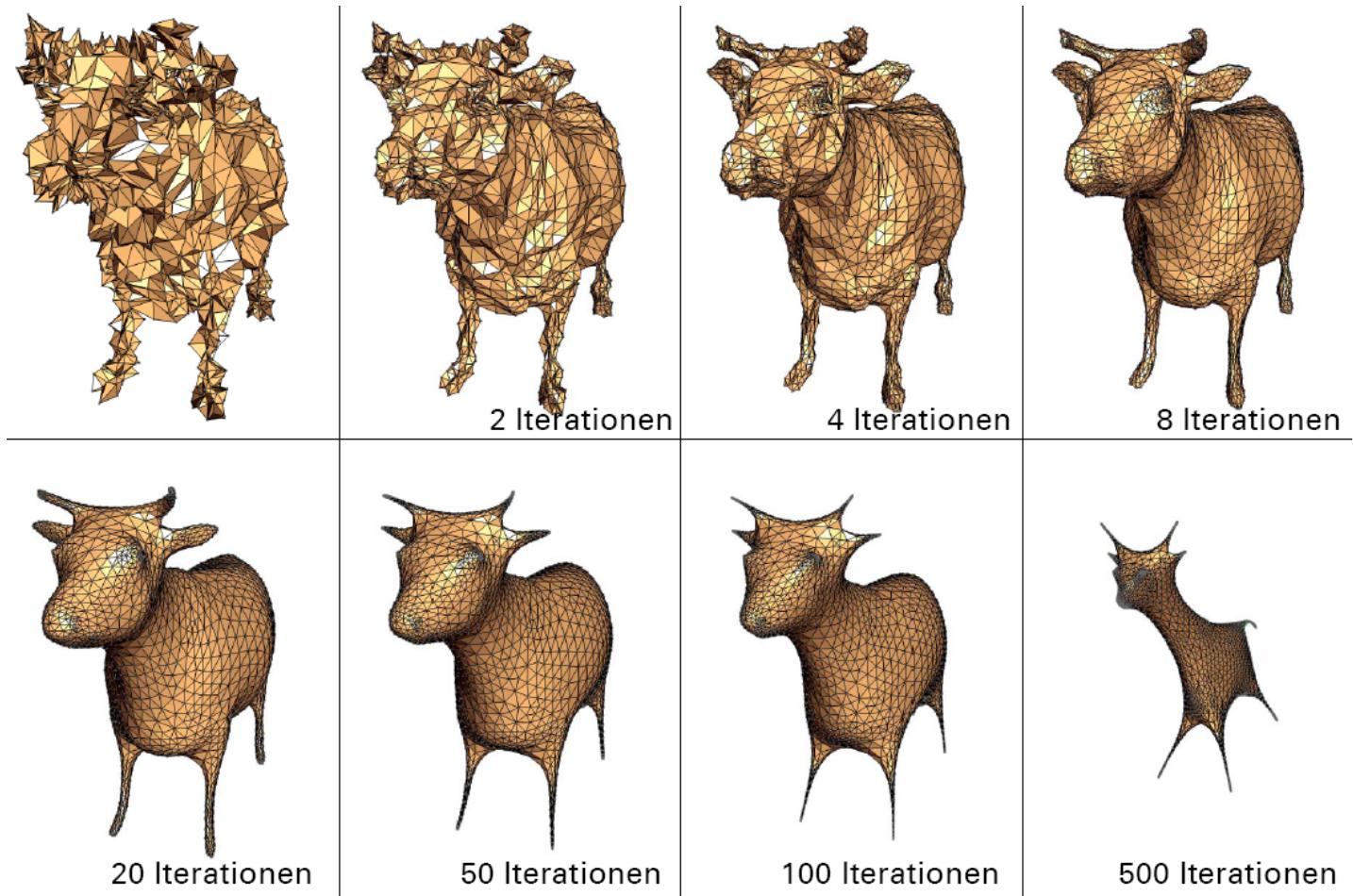
- [Hinweis: bestimme zunächst c_i (Caching), bewege dann p_i .]
- siehe: [4]



Bildquelle: Prof. Dr. Stefan
Gumhold (Technical University
Dresden), lecture slides

Laplace Glättung

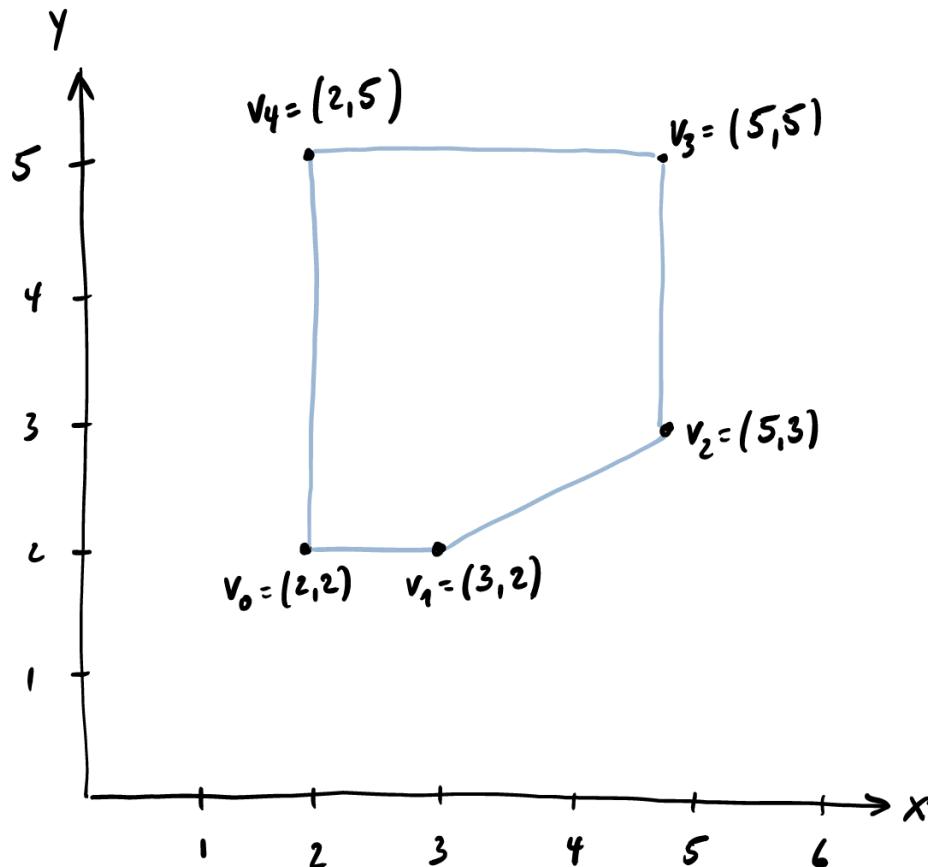
- $\alpha = 0.3$

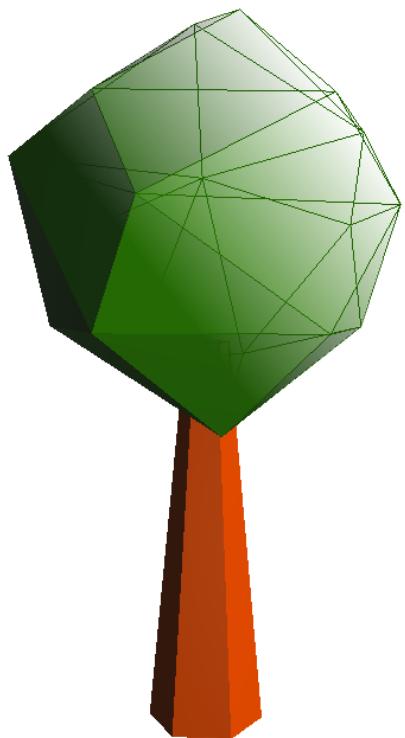


Bildquelle: Prof. Dr. Stefan Gumhold (Technical University Dresden)

Übung: Glättung

- Führen Sie einen Glättungsschritt mit $\alpha = 0.5$ durch





Grundlagen: Matrizen

Matrizen

- Matrix
- Transponierte
- Multiplikation
- Inverse

Zusammenfassung

- Dreiecksnetze
- Normalen
- Lokale Beleuchtungsmodelle
- Texturen
- Nachbarschaftsdatenstrukturen
- Glättung
- Grundlagen: Matrizen

Quellen

- [1] Hossam ElGindy, Hazel Everett, Godfried Toussaint: Slicing an ear using prune-and-search, in Pattern Recognition Letters, Volume 14, Issue 9, 1993
- [2] Möbius, 1827 (in Coxeter 1969, p. 217; Fauvel *et al.* 1993).
- [3] S. Campagna, L. Kobbelt, H.-P. Seidel: Directed Edges - A Scalable Representation For Triangle Meshes, in ACM Journal of Graphics Tools 3 (4), 1998.
- [4] G. Taubin: A Signal Processing Approach to Fair Surface Design, in Proceedings of SIGGRAPH 95, 351–358, 1995
- [5] James T. Kajiya: The rendering equation, in Proceedings of the 13th annual conference on Computer graphics and interactive techniques (SIGGRAPH '86), 1986
- [6] Bui Tuong Phong: Illumination for computer generated pictures, in Communications of the ACM 18, 6 (June 1975), 1975