

3.1 Motivation

Es existieren verschiedenste Einflussfaktoren auf die verwendete Modellierungsmethode und Objektrepräsentation:

- Objekt existiert real oder nur in der Computerdarstellung
- Herstellung des Objektes ist eng mit seiner Visualisierung verknüpft (Modellierung und Visualisierung als Werkzeug im Herstellungsprozess)
- Genauigkeit der Repräsentation orientiert sich an Anwendung (z. B. exakte CAD-Beschreibung vs. approximative Rendering-Beschreibung)
- Interaktion fordert mehrere (auch dynamisch erzeugte) Repräsentationen (z. B. LOD, level of detail-Darstellungen)



3.1 Motivation

Die Modellierung und Repräsentation von Objekten betrifft insbesondere die folgenden Aspekte:

- Erzeugung dreidimensionaler Computergrafik-Darstellungen CAD-Interface, Digitizer, Laser-Scanner, analytische Techniken
- Wahl, Repräsentation und Verarbeitung der Datenstruktur Polygonnetz (z. B. als triangle mesh) als häufigste (aber nicht nur) Maschinenrepräsentation, CSG als Benutzerrepräsentation, die zum Rendern umgerechnet wird
- Manipulation der Repräsentation → Formänderung des Objektes z. B. Polygonnetzrepräsentation oft ungeeignet, Kontrolle über die einzelnen Punkte?



3.1 Motivation

Wir beschäftigen uns in diesem Kapitel näher mit

- a) Boundary Representation dreidimensionaler Objekten,
speziell polygonale Repräsentation
- b) CSG-Repräsentation von Objekten
- c) Raumteilungsverfahren für die Objektrepräsentation
- d) Objektrepräsentation mittels impliziter Darstellungen
- e) Szenenbeschreibung und Szenen-Management



3.2 Polygonale Repräsentation

Begriffe:

Boundary Representation:

Darstellung eines dreidimensionalen Objektes durch die das Objekt begrenzenden Flächen.

- Wir betrachten die einfachste Form der BRep, die Beschreibung eines Objektes durch ein Menge/ein Netz von planaren Polygonen/Facetten (oft Dreiecken). Diese polygonale Repräsentation stellt eine klassische Repräsentationsform in der CG dar.
- Wir entwickeln Datenstrukturen zur effizienten Speicherung und Verarbeitung großer polygonaler Netze.



3.2 Polygonale Repräsentation

Begriffe: (cont.)

Polygonale Repräsentation:

Die polygonalen Facetten stellen i. a. eine Approximation gekrümmter Flächen dar, die das Objekt begrenzen
 → piecewise linear approximation



Die Genauigkeit der Approximation (Anzahl und Größe der Polygone) kann gewählt werden, aber:

Polygonaufösung/-anzahl für genaue bzw. „glatte“ Darstellung?



3.2 Polygonale Repräsentation

Begriffe: (cont.)

Polygonale Repräsentation: (cont.)

Oft verwendete Grundregel: Polygonaufösung an die lokale Krümmung der Fläche binden, aber

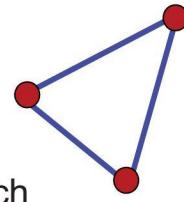
„(...) Constructing polygonal approximations to surfaces is an art, and there is no substitute for experience (...).“.

Woo, Neider, Davis
 OpenGL Programming Guide (Red Book)



3.2 Polygonale Repräsentation

Begriffe: (cont.)



Topologie:

Die Menge der Eigenschaften eines Objektes, die durch Starrkörpertransformationen **nicht** verändert werden – die Struktur des Modells.

im Beispiel: Das Polygon ist ein Dreieck

Geometrie:

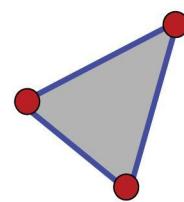
Die „Instanzierung“ der Topologie durch Spezifikation der räumlichen Lage – die Form des Modells.

im Beispiel: Die Koordinaten des Dreiecks



3.2 Polygonale Repräsentation

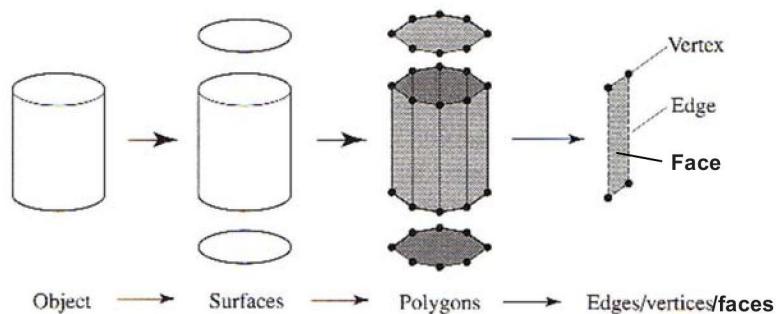
Begriffe: (cont.)



Die „Grundbausteine“:

- die Ecken/Eckpunkte oder **vertices**
- die Kanten oder **edges**
- die (planaren) Flächen oder **faces**

Hierarchie:



Bem.: Zur Speicherung und Verarbeitung von vertices, edges und faces werden geeignete Datenstrukturen benötigt!

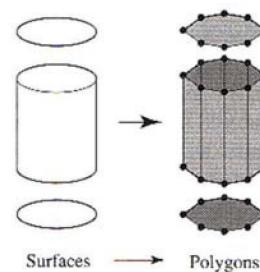


3.2 Polygonale Repräsentation

Bemerkung, Kanten:

Offensichtlich existieren in der approximierenden polygonalen Darstellung zwei Arten von Kanten:

- Kanten, die auch in Wirklichkeit existieren
→ diese sollen als Kanten sichtbar bleiben
 - Kanten, die in der Realität nicht existieren,
→ diese sollte der Renderer „verschwinden“ lassen
- 70er Jahre: Schattierungsalgorithmen
(interpolative shading algorithms)
- (siehe nächstes Kapitel!) in Hardware!
- Flat, Gouraud, Phong shading



Die Art der Kanten bestimmt in der Datenstruktur z. B. die Mehrfachspeicherung von Ecken oder Kanten.



3.2 Polygonale Repräsentation

Begriffe: (cont.)

Polyeder:

Ein Körper, der von Ebenen begrenzt wird.

Bem.: Bei unseren polygonalen Repräsentationen von dreidimensionalen Objekten handelt es sich um Polyeder.

Der Euler'sche Polyedersatz:

Ein konvexes Polyeder (d.h. ein Polyeder, das keine Löcher hat) erfüllt die Euler'sche Formel:

$$V - E + F = 2$$

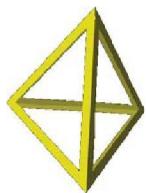
wobei V = #vertices, E = #edges, F = #faces



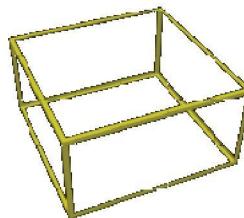
3.2 Polygonale Repräsentation

Begriffe: (cont.)

Der Euler'sche Polyedersatz $V - E + F = 2$: (cont.).



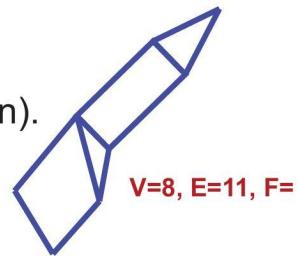
$$V=4, E=6, F=4$$



$$V=8, E=12, F=6$$

Bem.:

- Eine verallgemeinerte Euler'sche Formel behandelt auch konkave Polyeder (mit Löchern).
- Der Euler'sche Polyedersatz ist auch auf polygonale Netze anwendbar.



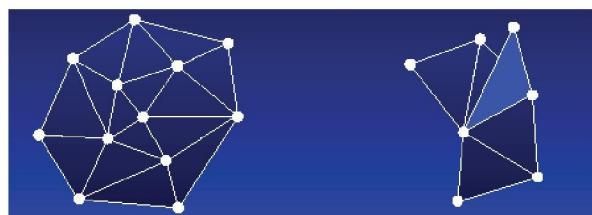
3.2 Polygonale Repräsentation

Begriffe: (cont.)

Mannigfaltigkeiten:

Sehr viele der existierenden, auf polygonalen Repräsentationen arbeitenden Algorithmen setzen voraus, dass das Objekt eine *Mannigfaltigkeit* ist:

- Die Umgebung einer Ecke ist eineindeutig (homeomorph) auf einen Kreis oder Halbkreis abbildbar.
- Eine Kante gehört zu nicht mehr als zwei faces.



3.2 Polygonale Repräsentation

Was wollen wir an polygonalen Netzen modifizieren?

- Wir beseitigen Mess - und Abtastfehler durch Glätten.
- Wir verfeinern Netze, um genauer modellieren und mehr Details darstellen zu können.
- Wir versuchen Netze auszudünnen (z. B. Kompression zur Web-Übertragung oder level of detail Darstellung).

Welche Fragen sind für die Wahl der Datenstrukturen wichtig?

- Wie trennen wir die Information über Topologie und Geometrie?
- Wie speichern wir die Information über Ecken, Kanten und Faces?
- Welche Operationen müssen zur Verfügung stehen?



3.2 Polygonale Repräsentation

Allgemeine Operatoren:

- Ecken, Kanten, Faces auswählen, einfügen, löschen, verschweißen, rotieren, skalieren, ...
- ganze Netze verschweißen, ...

Topologische Operatoren:

- Finde alle Kanten einer bestimmten Ecke!
- Finde alle Faces, die eine bestimmte Kante oder Ecke gemeinsam haben!
- Bestimme die Ecken, die von einer Kante verbunden werden!
- Bestimme die Kanten einer Face!
- Konsistenzprüfung – fehlt etwas?, ist redundante Info enthalten?



3.2 Polygonale Repräsentation

Organisationsprinzipien:

Je nach Anwendungsfall können die folgenden Fälle auftreten:

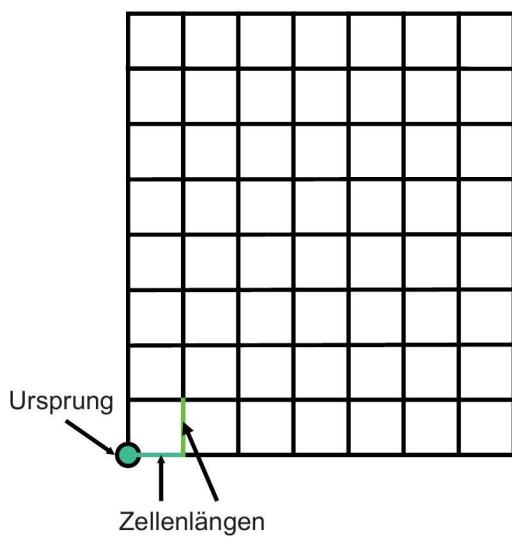
- regelmäßige Strukturen mit impliziter Topologie und Geometrie
- gemischte Strukturen mit impliziter Topologie und expliziter Geometrie
- „unorganisierte“ Daten mit expliziter Topologie und expliziter Geometrie – (allgemeine) polygonale Netze



3.2 Polygonale Repräsentation

Organisationsprinzipien: (cont.)

Regelmäßige Strukturen:



- rechtwinkliges, gleichmäßiges Gitter
- Topologie durch Anzahl der Punkte in x- und y-Richtung (und z-Richtung) gegeben
- Geometrie durch Angabe von Ursprung und Zellenlängen gegeben

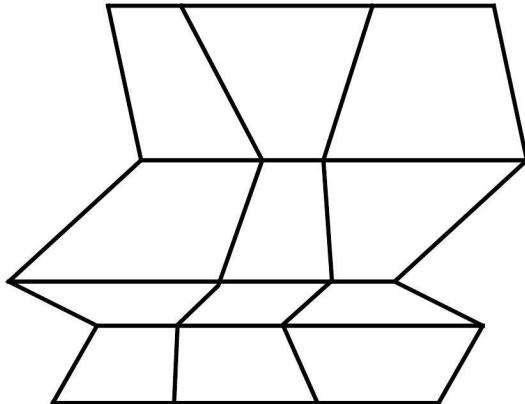
Bsp.: Ursprung = (-1,-1),
 $x\text{Dim} = 0.2$, $y\text{Dim} = 0.1$,
 $n_x = 10$, $n_y = 20$
 Koordinaten des rechten oberen Punktes?



3.2 Polygonale Repräsentation

Organisationsprinzipien: (cont.)

Gemischte Strukturen: (Bsp.)



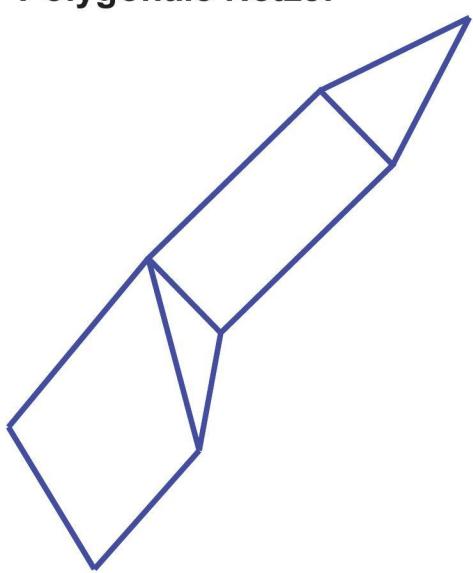
- Topologie durch Anzahl der Punkte in x- und y-Richtung (und z-Richtung) gegeben
- Geometrie durch explizite Angabe der Punktkoordinaten gegeben



3.2 Polygonale Repräsentation

Organisationsprinzipien: (cont.)

Polygonale Netze:



- Topologie und Geometrie muss explizit gespeichert werden!
- Das Problem ist die Topologie, die Geometrie ist Liste von Punktkoordinaten!

Möglichkeiten:

- *explizite Speicherung*
- *Eckenliste*
- *Kantenliste*

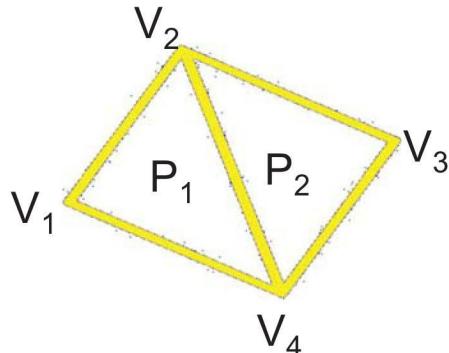


3.2 Polygonale Repräsentation

Polygonale Netze, explizite Speicherung:

- Jedes Polygon wird durch eine Liste seiner Eckpunktkoordinaten repräsentiert.
- Zwischen jedem Paar von Ecken ist eine Kante, auch zwischen letzter und erster Ecke.

Beispiel:



$$\begin{aligned} P_1 = & ((V_{2x}, V_{2y}, V_{2z}), \\ & (V_{1x}, V_{1y}, V_{1z}), \\ & (V_{4x}, V_{4y}, V_{4z})) \end{aligned}$$

$$\begin{aligned} P_2 = & ((V_{4x}, V_{4y}, V_{4z}), \\ & (V_{3x}, V_{3y}, V_{3z}), \\ & (V_{2x}, V_{2y}, V_{2z})) \end{aligned}$$



3.2 Polygonale Repräsentation

Polygonale Netze, explizite Speicherung: (cont.)

Bemerkungen:

- Speicheraufwendige Darstellung
- Die Koordinaten von Ecken werden mehrfach aufgeführt.
- Es gibt keine Speicherung gemeinsamer Ecken und Kanten.

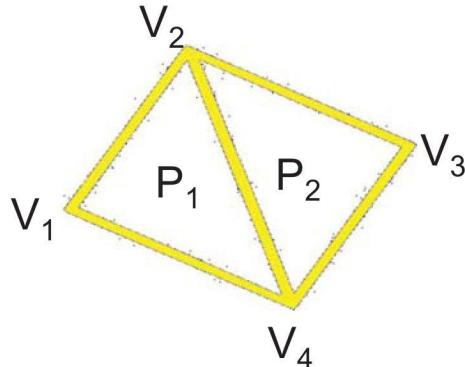


3.2 Polygonale Repräsentation

Polygonale Netze, Eckenliste:

- Alle benötigten Ecken werden in einer Punktliste abgelegt.
- Ein Polygon wird als Liste von Indizes (Verweisen) in die Punktliste definiert.

Beispiel:



$$V = (V_4, V_2, V_1, V_3)$$

$$P_1 = (3, 1, 2)$$

$$P_2 =$$

Auf Durchlaufrichtung achten!



3.2 Polygonale Repräsentation

Polygonale Netze, Eckenliste: (cont.)

Bemerkungen:

- Jede Ecke wird nur einmal gespeichert.
- Die Geometrie kann unabhängig von der Topologie verändert werden.
- Gemeinsame Kanten und Ecken von Polygonen zu finden ist immer noch schwer!
- Bei der grafischen Darstellung werden Kanten mehrfach dargestellt!

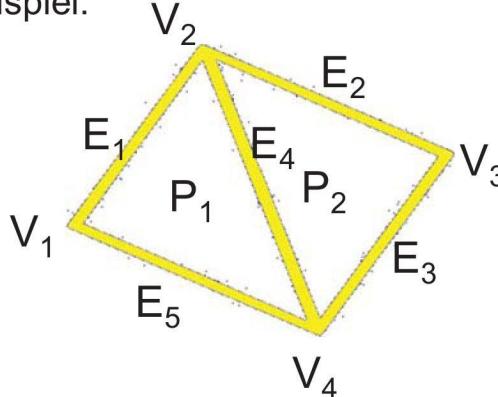


3.2 Polygonale Repräsentation

Polygonale Netze, Kantenliste:

- Alle benötigten Ecken werden in einer Punktliste abgelegt.
- Die Kanten werden in einer eigenen Liste abgelegt.
- Ein Polygon wird als Liste von Indizes in die Kantenliste definiert.

Beispiel:



$$V = (V_1, V_2, V_3, V_4)$$

$$\begin{aligned}E_1 &= (2, 1, P_1, N), \\E_2 &= (3, 2, P_2, N), \\E_3 &= (4, 3, P_2, N), \\E_4 &= (4, 2, P_1, P_2), \\E_5 &= (1, 4, P_1, N), \\E &= (E_1, E_2, E_3, E_5, E_4)\end{aligned}$$

$$P_1 = (1, 4, 5), P_2 =$$



3.2 Polygonale Repräsentation

Polygonale Netze, Kantenliste: (cont.)

Bemerkungen:

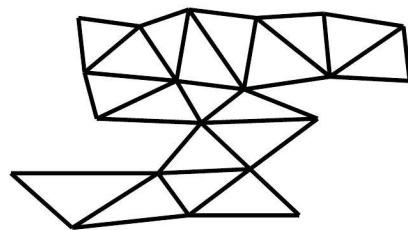
- Die Geometrie kann unabhängig von der Topologie verändert werden.
- Gemeinsame Kanten von Polygonen sind leichter, gemeinsame Ecken immer noch schwer zu finden.
- Bei der grafischen Darstellung werden Kanten nicht mehrfach dargestellt!



3.2 Polygonale Repräsentation

Dreiecksnetze

„(...) the basic shape is the triangle (...)“



- Eine Spezialform polygonaler Netze!
- Die Grundform, die am Ende der Grafikpipeline ankommt ist ein Dreieck oder eine strukturierte Menge von Dreiecken.
- Für Dreiecksnetze existieren **spezielle Datenstrukturen** (z. B. **Triangle Strip**, **Triangle Fan**), welche die Topologie implizit codieren/speichern, einen geringeren Specheraufwand verursachen und bessere Performance auf der Grafikhardware hervorrufen. (z. B. unterstützt in OpenGL, Direct3D und Java3D)



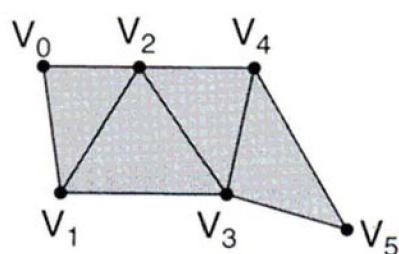
3.2 Polygonale Repräsentation

Dreiecksnetze (cont.)

Triangle Strip:

- Folge von mindestens drei Eckpunkten
- je drei aufeinander folgende Eckpunkte definieren ein Dreieck
- $n+2$ Eckpunkte definieren n Dreiecke

Beispiel: $(V_0, V_1, V_2, V_3, V_4, V_5)$



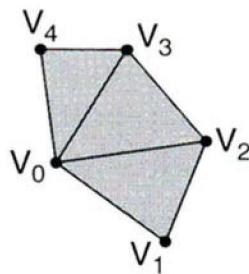
3.2 Polygonale Repräsentation

Dreiecksnetze (cont.)

Triangle Fan:

- Folge von mindestens drei Eckpunkten
- je zwei aufeinander folgende Eckpunkte definieren mit dem ersten Eckpunkt ein Dreieck
- $n+2$ Eckpunkte definieren n Dreiecke

Beispiel: $(V_0, V_1, V_2, V_3, V_4)$



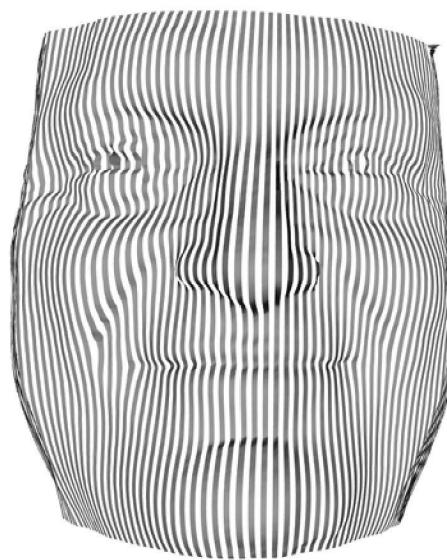
3.2 Polygonale Repräsentation

Dreiecksnetze (cont.)

Beispiele: Triangle Strips



2297 Strips mit einer Durchschnittslänge von 3.94, längster Strip 101.



134 Strips mit einer Länge von 390.



3.2 Polygonale Repräsentation

Bemerkung, Datenstruktur:

Praktische Datenstrukturen beinhalten nicht nur Topologie und Geometrie der polygonalen Darstellung, sondern auch Attribute, die für Anwendungen und Renderer benötigt werden. Dies sind:

- Flächen/Face-Attribute:
Dreieck?, Fläche, Flächennormale, Koeffizienten der Fläche, konvex?, Löcher?
 - Kanten-Attribute:
Länge, Kante zwischen Polygonen oder Oberflächen?, Abschlußkante?
 - Eckpunkt-Attribute:
beteiligte Polygone, Eckpunktnormale, Texturkoordinaten
- ein Mittel der Flächennormalen der an der Ecke angrenzenden faces



3.2 Polygonale Repräsentation

Erzeugung polygonaler Objekte:

Manuelle und semi-manuelle Verfahren

- „manuelles“ Verschieben von (Gruppen von) Eckpunkten mittels dreidimensionaler Eingabegeräte oder Schnittstellen
 - komplex, schwer handhabbar
 - nur für einfache Objekte bzw. für einfache „Manipulationen“ geeignet
- 3D-Digitizer
manuelles Anbringen von Punkten auf Objekten, die mittels Digitizer zu Polygon-Eckpunkten werden sollen

Beispiel: Netze über Objektoberflächen „ziehen“
→ erste 3D-Darstellungen von Karosserien (1974)



3.2 Polygonale Repräsentation

Erzeugung polygonaler Objekte: (cont.)

Automatische Verfahren

hier: Laserscanner

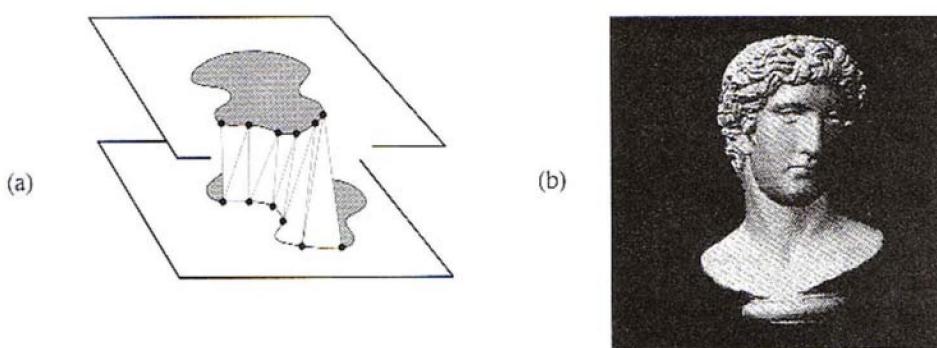
- Objekt wird rundherum scheibenweise mit einem Laserstrahl abgetastet; dieser mißt den Abstand zur Objektoberfläche
- aus den gemessenen 2D-Konturen werden mittels eines „skinning“-Algorithmus, der geeignet benachbarte Punkte verbindet, Dreiecksflächen erzeugt (Abb. a))
- dieser Ansatz tendiert dazu, (zu) viele Dreiecke zu erzeugen! (Abb. b): 400.000 Dreiecke)
- Anwendung: z. B. virtuelles Kaufhaus, Maßanzüge, ...



3.2 Polygonale Repräsentation

Erzeugung polygonaler Objekte: (cont.)

Automatische Verfahren (cont.)



- Problem: ist das Objekt stellenweise „zu konkav“, gibt es Flächen, die vom Laserstrahl nicht erfaßt werden können



3.2 Polygonale Repräsentation

Erzeugung polygonaler Objekte:

Mathematische Verfahren

Erzeugung von polygonalen Darstellungen über zugrunde liegende Kurven und Flächen → CAD-Anwendungen

Vorteile:

- Benutzer arbeitet mit high-level Objektbeschreibung
- Objektform ist direkt mit mathematisch exakter Objektbeschreibung gekoppelt

Beispiele: Revolution, bikubische Parameterflächen, ...



3.2 Polygonale Repräsentation

Erzeugung polygonaler Objekte:

Prozedurale Verfahren

Ein verbreitetes prozedurales Verfahren zur Erzeugung polygonaler Objekte sind die so genannten *Fractals*.

Fractals gehen in ihrem theoretischen Ansatz auf die Mandelbrot-Geometrie zurück und werden im Bereich der Computergrafik für die Modellierungen von terrain models eingesetzt.

Ein dabei wesentlicher Aspekt war und ist das Pilotentraining in Simulatoren.



3.3 CSG-Repräsentation

Eigenschaften der Constructive Solid Geometry (CSG):

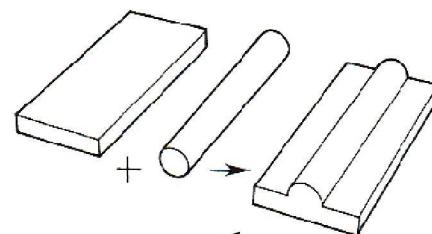
- oft im Konstruktionsbereich verwendet
- Volumenrepräsentation dreidimensionaler Objekte
- Repräsentation komplexer Objekte durch Zusammensetzen einfacher Grundobjekte (Primitive) mittels boolescher Mengenoperationen und Transformationen
- geometrische Primitive: Kugel, Kegel, Zylinder, Quader, ...
- Nachteil: Darstellung von CSG-Objekten erfordert spezielle Rendering-Techniken oder Umwandlung in eine polygonale Repräsentation



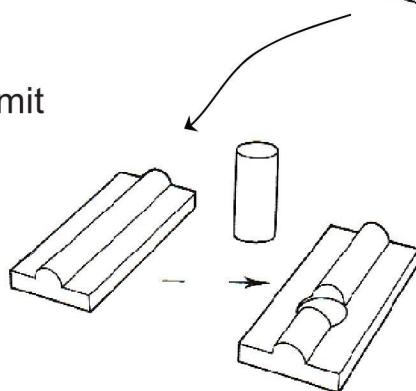
3.3 CSG-Repräsentation

Beispiel:

Vereinigung von Quader und Zylinder ...



... anschließend Differenzbildung mit einem weiteren Zylinder ...



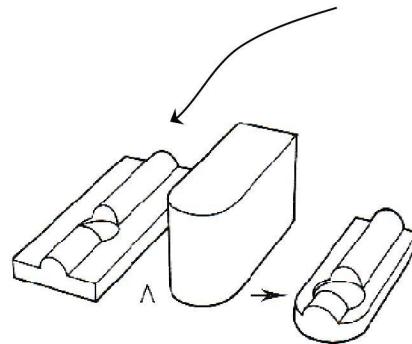
3.3 CSG-Repräsentation

Beispiel: (cont.)

... anschließend Schnitt mit einem Objekt, das aus der Vereinigung eines Quaders mit einem weiteren Zylinder entstanden ist.

Bemerkung: Ein bestimmtes Objekt kann durch verschiedenste CSG-Operationen definiert werden.

HIER: Schritte 2 und 3 könnten auch vertauscht werden!



Intern wird eine CSG-Repräsentation durch einen Baum verwaltet, dessen innere Knoten als Information den zu verwendenden booleschen Operator und Angaben über die räumliche Beziehung zwischen ihren Söhnen enthalten. Die Blätter des Baumes enthalten jeweils den Namen eines Primitivs und dessen Dimension.



3.4 Raumteilungsverfahren

Bei der Repräsentation eines Objekts mittels Raumteilungsverfahren (space subdivision techniques) wird der Objektraum in einzelne Elemente zerlegt. Dabei wird für jedes Element vermerkt, ob der zugehörige Raum durch das Objekt belegt ist oder nicht.

Standardverfahren:

- Unterteilung des Raumes durch ein festes, regelmäßiges Gitter in Zellen identischer Geometrie
- Im 3D-Raum erhält man würfelförmige Zellen, die als **Voxel** (volume element) bezeichnet werden.
→ Analogie zu **Pixel** (picture element) im 2D



3.4 Raumteilungsverfahren

Vorteile:

- Es ist sehr einfach zu bestimmen, ob ein gegebener Punkt innerhalb oder außerhalb eines Objekts liegt.
- Es ist einfach zu ermitteln, ob zwei Objekte einander berühren.
- Die Repräsentation eines gegebenen Objektes ist eindeutig.

Nachteile:

- Es können nur teilweise belegte Zellen existieren.
 - Objekte können im allgemeinen nur approximiert werden.
 - Für eine Auflösung von n Voxeln in jeder Dimension werden n^3 Voxel benötigt. Diese Art der Objektrepräsentation ist also **extrem specheraufwendig!**
- günstigere Repräsentation durch Octrees



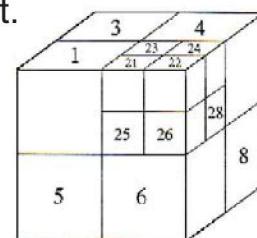
3.4 Raumteilungsverfahren

Octrees

Ein Octree ist eine hierarchische Datenstruktur zur effizienten Speicherung einer ungleichmäßigen Unterteilung des 3D-Raums.

Prinzip:

- Initiales Element ist ein Würfel, der den gesamten Objektraum umfasst und die Zustände belegt / nicht belegt annehmen kann.
- Der Belegungszustand des Würfels wird bestimmt. Sollte er **nur teilweise** vom Objekt belegt sein, so wird er entlang jeder Dimension halbiert.
- Auf jeden entstehenden (Teil-)Würfel wird dieses Verfahren rekursiv so lange angewandt, bis es stoppt oder die gewünschte Auflösung erreicht ist.



3.4 Raumteilungsverfahren

Octrees (cont.)

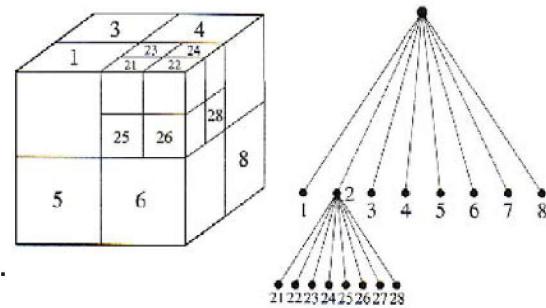
In einem Octree besitzen alle inneren Knoten genau acht direkte Nachfolger.

Die Wurzel des Baumes repräsentiert den initialen Würfel.

Bei einer Unterteilung wird für jeden entstehenden Teilwürfel nach einem festen Numerierungsschema ein neuer Knoten als Sohn eingefügt.

Jedes Blatt speichert den Zustand des zugehörigen (Teil-)Würfels.

Jeder innere Knoten repräsentiert einen nur teilweise belegten Würfel.

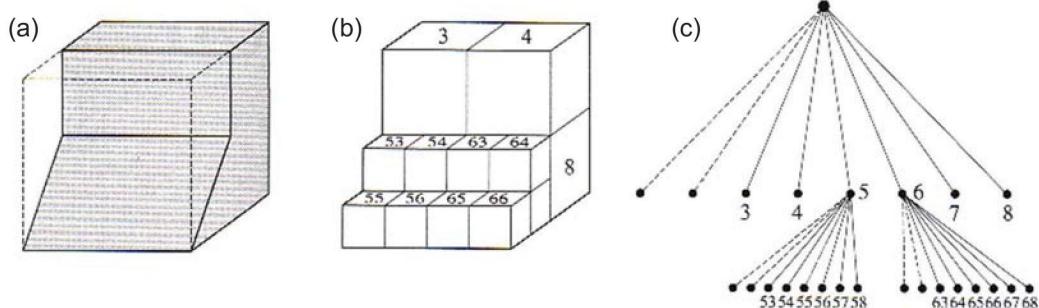


3.4 Raumteilungsverfahren

Octrees (cont.)

Beispiel: Repräsentation eines 3D-Objekts durch einen Octree

- (a) Objekt, eingebettet in den initialen Würfel
- (b) Repräsentation des Objektes bei maximal zweimaliger Unterteilung des Raumes
- (c) zugehörige Octree-Datenstruktur



3.4 Raumteilungsverfahren

Octrees (cont.)

Wesentlich häufiger als zur direkten Repräsentation von Objekten werden Octrees zur räumlichen Unterteilung der Objekte innerhalb einer Szene eingesetzt.

- Die einzelnen Objekte werden hierbei durch Standard-Datenstrukturen repräsentiert (z. B. polygonal).
- Der Informationsgehalt der Zellen des Octrees wird von dem binären „belegt“-Zustand auf eine Liste von Objekten (bzw. Polygone, ...), die in der Zelle enthalten sind, erweitert.

Dies führt zu einer wesentlichen Beschleunigung von Algorithmen, die lokal auf einzelnen Regionen des Raumes arbeiten (z. B. ray tracing).



3.4 Raumteilungsverfahren

Quadtrees

Das beim Octree realisierte Prinzip der Unterteilung des dreidimensionalen Raumes kann allgemein auch auf den n-dimensionalen Raum angewandt werden.

Für den Fall $n = 2$ erhält man eine Unterteilung der Ebene in Form eines sogenannten **Quadtrees**, bei dem jeder innere Knoten des Baumes genau vier Söhne besitzt.

Historisch gesehen sind Quadtrees die älteren Bäume. Sie wurden bereits in den späten 60er Jahren erstmalig verwendet. Octrees wurden von den Quadtrees abgeleitet und kamen erst ab Ende der 70er, Anfang der 80er Jahre zum Einsatz.

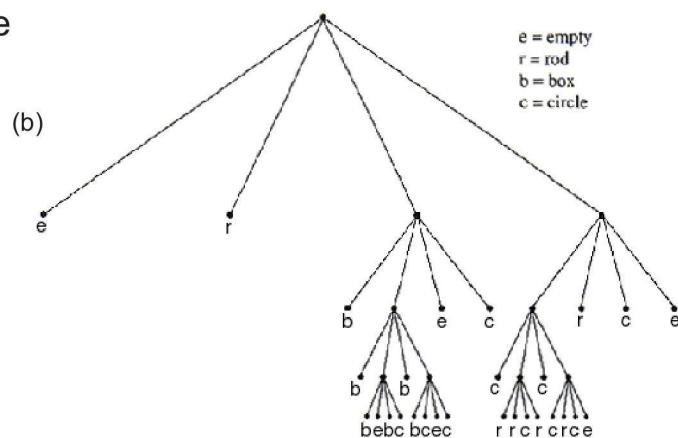
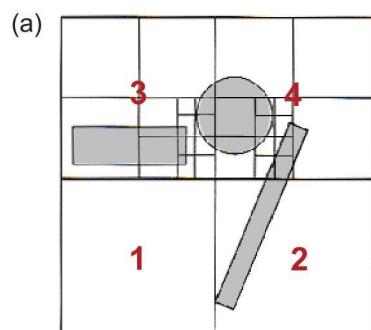


3.4 Raumteilungsverfahren

Quadtrees (cont.)

Beispiel: Unterteilung einer 2D-Szene durch einen Quadtree

- (a) Unterteilung des Raumes, bis die Zellen maximal eine Objektreferenz enthalten
- (b) zugehörige Quadtree -Datenstruktur



3.4 Raumteilungsverfahren

Binary Space-Partitioning Trees (BSP-Bäume)

Octrees und Quadtrees unterteilen rekursiv für jede Stufe den Raum wechselseitig senkrecht zueinander in allen Dimensionen.

Ein BSP-Baum liefert eine alternative Repräsentation, indem der Raum rekursiv auf jeder Stufe mittels einer beliebigen Ebene (Gerade) in **zwei** Unterräume geteilt wird.

Dies wird benutzt:

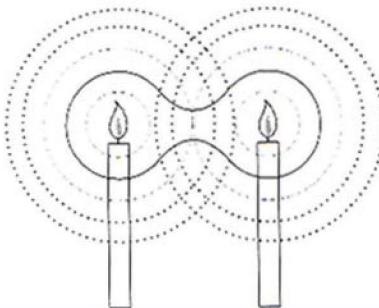
- zur Darstellung beliebiger Polyeder
- zur Unterteilung von Szenen (an keinerlei Raster gebunden!)
- zur Bestimmung der *Sichtbarkeit von Objekten*
(welche Objekte können andere verdecken?)



3.5 Implizite Beschreibungen

Idee: Beschreibung von Objekt-Flächen bzw. -Volumina als Isoflächen in Skalarfeldern. Die Skalarfelder ihrerseits entstehen kontrolliert durch erzeugende Primitive (Funktionen).

Beispiel: Punkthitzequellen erzeugen isoliert voneinander sphärische Feldfunktionen. In Kombination entsteht durch Superposition ein globales Skalarfeld.



3.5 Implizite Beschreibungen

Wir betrachten nun beispielhaft sogenannte diskrete **Metaballs**:

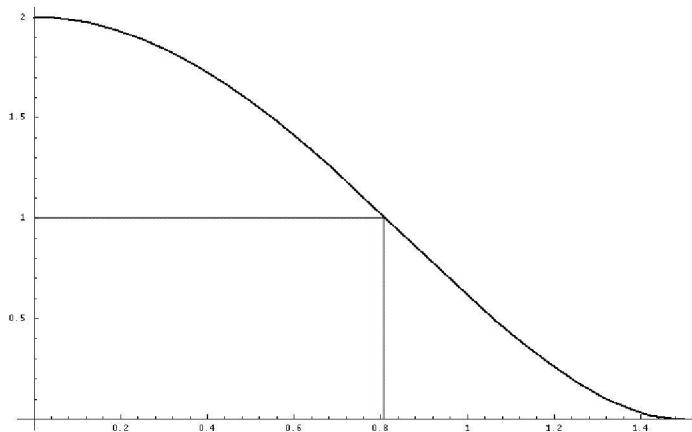
Ausgangspunkt zur Definition eines Metaballs ist eine Feldfunktion $F_d: \mathbb{R}^3 \rightarrow \mathbb{R}$ um einen Mittelpunkt $P \in \mathbb{R}^3$. F_d ist hierbei im wesentlichen die Komposition einer monoton fallenden Funktion - der sogenannten Einflussfunktion - und eines Abstandsmaßes (hier euklidisches Maß) zwischen der freien Variablen $x \in \mathbb{R}^3$ und dem Mittelpunkt P . Der Index d deutet an, daß die Feldfunktion um einen diskreten Mittelpunkt definiert ist.

Als eigentlichen Metaball bezeichnet man in diesem Zusammenhang diejenige Fläche, die bei einem vorgegebenen Isowert vom Skalarfeld von F_d implizit definiert wird.



3.5 Implizite Beschreibungen

Beispiel einer Einflussfunktion:



Damit lautet die radialsymmetrische Feldfunktion eines diskreten Metaballs im Punkt P:

$$F_d : \mathbb{R}^3 \rightarrow \mathbb{R} \text{ mit } F_d(x) = f(\|x-P\|_2)$$



3.5 Implizite Beschreibungen

Existieren mehrere Metaballs mit Feldfunktionen $F_d(i; \bullet) : \mathbb{R}^3 \rightarrow \mathbb{R}$ um Mittelpunkte $P_i \in \mathbb{R}^3$ ($i=1, \dots, n$), so überlagern sich ihre Einflüsse nach dem Superpositionsprinzip; die resultierende Feldfunktion des entstehenden Skalarfeldes ergibt sich dann einfach als Summe der einzelnen Feldfunktionen:

$$F : \mathbb{R}^3 \rightarrow \mathbb{R}$$

$$F(x) = \sum_{i=1}^n F_d(i; x)$$

Ist c eine vorgegebene Konstante kleiner als der maximal im Skalarfeld vorkommende Wert, so definiert die Menge aller $x \in \mathbb{R}^3$ mit $F(x) = c$ eine Isofläche S vom Niveau c. Die Fläche S wird durch diese Gleichung implizit definiert. c wird auch Isowert genannt, S auch Niveaumenge.



3.5 Implizite Beschreibungen

Das *Erscheinungsbild* der zugehörigen Isofläche ist bei sinnvoller Anordnung der Mittelpunkte und Benutzung einfacher Feldfunktionen noch gut kontrollierbar.

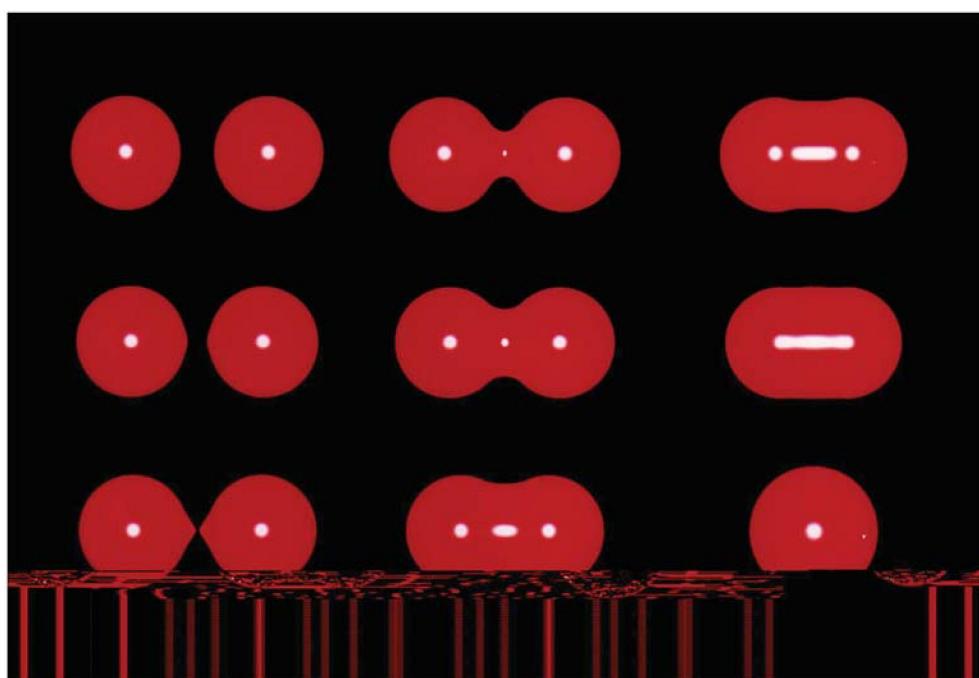
Die nachfolgende Abbildung zeigt Isoflächen, die von jeweils zwei radialsymmetrischen Feldfunktionen erzeugt werden.

Dabei wurden die beiden zugehörigen Mittelpunkte immer weiter aufeinander zu bewegt und schließlich zur Deckung gebracht.

Man erkennt deutlich den sogenannten *Verschmelzungseffekt*, bei dem die separierten Isoflächen glatt (in diesem Fall C¹-stetig) ineinander übergehen, falls der Abstand der Mittelpunkte klein genug wird. Bei umgekehrter Reihenfolge wird dementsprechend von einem *Zerreisseffekt* gesprochen.

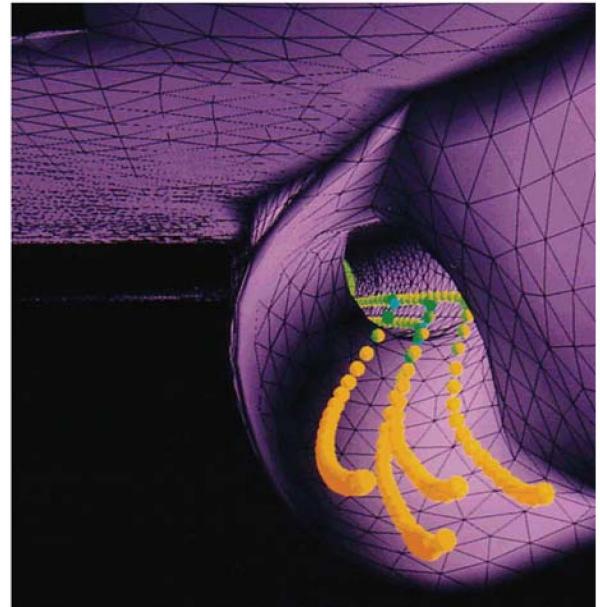


3.5 Implizite Beschreibungen



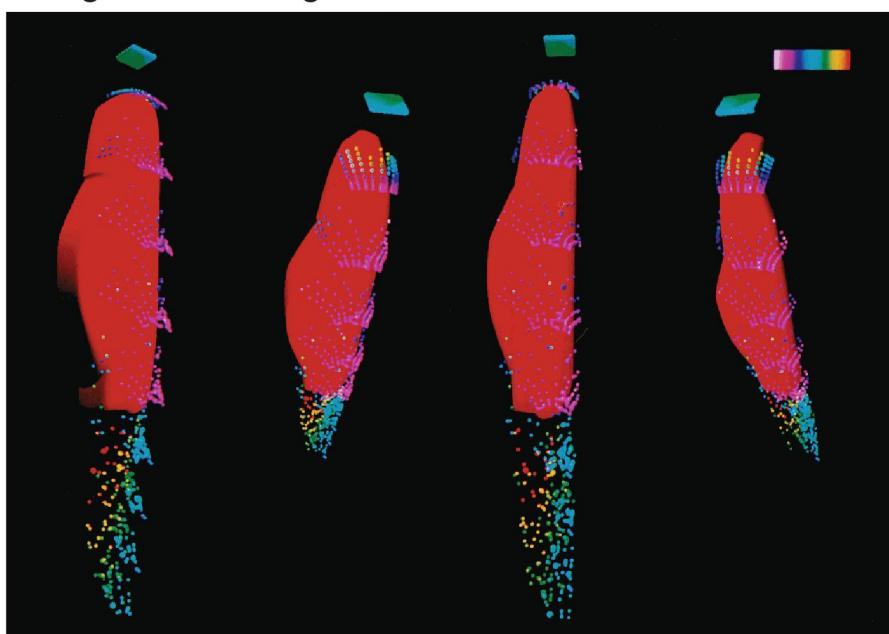
3.5 Implizite Beschreibungen

Anwendungen: Strömungssimulation



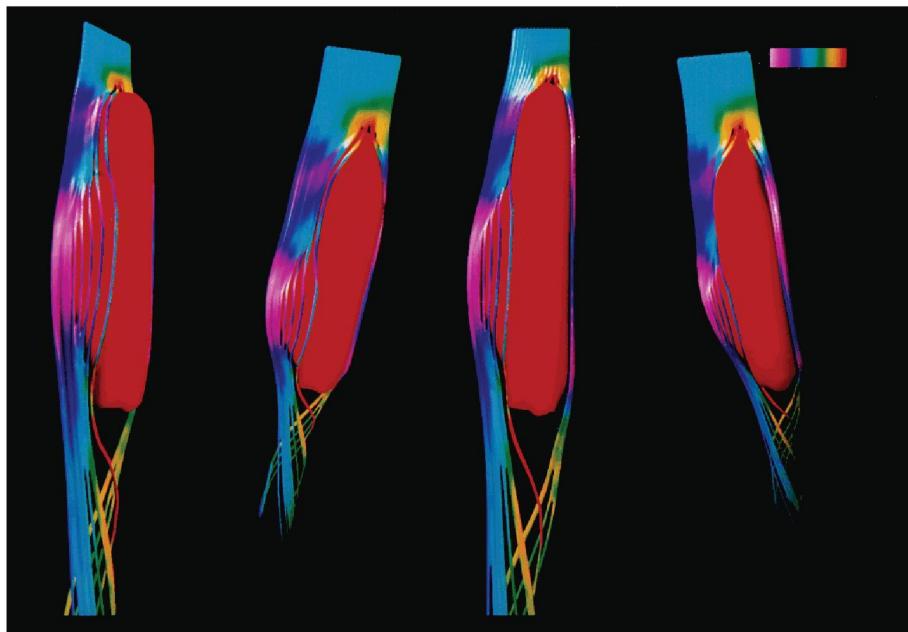
3.5 Implizite Beschreibungen

Anwendungen: Strömungssimulation



3.5 Implizite Beschreibungen

Anwendungen: Strömungssimulation



3.6 Szenenbeschreibung und -management

Mit steigenden Forderungen nach hochqualitativer Realzeit-Computergrafik, insbesondere aus den Anwendungsbereichen Spieleindustrie und Virtual Reality, wird die *effiziente Beschreibung, die Organisation und das Management der Szene* immer bedeutsamer.

Zugleich verschiebt sich das eigentliche Rendering einzelner Objekte immer mehr in Richtung Hardware (heutige Grafikkarten!).

Während die Zusammenfassung und Organisation einzelner Objekte zu/in einer Szene über hierarchische Baumstrukturen - den **Szenengraphen** - organisiert wird, tragen vor allem viele aktuelle Entwicklungen im **LOD**-Bereich heutigen komplexen Szenen Rechnung.



3.6 Szenenbeschreibung und -management

Szenengraph

- hierarchische Baumstruktur zur Speicherung einer Szene mit ihren Modellen
- praktisch alle heutigen Grafiksprachen bieten eine Organisation über eine Szenengraph-Struktur an

im Einzelnen kann der Szenengraph enthalten:

- shapes (Formen, Geometrie, Erscheinungsbild)
- Gruppierungen
- Transformationen
- Lichtquellen, Hintergrund, Nebel,...
- Sichtdefinition
- Verhalten (z. B. bei Animationen)
- anwendungsspezifische Attribute, Sound, ...



3.6 Szenenbeschreibung und -management

Szenengraph (cont.)

- Die eben betrachtete Datenstruktur für Szenengraphen nennen wir **gerichteten azyklischen Graph** oder **directed acyclic graph** – kurz **DAG**.
- DAGs sind der momentane Standard für die Szenen- bzw. Weltbeschreibungen.

Wir betrachten nun eine Szenenbeschreibung am Beispiel VRML:



3.6 Szenenbeschreibung und -management

Szenengraph (cont.)

```
#VRML V2.0 utf8
# our first VRML example

Shape {
    appearance Appearance {
        material Material { }
    }
    geometry Cylinder { }
```



3.6 Szenenbeschreibung und -management

Szenengraph (cont.)

```
#VRML V2.0 utf8
# our first VRML example, bigger
```

```
Shape {
    appearance Appearance {
        material Material { }
    }
    geometry Cylinder {
        radius 3
        height 6
        side TRUE
        top TRUE
        bottom TRUE
    }
}
```

nodes beginnen mit Großbuchstaben

↑
fields beginnen mit Kleinbuchstaben

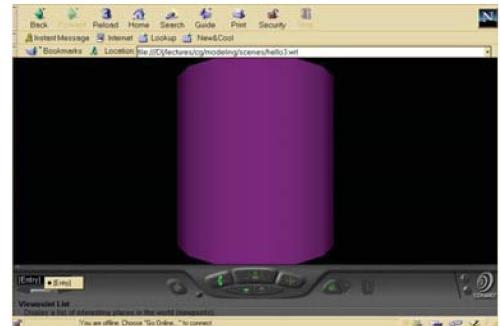


3.6 Szenenbeschreibung und -management

Szenengraph (cont.)

```
#VRML V2.0 utf8
# our first VRML example, bigger, coloured

Shape {
    appearance Appearance {
        material Material {
            diffuseColor      0.5 0.0 0.5
            shininess       0.5
        }
    }
    geometry Cylinder {
        radius      3
        height      6
        side        TRUE
        top         TRUE
        bottom      TRUE
    }
}
```



3.6 Szenenbeschreibung und -management

LOD (level of detail)

Motivation: Allgemein tendieren Verfahren zur Erzeugung polygonaler Modelle dazu, „zu viele“ Polygone zu produzieren.

Problem:

In den überwiegenden Fällen ist das Verhältnis (Polygonanzahl des Objektes) / (projizierte Fläche des Objekts) viel zu groß.

Diesen Overhead bei der Speicherung, Übertragung, Bearbeitung und Visualisierung „unnötiger“ Polygone begegnet man mit verschiedenen „polygonalen Auflösungen“ der Objektrepräsentation - level of detail (LOD).

Diese werden als sogenannte „detail pyramid“ verwaltet.



3.6 Szenenbeschreibung und -management

LOD (cont.)

Folgende Aspekte stehen im aktuellen Blickfeld der Forschung und Entwicklung:

- mesh simplification
Reduzierung der Polygone auf eine Ebene, die gerade für die aktuelle Qualitätsanforderung ausreicht
- level of detail approximation
Vermeidung von „popping“, d. h. visuellen Sprüngen beim Umschalten zwischen verschiedenen Detaillierungsgraden
→ geomorphs (sanfte visuelle Übergänge)
- progressive transmission
3D-Äquivalent zur progressiven Übertragung verschiedener Detaillierungsgrade bei 2D-Bitmap-Bildern



3.6 Szenenbeschreibung und -management

LOD (cont.)

Forschungs- und Entwicklungsaspekte: (Fortsetzung)

- mesh compression
Minimierung des Speicherplatzes der detail pyramid mittels Kompressionsverfahren
- selective refinement
dynamische kontextabhängige LOD-Technik
z. B. Pilot fliegt mit Flugzeug über eine Landschaft, diese muss nur dort volldetailliert dargestellt werden, wo sich das Flugzeug befindet / wo der Pilot hinschaut

