

Teil 1: Modellierung

Objekte und ihre Beschreibung

Einleitung

Computergraphik: 3D sehr wichtig

- ◆ photo-realistic rendering
- ◆ Computer-Animation, Modellierung
- ◆ Visualisierung, Virtual Reality

Ansatz:

- ◆ per **rendering** wird eine **3D-Szene** dargestellt

Frage: **Beschreibung** der Szene?

3D Szene – Inhalt

Virtuelle Welt:

- ◆ Objekte als Abbild wirklicher Objekte: Architektur, Objekte des tägl. Lebens, ...
- ◆ Oberflächen: eben, uneben, fraktal
- ◆ Volumensobjekte: innere Struktur
- ◆ real-3D objects: Fraktale wie Wolken, ...

Aber wie beschreiben?

3D Szene – Beschreibung

Modellierung:

- ◆ Virtuelle 3D Welt – soll dargestellt werden
- ◆ Szene-Beschreibung (Graph-Form)
- ◆ Teile:
 - ◆ Objekte – 0D, 1D, 2D, 3D
 - ◆ Objekt-Attribute
 - ◆ Transformationen
 - ◆ Beleuchtung
 - ◆ Kamera

In der Szene: Objekte

Einfache Objekte

- ◆ **Primitiva:** Punkt (0D), Liniensegment (1D), Bézier-Kurve (1D), Dreieck (2D), Patch (2D), Voxel (3D), ...

Komplexe Objekte: z.B.:

- ◆ **Sammel-Objekte:** Poly-Linie (1D), Mesh (2D), Volumensdatensatz (3D), ...
- ◆ **Transformierte Objekte:** sweeps, ...
- ◆ **Kombinierte Objekte:** CSG, ...

Objekte – Überblick

Basis: Punkte, Liniensegmente, etc.

Diskrete Approximation: meshes

Erweiterung: terrains, fraktale Gebirge

Modellierung durch sweeps

Modellierung durch soft objects

Modellierung: Partikelsysteme

Basis: Punkte im 3D

Start im 3D: Punkt:

- ◆ Darstellung: vektoriell $\rightarrow \mathbf{x} = (x_1, x_2, x_3)^T$

Unterschied Punkt \Leftrightarrow Vektor

- ◆ Vektor = translationsinvariant
- ◆ Vektor = (End-)Punkt \mathbf{q} – (Start-)Punkt \mathbf{p}
 $= \mathbf{q} - \mathbf{p} = (q_1 - p_1, q_2 - p_2, q_3 - p_3)^T$
 $= \mathbf{v} = (v_1, v_2, v_3)^T$
- ◆ Liniensegment: durch \mathbf{p} und \mathbf{q}
oder \mathbf{p} und \mathbf{v} gegeben

WH (CGR3): Meshes

Basis:

- ◆ Dreieck, Viereck
- ◆ Infos: Eckpunkte, Normale(n), Nachbarn

Sehr oft in Verwendung: Meshes:

- ◆ Liste von Dreiecken bzw. Δ -strips
- ◆ BRep (WH): Punkte \rightarrow Kanten \rightarrow Flächen
- ◆ Winged-Edge DS (WH): Kante++

Aspekte: Speicherplatz, Berechnungen

Datenstruktur: Δ -strip

13 Punkte:
11 Dreiecke

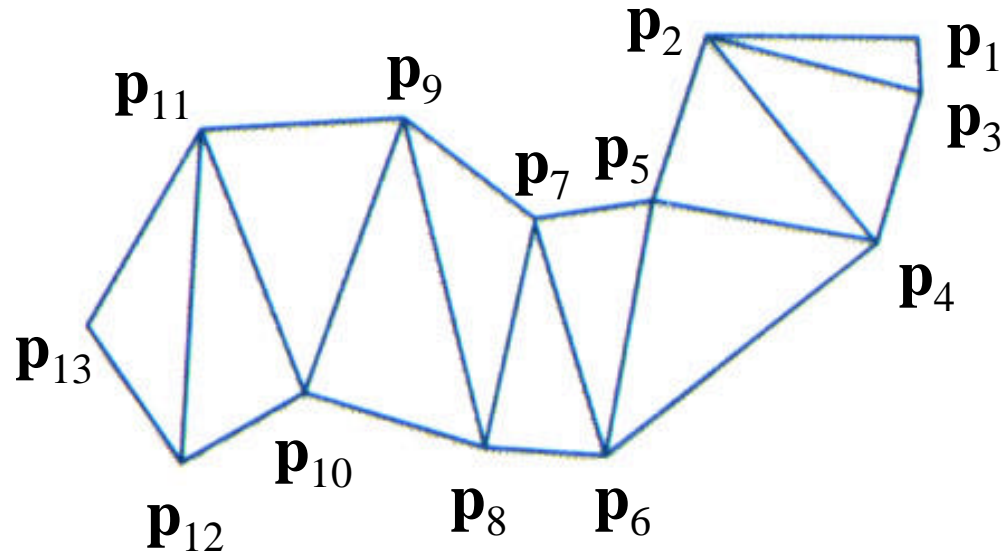


Figure 10-6

A triangle strip formed with 11 triangles connecting 13 vertices.

Quadrilateral Mesh

5*4 Punkte:
12 Vierecke!

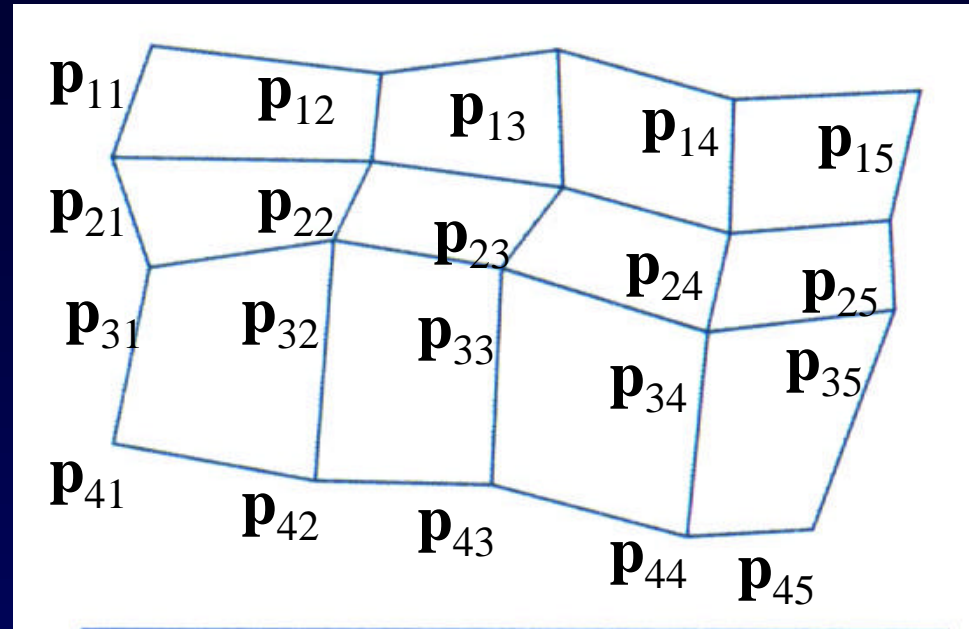


Figure 10-7

A quadrilateral mesh containing 12 quadrilaterals constructed from a 5 by 4 input vertex array.

BRep: Beispiel

Mesh:

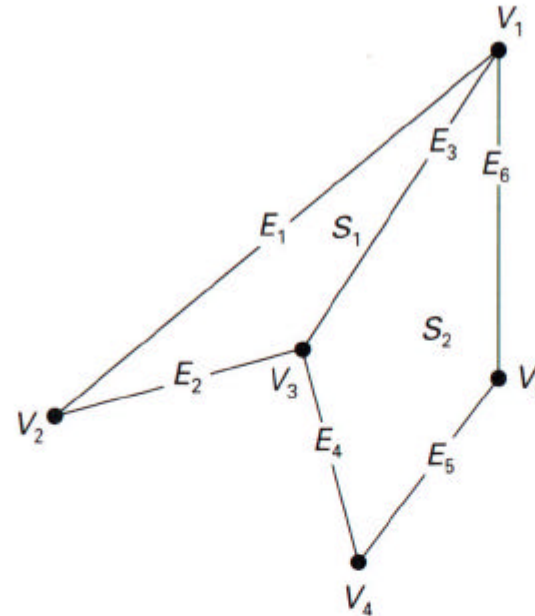
- ◆ Dreieck S_1
- ◆ Viereck S_2

6 Kanten:

- ◆ 1 gemeinsam

5 Eckpunkte

- ◆ 2 gemeinsam



VERTEX TABLE

V_1 :	x_1, y_1, z_1
V_2 :	x_2, y_2, z_2
V_3 :	x_3, y_3, z_3
V_4 :	x_4, y_4, z_4
V_5 :	x_5, y_5, z_5

EDGE TABLE

E_1 :	V_1, V_2
E_2 :	V_2, V_3
E_3 :	V_3, V_1
E_4 :	V_3, V_4
E_5 :	V_4, V_5
E_6 :	V_5, V_1

POLYGON-SURFACE
TABLE

S_1 :	E_1, E_2, E_3
S_2 :	E_3, E_4, E_5, E_6

Objekte – Überblick

Basis: Punkte, Liniensegmente, etc.

Diskrete Approximation: meshes

Erweiterung: terrains, fraktale Gebirge

Modellierung durch sweeps

Modellierung durch soft objects

Modellierung: Partikelsysteme

Terrains

Definition:

- ◆ Basis-Gitter (2D),
pro Gitter-Punkt: 1 Höhenwert

Mögliche Erweiterung:

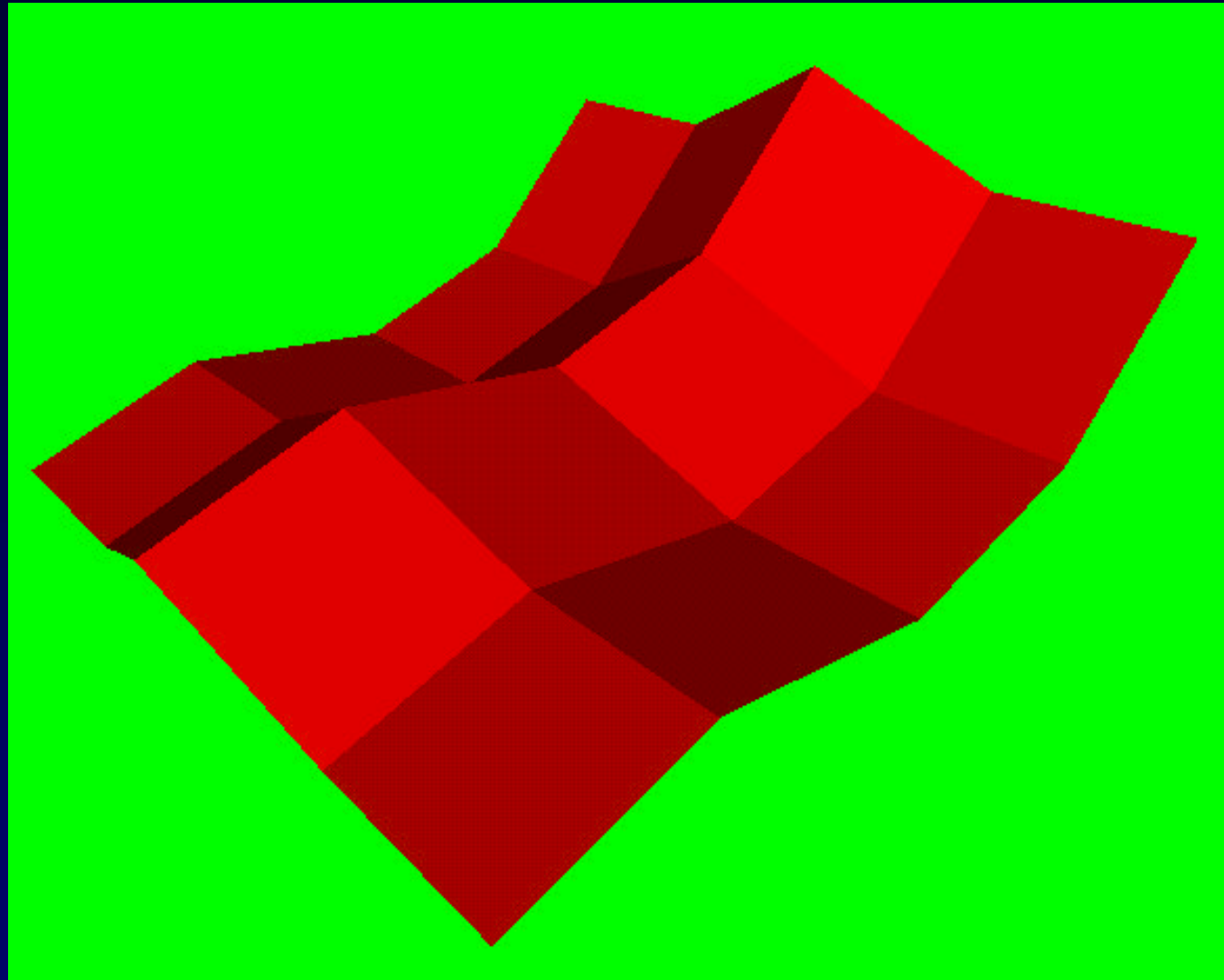
- ◆ Farbinformationen
- ◆ Texturen

VRML-Beispiel: terrain (1)

```
geometry ElevationGrid {  
    xDimension 5  
    zDimension 5  
    xSpacing 2  
    zSpacing 2  
    height [ 2, 2, 3, 2, 2,  
            1, 1, 2, 1, 1,  
            1, 1, 2, 1, 1,  
            2, 2, 3, 2, 2,  
            2, 2, 3, 2, 2 ]  
}
```

VRML-Beispiel: terrain (2)

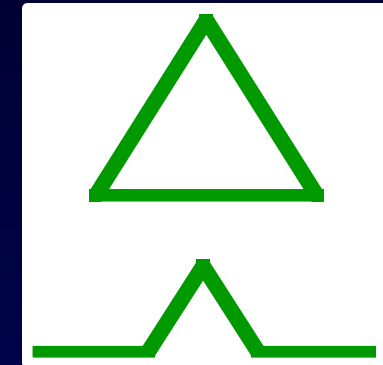
5*5
Terrain



Fraktale Gebirge (1)

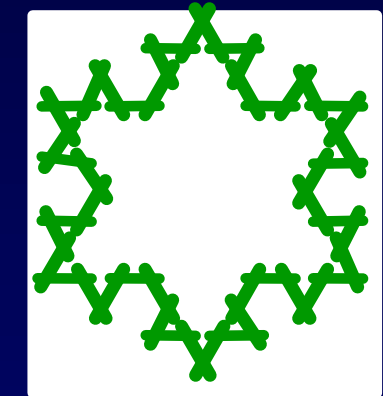
1D-Beispiel: Koch-Kurve

- ◆ Initiator: Startpolygon
- ◆ Generator: Ersetzungsregel



Fraktales Gebirge:

- ◆ Generator + Zufallszahlen
- ◆ Initiator: 1 Dreieck oder 2
- ◆ Generator:
jede Kante i. d. Mitte teilen,
Mittelpunkt per Zufallszahl verschieben



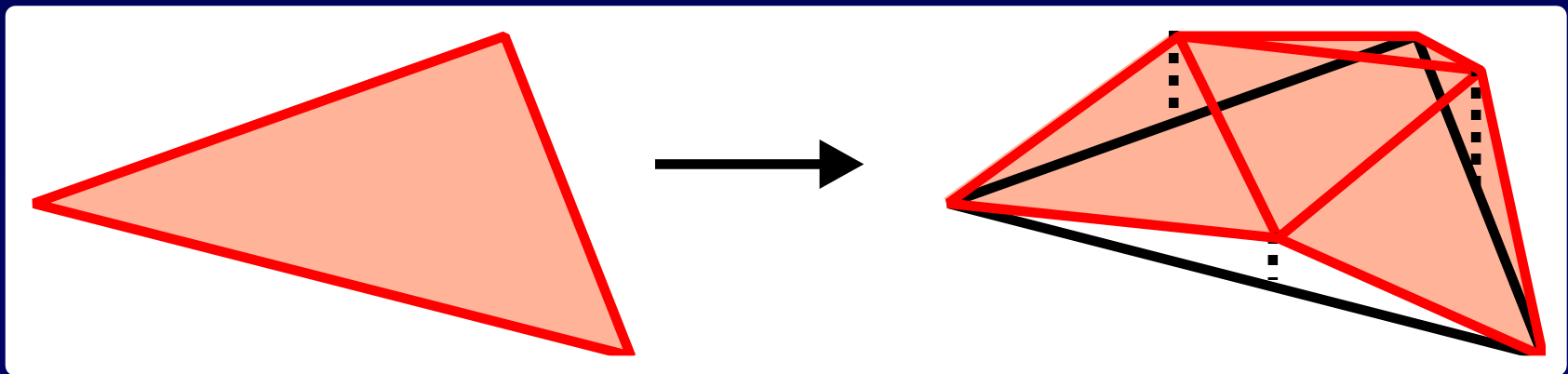
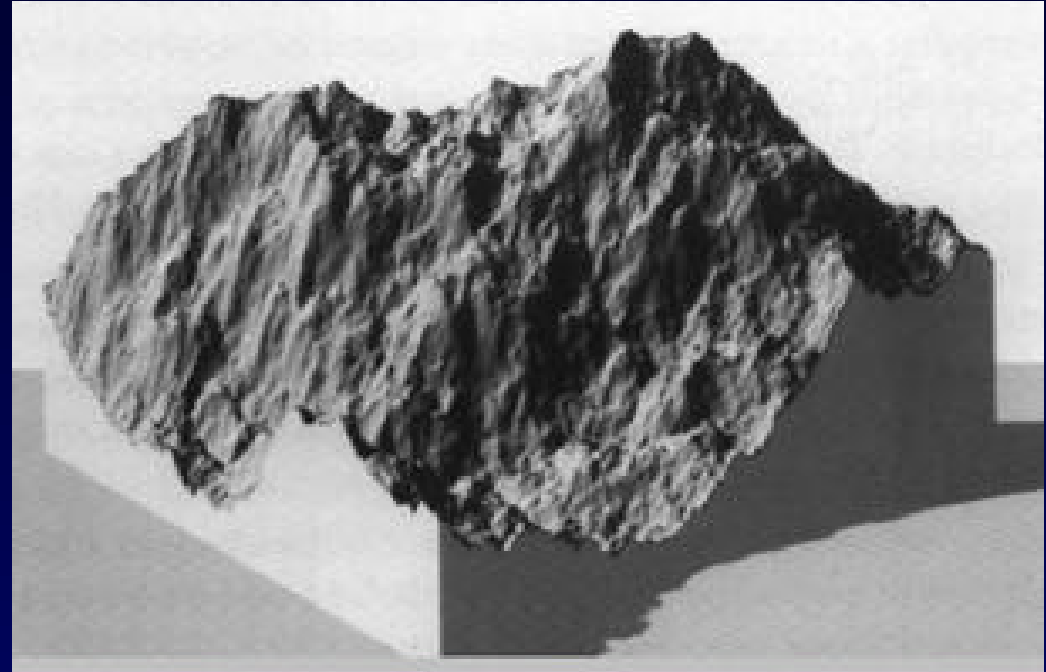
Fraktales Gebirge (2)

Pro Schritt:

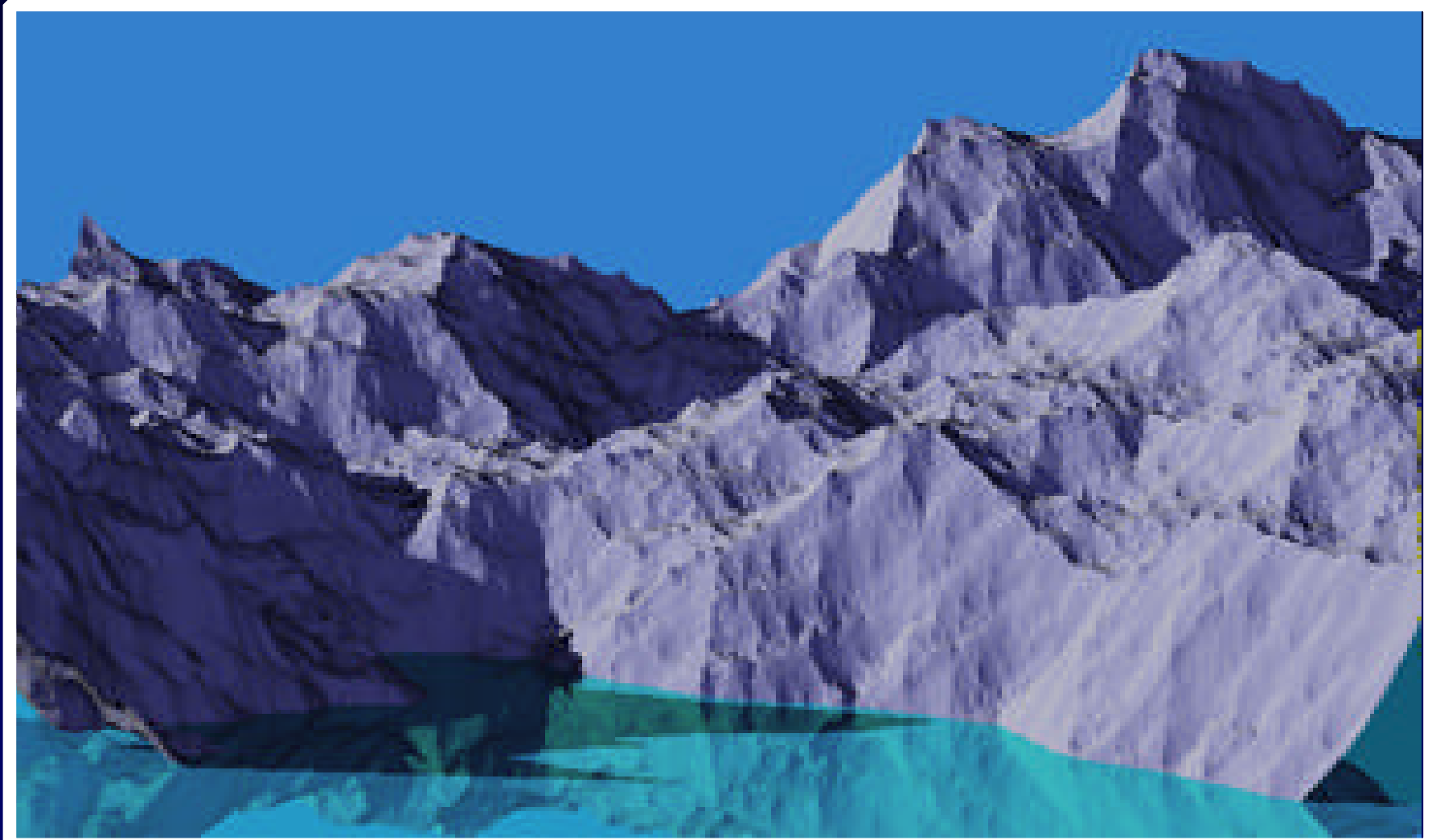
- ◆ 1 Dreieck →
4 Dreiecke

Stopp wenn:

- ◆ Unterteilung fein
genug



Beispiel: Fraktales Gebirge



Objekte – Überblick

Basis: Punkte, Liniensegmente, etc.

Diskrete Approximation: meshes

Erweiterung: terrains, fraktale Gebirge

Modellierung durch sweeps

Modellierung durch soft objects

Modellierung: Partikelsysteme

Sweeps

Idee:

- ◆ 2D Kontur + Transformation

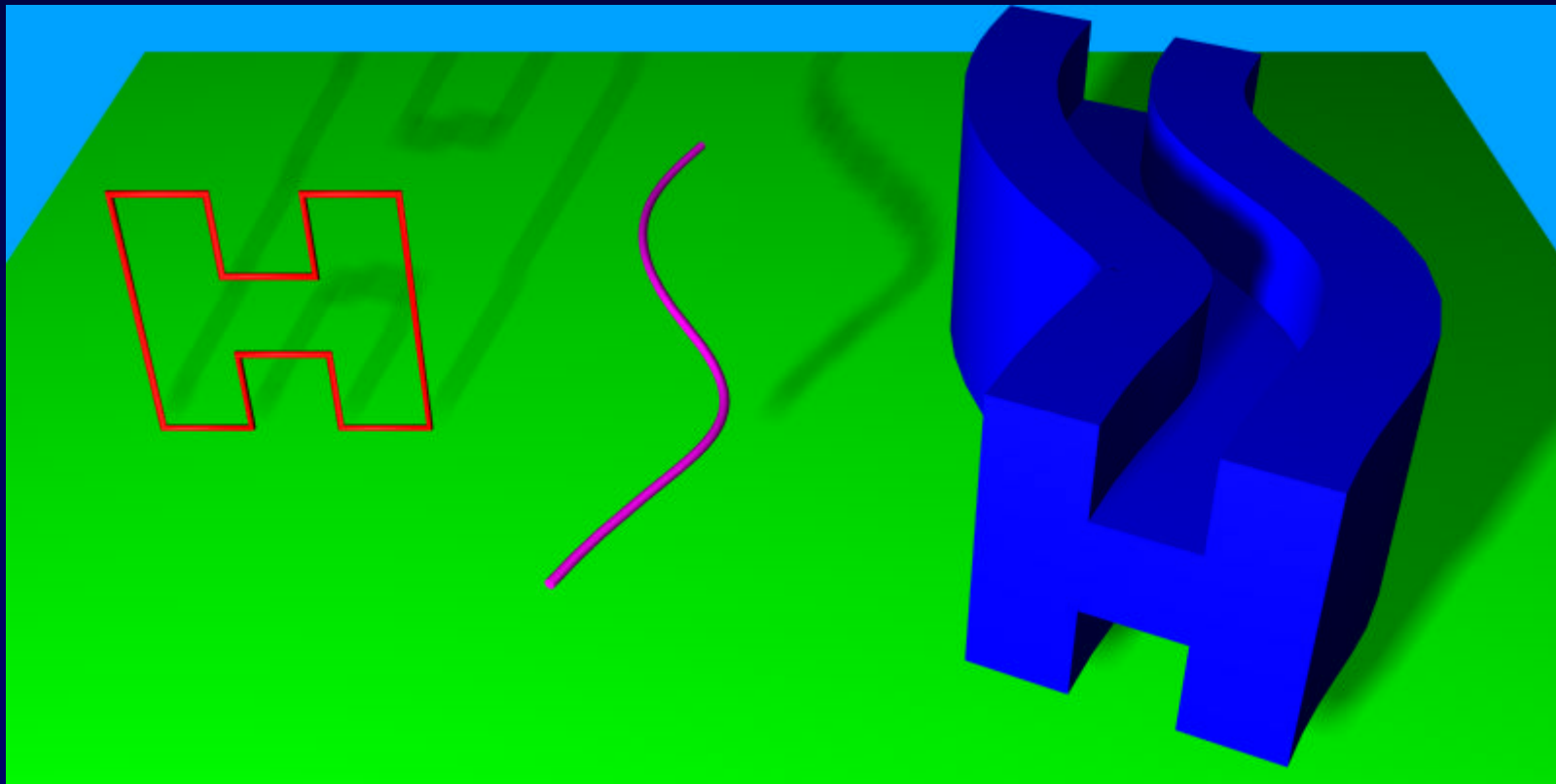
Formen:

- ◆ translational sweep
- ◆ rotational sweep
- ◆ conical sweep
- ◆ spherical sweep
- ◆ general cylinder

Translational Sweep

Definition:

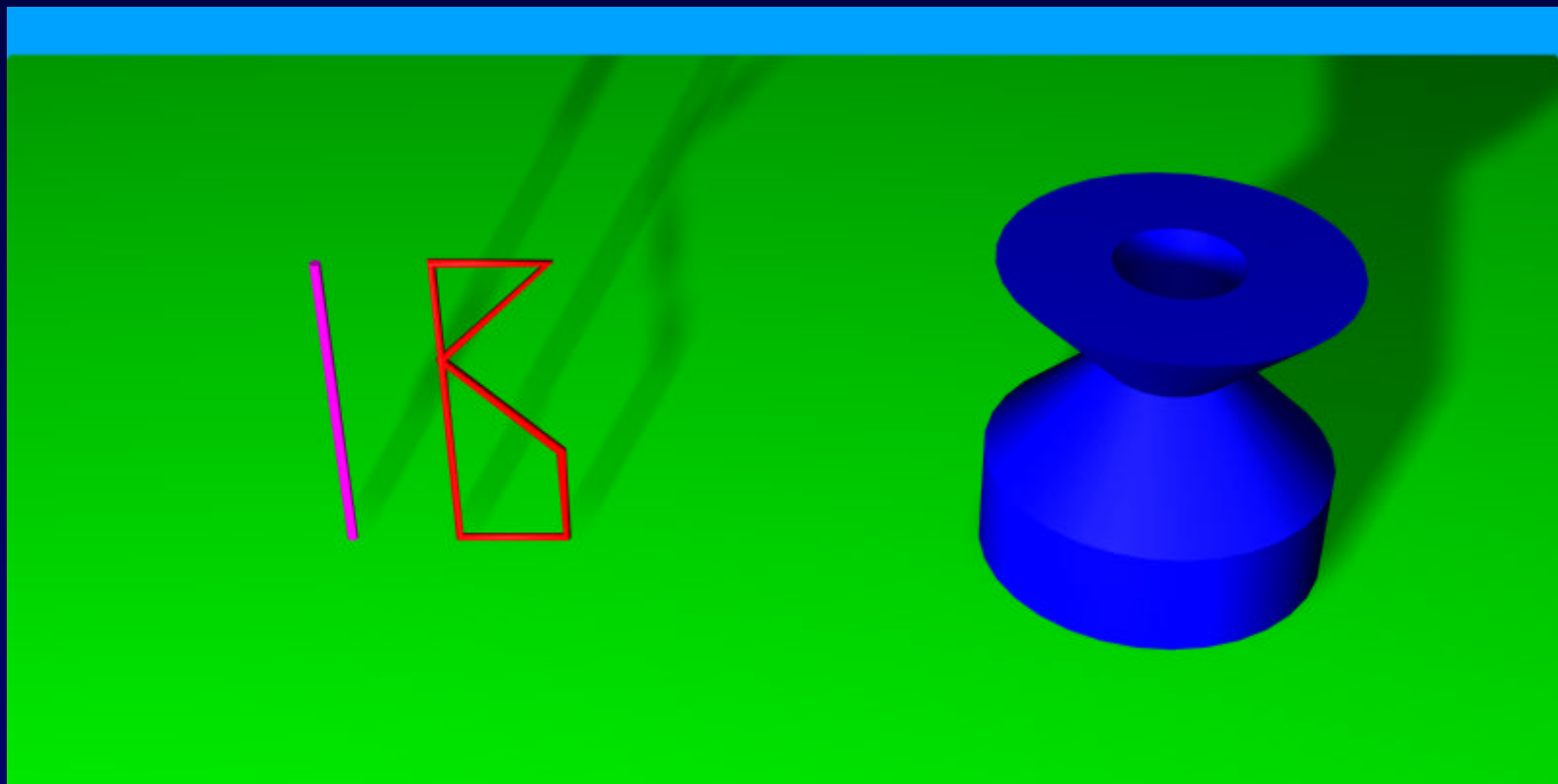
- ◆ 2D-Kontur + Translation entlang Pfad



Rotational Sweep

Definition:

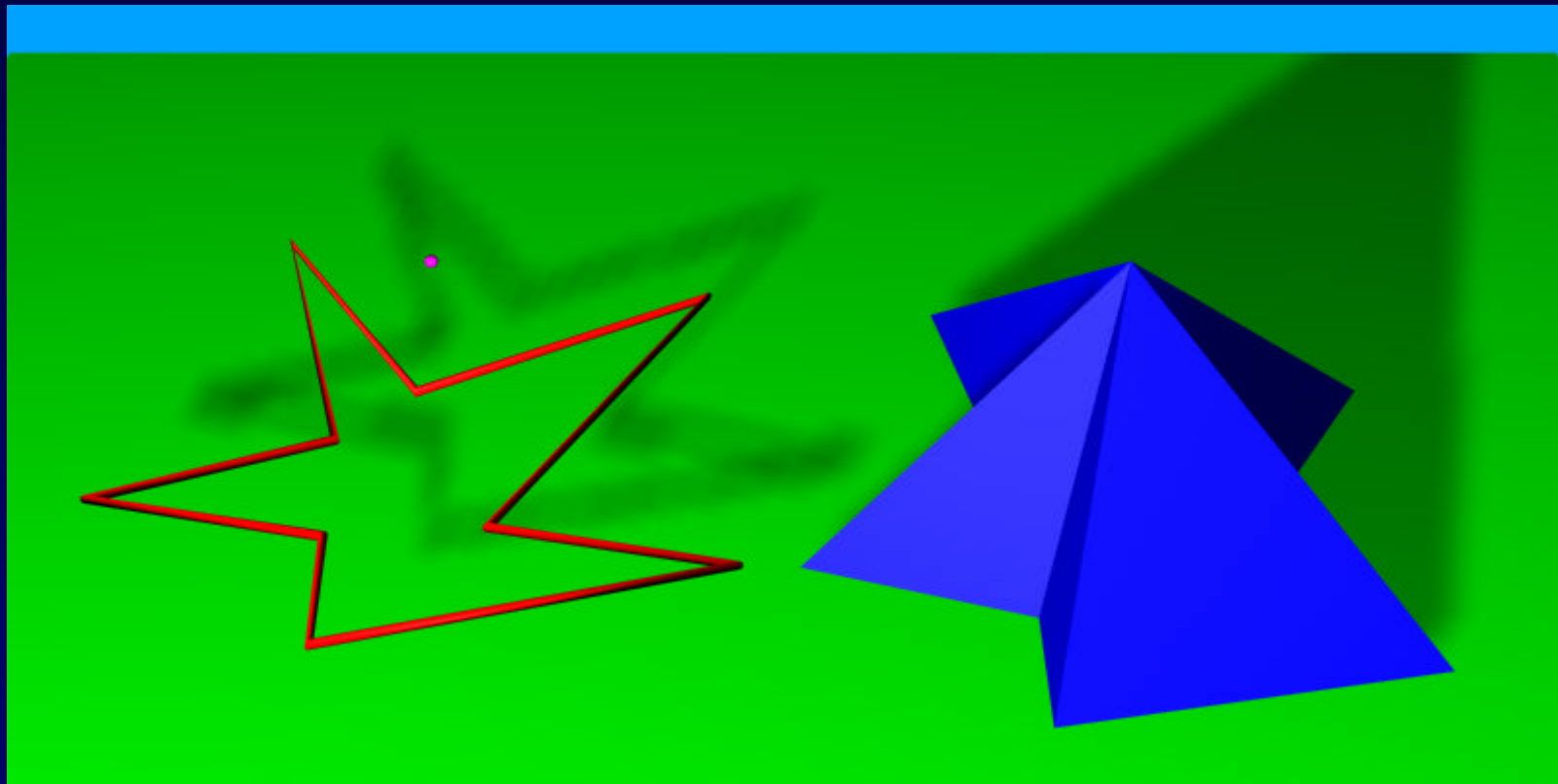
- ◆ 2D-Kontur + Rotation um Achse



Conical Sweep

Definition:

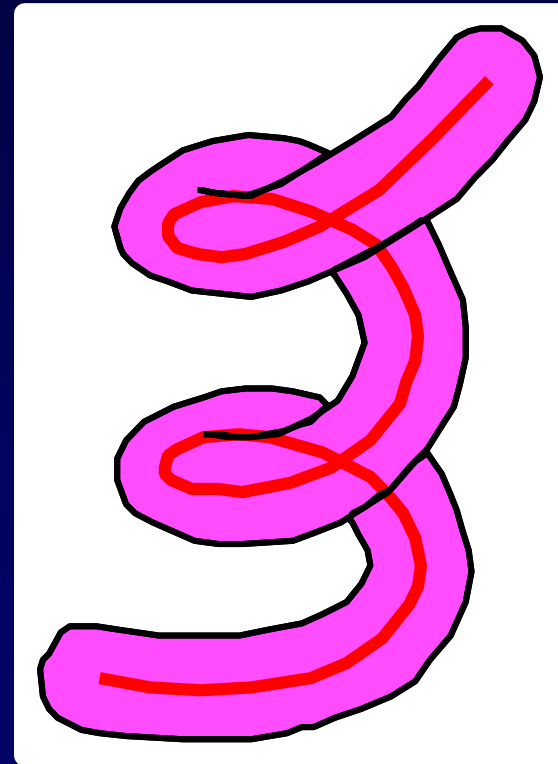
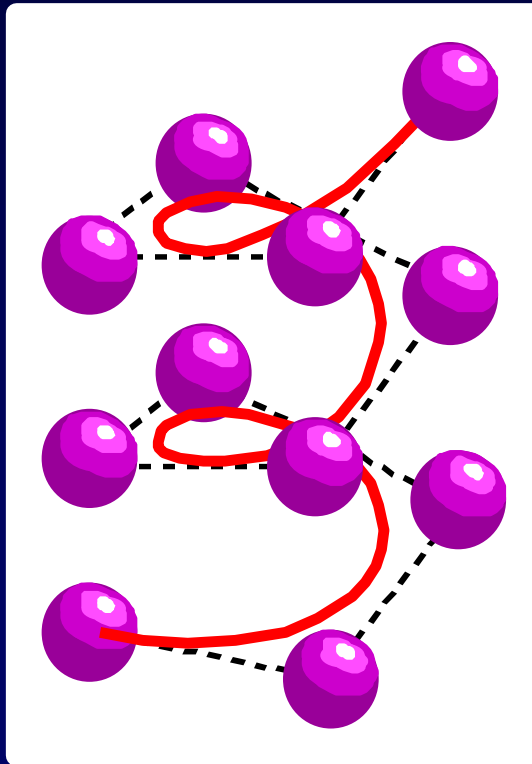
- ◆ 2D-Kontur + Verjüngung zu Punkt hin



Spherical Sweep

Definition:

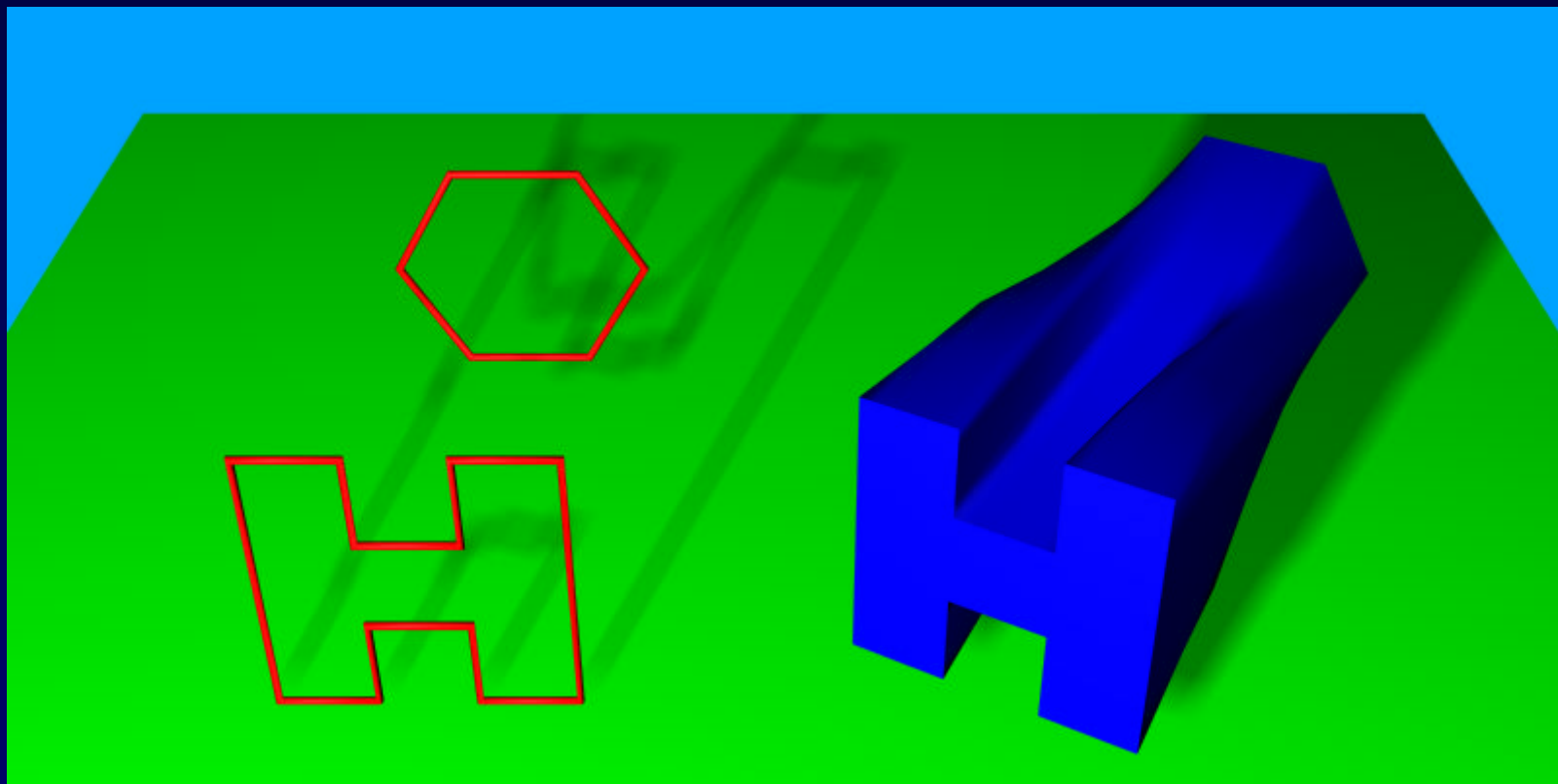
- ◆ Kugel (o.Ä.) + sweep entlang Pfad



General Cylinder

Definition:

- ◆ 2 2D-Konturen + Verbindung dazwischen



Sweeps – Repräsentation

Idee = nur Modellierung —

Analytische Form:

- ◆ abhängig von Definition (evtl. nicht-trivial)
- ◆ abhängig von notwendigen Operationen

Approximation:

- ◆ Mesh = tessellation = Zerteilung in kleine Flächenteile → Approximation durch Δ

Objekte – Überblick

Basis: Punkte, Liniensegmente, etc.

Diskrete Approximation: meshes

Erweiterung: terrains, fraktale Gebirge

Modellierung durch sweeps

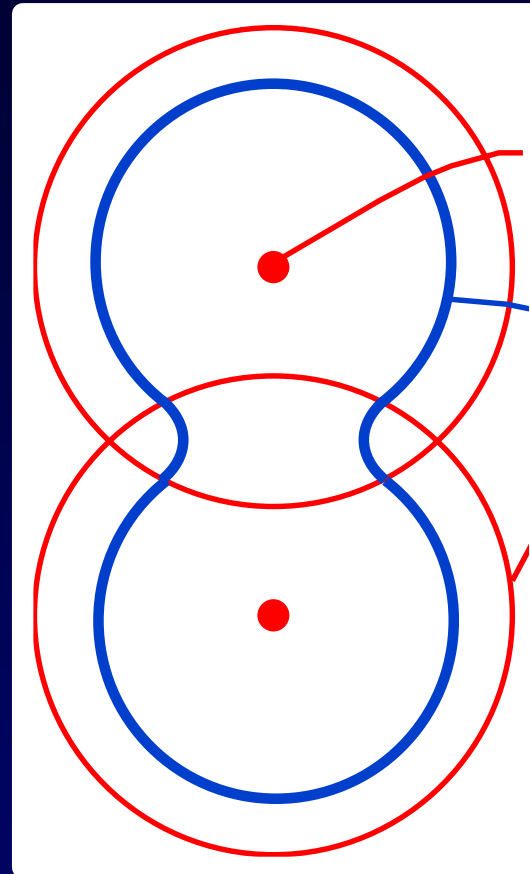
Modellierung durch soft objects

Modellierung: Partikelsysteme

Soft Objects (1)

Definition:

- ◆ Schlüsselpunkte, Einflußfunktion sowie Iso-Wert → Iso-Fläche

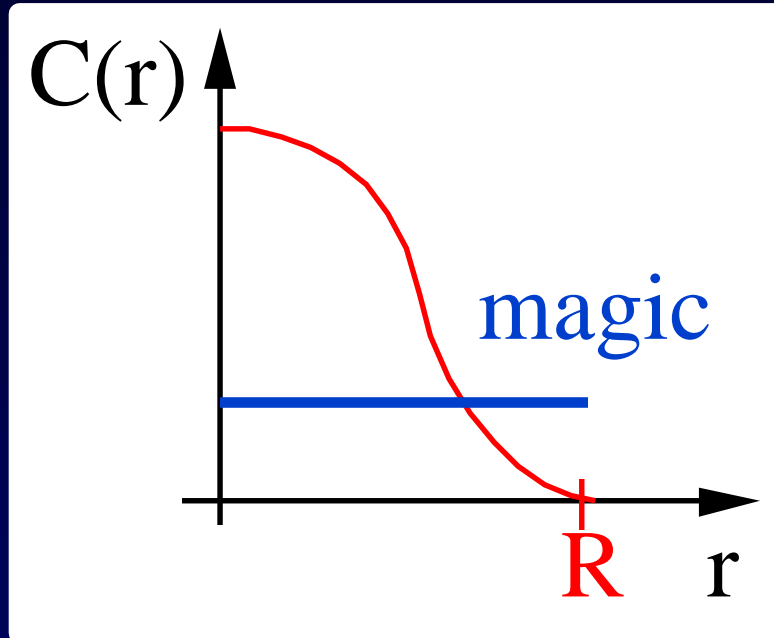


Schlüsselpunkt

Soft Object

Einflußradius

Soft Objects (2)



$$C(r) = 2 \frac{r^3}{R^3} - 3 \frac{r^2}{R^2} + 1$$

oder

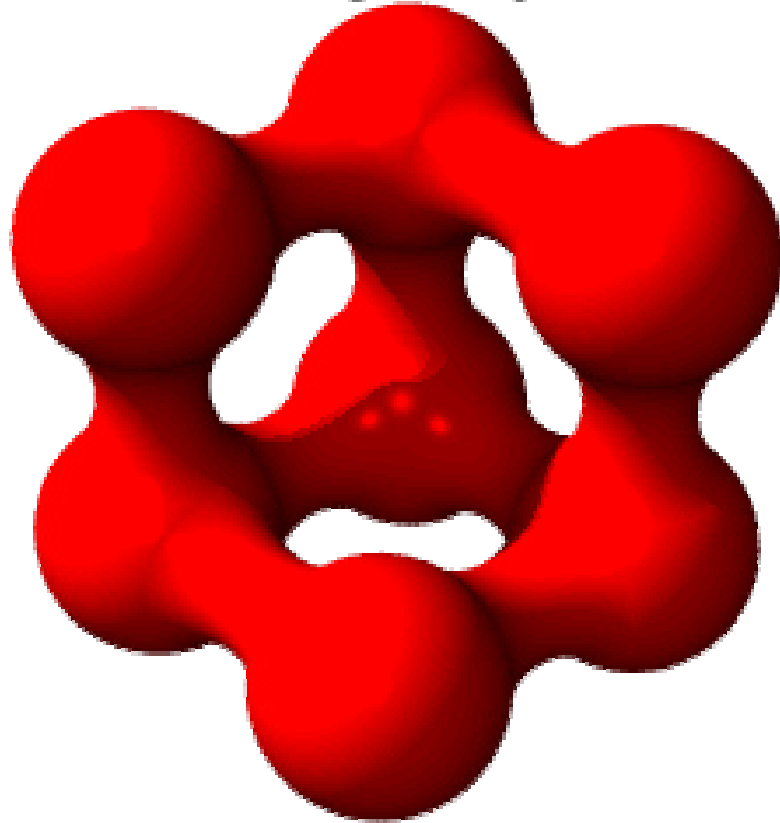
$$C(r) = a \frac{r^6}{R^6} + b \frac{r^4}{R^4} + c \frac{r^2}{R^2} + 1$$

$$a = -0.4 \quad b = 1.8 \quad c = -2.4$$

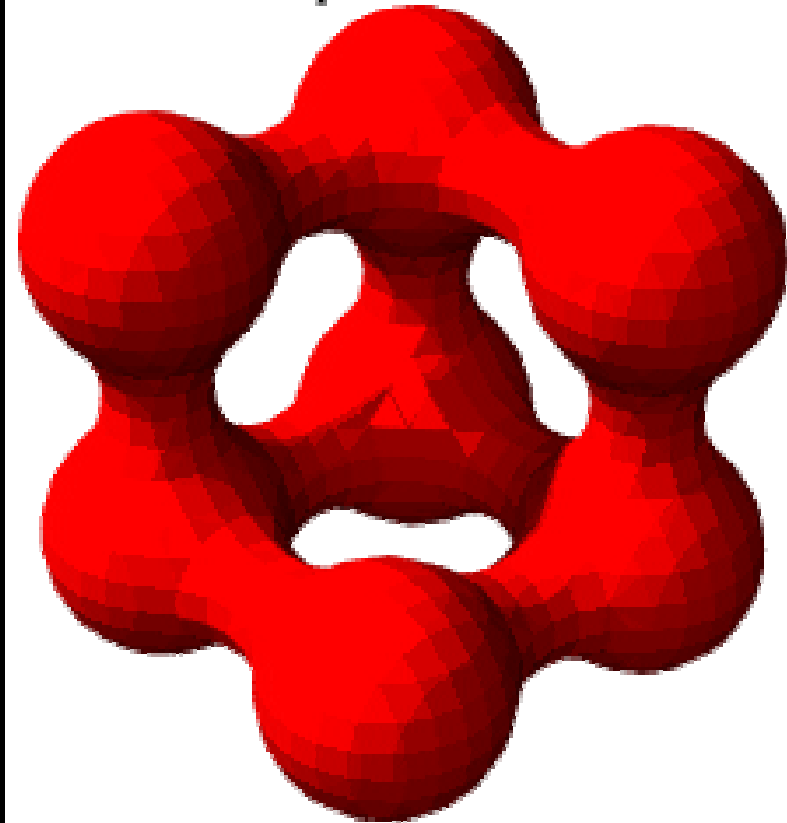
**Einfluß-Funktion fällt mit Entfernung r
Iso-Wert magic definiert Objekt**

Beispiel: Soft Objects

The Original Shape



20 Adaptive Subdivisions



Repräsentation? Analytisch? Dreiecke?

Objekte – Überblick

Basis: Punkte, Liniensegmente, etc.

Diskrete Approximation: meshes

Erweiterung: terrains, fraktale Gebirge

Modellierung durch sweeps

Modellierung durch soft objects

Modellierung: Partikelsysteme

Partikelsysteme (1)

Definition:

- ◆ Meist große Menge von Punkten
- ◆ charakteristisches Aussehen pro Partikel
- ◆ Verhalten von Partikel

Anwendungen:

- ◆ Feuer, Rauch, etc.
- ◆ Partikel-sweeps: natürliche Objekte

Partikelsysteme (2)

Pro Partikel:

- ◆ Eigenschaften:
 - ◆ Geometrische Repräsentation (einfach!)
 - ◆ Farbe, Transparenz
- ◆ Bewegung:
 - ◆ Richtung, Geschwindigkeit
 - ◆ Lebensdauer

Global:

- ◆ Anwendung von Zufallszahlen

Partikelsysteme: Ablauf

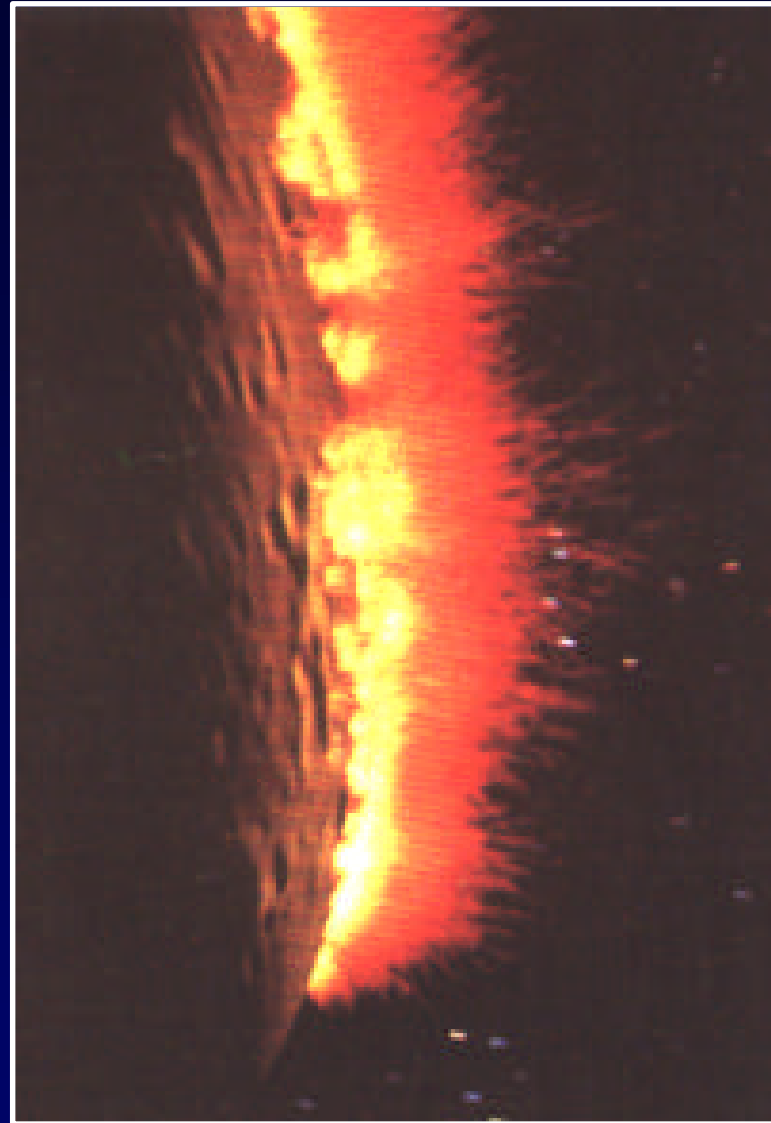
Pseudo-Code:

- ◆ Wenn Lebensdauer abgelaufen: löschen,
- ◆ Sonst Partikeldaten auf neuen Stand.
- ◆ Evtl. neue Partikel erzeugen
- ◆ Alle aktuellen Partikel darstellen

Auch hierarchisch möglich:

- ◆ Wald-System → Baum-System → Blätter

Beispiel: Partikelsysteme



WH (CGR3): CSG

Per booleschen Operationen:
Primitiva \rightarrow komplexe Objekte:

- ◆ logisches UND_2 : Durchschnitt
- ◆ logisches ODER_2 : Vereinigung
- ◆ MINUS_2 : „Wegschneiden“

