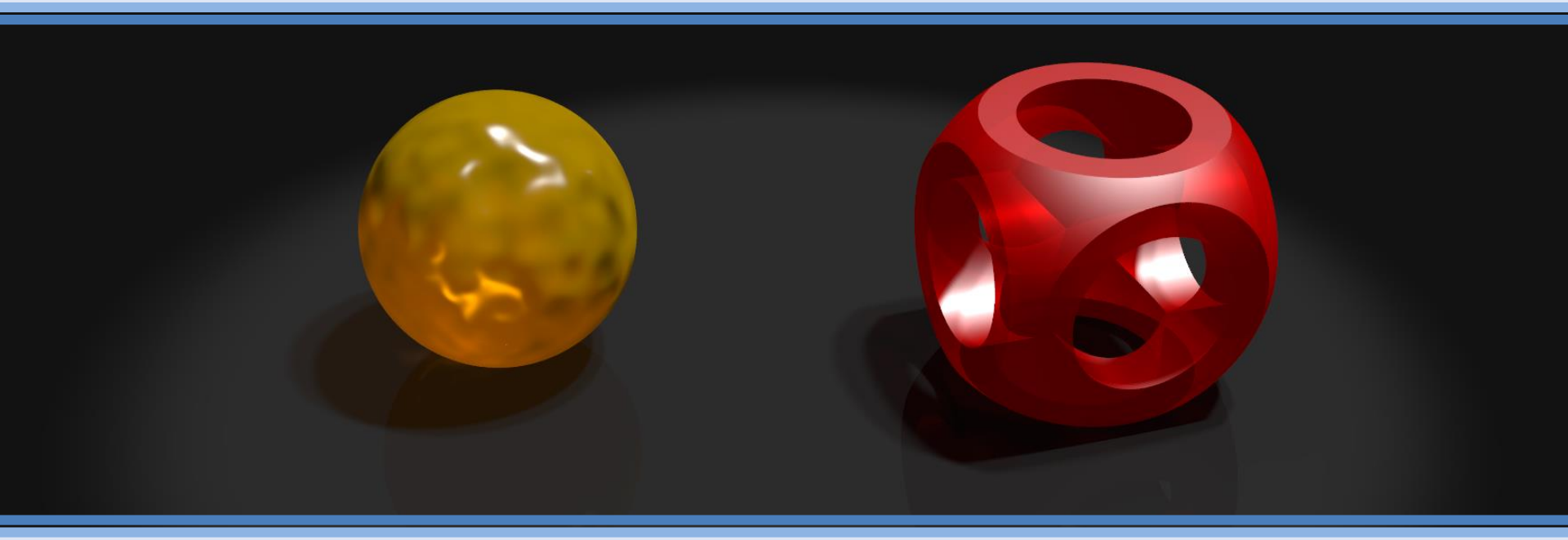
Computergrafik

T. Hopp



Themenübersicht

1. Einführung
2. Programmierbibliotheken / OpenGL
3. Geometrische Repräsentation von Objekten
4. Koordinatensysteme und Transformationen
5. Zeichenalgorithmen
6. Buffer-Konzepte
7. Farbe, Beleuchtung und Schattierung
8. Texturen
9. **Animationen**
10. Raytracing
11. Volumenvisualisierung

Animation

- Jegliche Veränderung einer Szene mit der Zeit
- Häufigste Animationen
  - Bewegung des Augpunktes
  - ***Bewegung von Objekten***
  - Bewegung von Lichtquellen
  - ***Veränderung der Gestalt eines Objektes*** (Form, Farbe, Textur, …)

- Unterscheidung von Hierarchieebenen für Animationen:
  - **Pfadanimation**: Bewegung starrer Objekte entlang einer räumlichen Bahn
  - **Artikulation**: Bewegung innerer Freiheitsgrade eines Objektes (z.B. Gelenk)
  - **Morphing**: elastische oder plastische Verformung eines Objektes
  - **Partikelsysteme**: gleichartige Bewegung einer Objektgruppe

9.1. PFADANIMATION

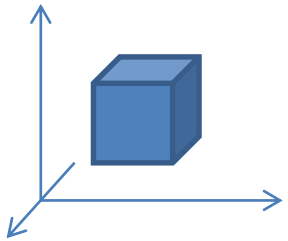
Pfadanimation

- Bewegung eines starren Objektes folgt einer Bahnkurve $\mathbf{K}$ im dreidimensionalen euklidischen Raum
- Darstellung über parametrische Funktion:

$$\mathbf{K}(t) = \begin{pmatrix} x(t) \\ y(t) \\ z(t) \end{pmatrix}$$

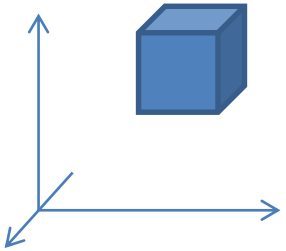
- Beispiel: geradlinige Bewegung entlang der x-Achse mit konstanter Geschwindigkeit v:

$$\mathbf{K}(t) = \begin{pmatrix} x(t) = v \cdot t \\ y(t) = 0 \\ z(t) = 0 \end{pmatrix}$$



- Beispiel: Kreisbewegung in der x-y-Ebene mit Radius R und konstanter Winkelgeschwindigkeit w

$$\mathbf{K}(t) = \begin{pmatrix} x(t) = R \cdot \cos(w \cdot t) \\ y(t) = R \cdot \sin(w \cdot t) \\ z(t) = 0 \end{pmatrix}$$



Pfadanimation

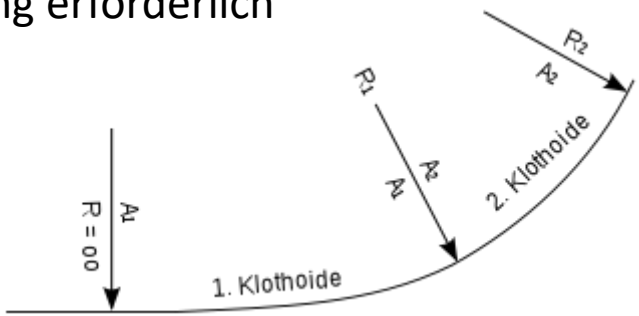
- Kinematische Betrachtung, keine Abbildung der physikalischen Realität
  - Z.B. Übergang von gerader Bewegung in Kurve durch langsam ansteigende Krümmung

- Mathematische Formulierung mit langsam ansteigender Krümmung: *Klothoide*

$$\mathbf{K}(t) = \begin{pmatrix} x(t) = a\sqrt{\pi}\int_0^t \cos\left(\frac{\pi u^2}{2}\right) du \\ y(t) = a\sqrt{\pi}\int_0^t \sin\left(\frac{\pi u^2}{2}\right) du \\ z(t) = 0 \end{pmatrix}$$

Laufvariable t in der Obergrenze des Integrals
→ Numerische Berechnung erforderlich

- Aufwändige Berechnung des Integrals!



Pfadanimation

- Bei komplexen Raumkurven meist Vorberechnung an diskreten Stellen
  - Interpolation (Lineare Interpolation, Splines) zwischen berechneten Stützstellen.

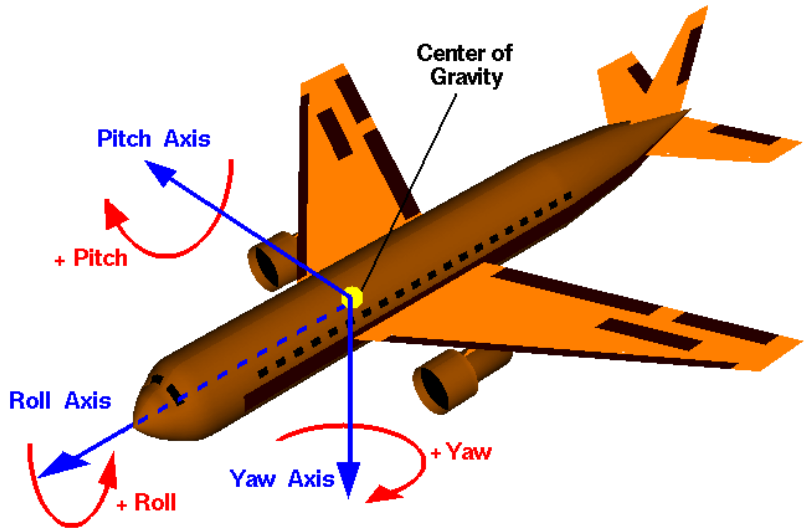
- Allgemein lässt sich dies in die *Key Frame* Technik überführen: Definition einer Bewegung durch n diskrete Abtastpunkte:

$$\mathbf{K} = \begin{pmatrix} x = (x_0, x_1, \dots, x_n) \\ y = (y_0, y_1, \dots, y_n) \\ z = (z_0, z_1, \dots, z_n) \end{pmatrix}$$

- Abspeicherung der Position der animierten Objekte zu diskreten Zeitpunkten.
- Zwischenwerte werden durch Interpolation gewonnen.

Pfadanimation

- Bisher nur Betrachtung von Translation
- Darüber hinaus üblicherweise Drehwinkel des Objekts um die Koordinatenachsen

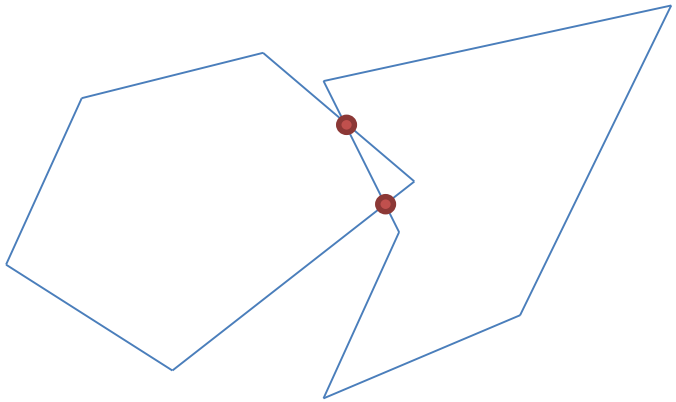


$$\mathbf{K} = \begin{pmatrix} x = (x_0, x_1, \dots, x_n) \\ y = (y_0, y_1, \dots, y_n) \\ z = (z_0, z_1, \dots, z_n) \\ h = (h_0, h_1, \dots, h_n) \\ p = (p_0, p_1, \dots, p_n) \\ r = (r_0, r_1, \dots, r_n) \end{pmatrix}$$

https://de.wikipedia.org/wiki/Roll-Nick-Gier-Winkel

Kollisionserkennung

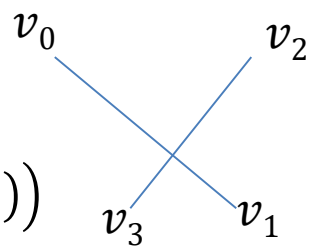
- Komplexität der Berechnung abhängig von Objektart und Komplexität der Szene bzw. Menge der Objekte
  - Analytische Lösungen für einfache Objekte, z.B. Schnittpunkte zwischen Geraden, zwischen Gerade und Kreis etc.
- Naiver Algorithmus zur Detektion einer Kollision zwischen zwei Polygonen:



  - Test auf Schnittpunkte aller Kanten miteinander

Kollisionserkennung

- Berechnung des Schnittpunkts zweier Strecken
  - Parametrische Darstellung der Strecken:



$$g_1 = v_0 + s(v_1 - v_0)$$
$$\Rightarrow (x(s), y(s)) = \left(v_{0,x} + s(v_{1,x} - v_{0,x}),\ v_{0,y} + s(v_{1,y} - v_{0,y})\right)$$

$$g_2 = v_2 + t(v_3 - v_2)$$
$$\Rightarrow (x(t), y(t)) = \left(v_{2,x} + t(v_{3,x} - v_{2,x}),\ v_{2,y} + t(v_{3,y} - v_{2,y})\right)$$

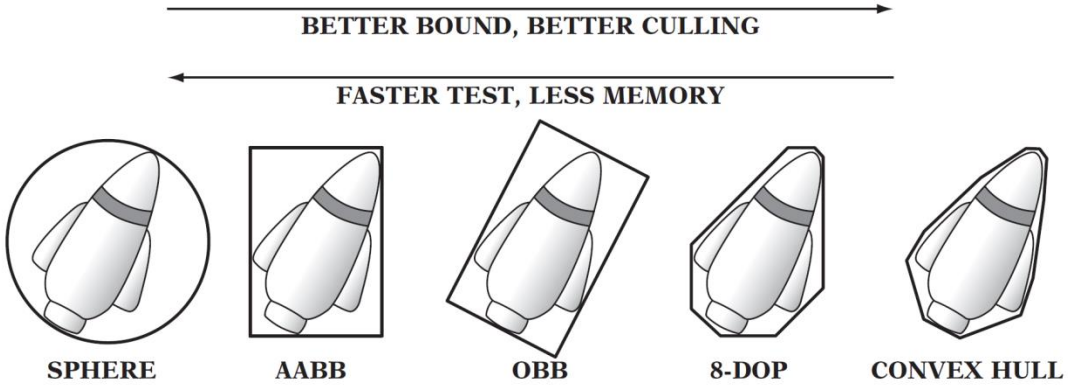
  - Schneiden sich die Strecken, so muss der Schnittpunkt (x_0, y_0) der zugehörigen Geraden Parameter s_0 und t_0 haben mit der Eigenschaft $0 \leq s_0, t_0 \leq 1$
  - Die Schnittparameter s_0 und t_0 sind Lösung des LGS:

$$s(v_{1,x} - v_{0,x}) - t(v_{3,x} - v_{2,x}) = v_{2,x} - v_{0,x}$$
$$s(v_{1,y} - v_{0,y}) - t(v_{3,y} - v_{2,y}) = v_{2,y} - v_{0,y}$$

  - Die Lösung erfolgt z.B. über die Cramer‘sche Regel. Man erhält s_0, t_0.
  - Anschließend wird getestet ob die Bedingung $0 \leq s_0, t_0 \leq 1$ erfüllt ist.
  - Gegenebenfalls wird der Schnittpunkt durch Einsetzen in eine Geradengleichung berechnet.

Kollisionserkennung

- Kollisionserkennung bei komplexeren Objekten durch Hüllkörper: Vereinfachung des komplexen Objekts

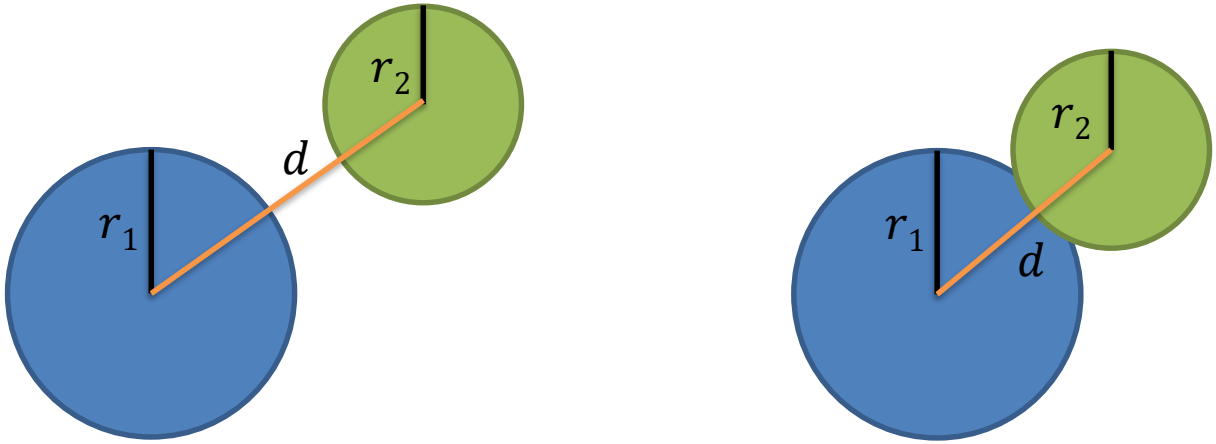


- Approximation durch
  - Kugel
  - Achsen-Parallele Bounding Box (AABB)
  - Am Objekt ausgerichtete Bounding Box (OBB)
  - Polytope, auch k-DOP (Discrete Oriented Polytopes)
  - Konvexe Hülle

C. Ericson: „Real Time Collision Detection“, Elsevier, 2005

Kollisionserkennung

- Beispiel: Kugel-Hüllkörper



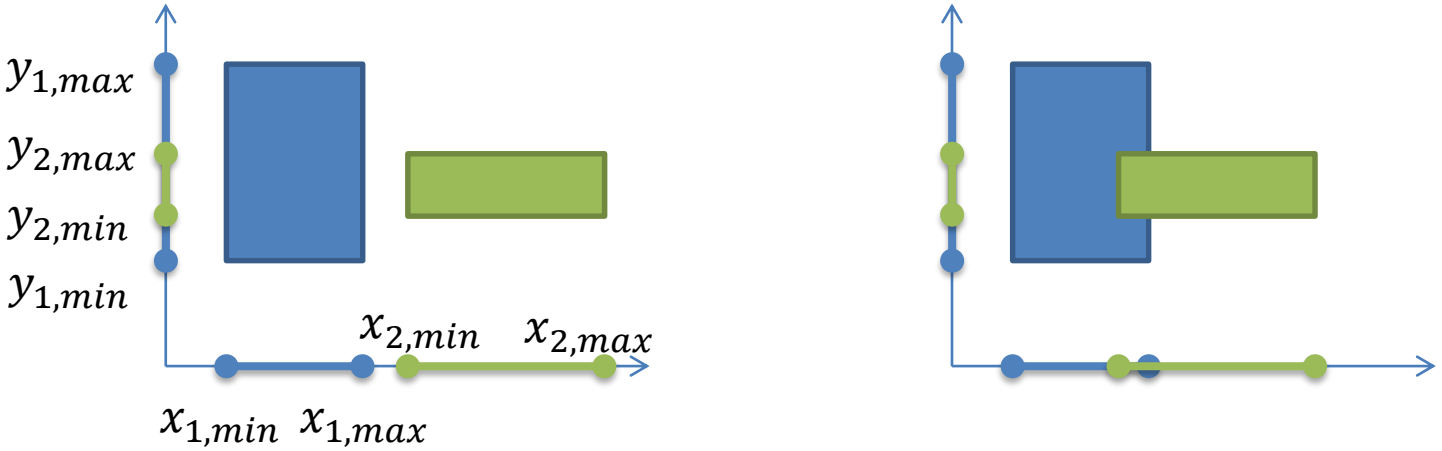
$d = r_1 + r_2$: Kugeln berühren sich in einem Punkt (= Kollision)

$d < r_1 + r_2$: Kugeln schneiden sich (= Kollision)

$d > r_1 + r_2$: Kugeln berühren und schneiden sich nicht (= keine Kollision)

Kollisionserkennung

- Beispiel: AABB Hüllkörper



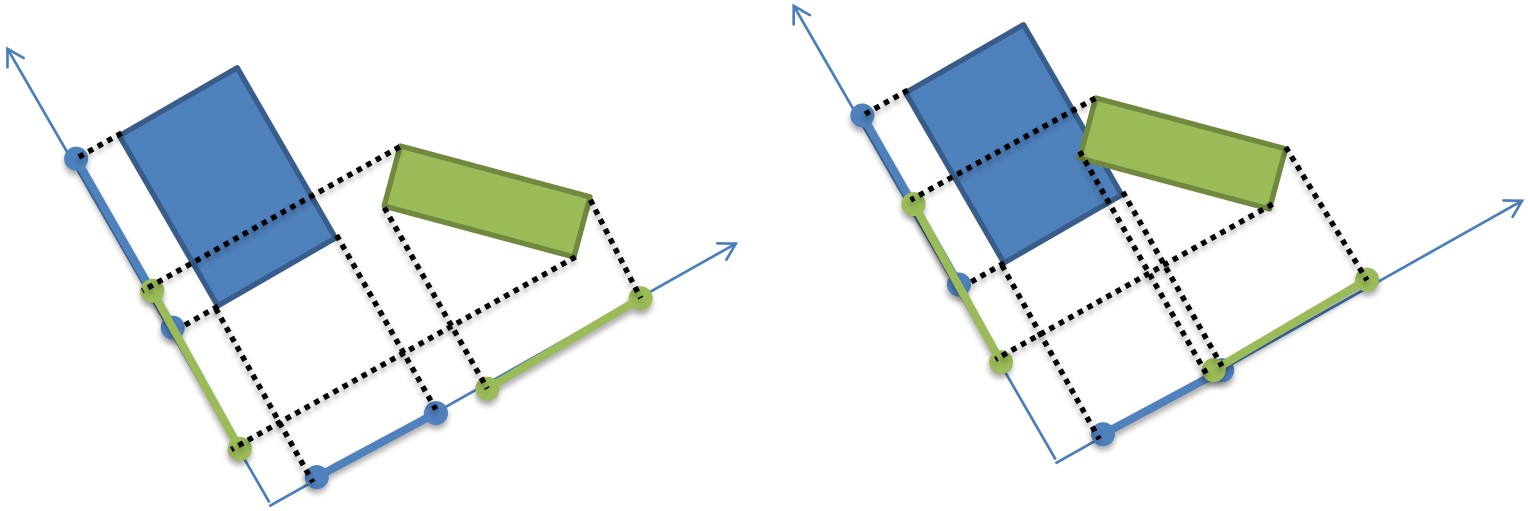
$x_{1,max} < x_{2,min} \vee y_{1,max} < y_{2,min}$: keine Kollision

$x_{1,max} \geq x_{2,min} \wedge y_{1,max} \geq y_{2,min}$: Kollision

Entsprechender Test für $x_{2,max}$und $x_{1,min}$ bzw. $y_{1,min}$ und $y_{2,max}$

Kollisionserkennung

- Beispiel: OBB Hüllkörper



  - Projektion der Körper auf Achsen, die Parallel zu einer Seite des OBB Hüllkörpers verlaufen
  - Testbedingungen analog zum AABB Hüllkörper

Kollisionserkennung

- Bei großer Anzahl von Objekten sind sehr viele Tests nötig: Bei n Objekten $\frac{n \cdot (n-1)}{2}$ Tests
  - 10 Objekte: 45 gegenseitige Tests
  - 100 Objekte: 4950 gegenseitige Tests!

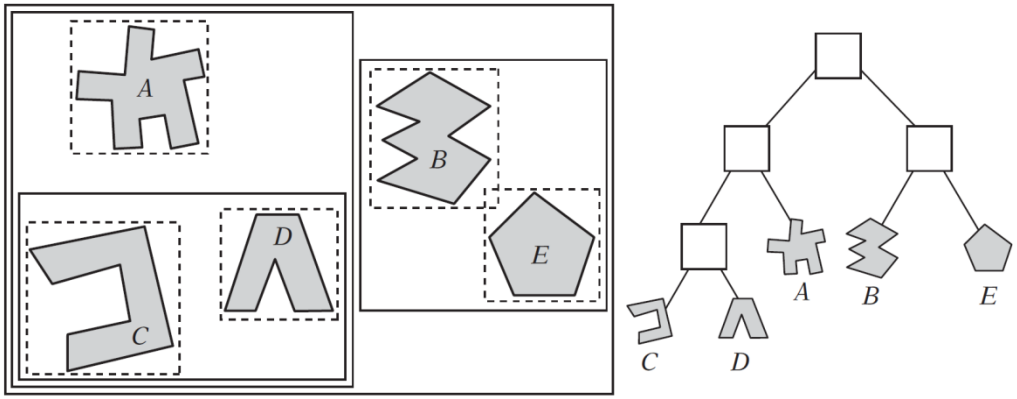
- Abhilfe: Hierarchiebäume und Raumunterteilung

- Hüllkörper meist nur in erster Näherung genutzt.
  - Wenn eine Kollision entdeckt wird, wird der Test verfeinert.

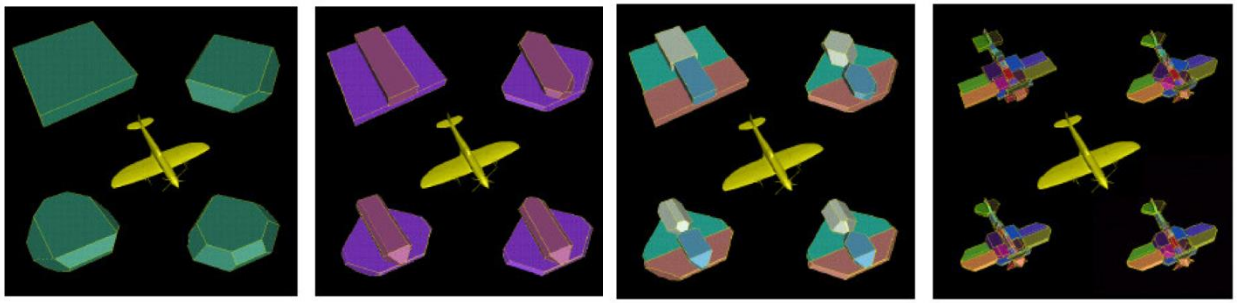
  → Hierarchiebäume

Kollisionserkennung

- Hierarchiebäume
  - Zusammenfassen von mehreren Objekten



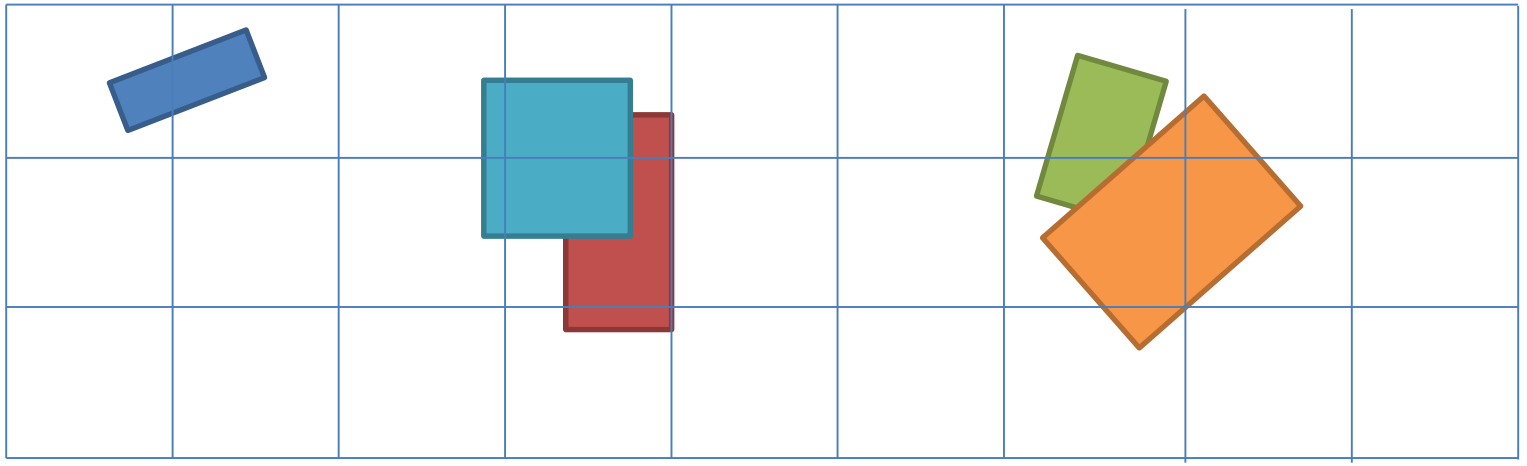
  - Weitere Unterteilung von Hüllkörpern wenn Kollision entdeckt wurde



J.T. Klosowski et al.: „Efficient collision detection using bounding volume hierarchies of k-DOPs", IEEE Trans. on Visualization and Computer Graphics 4(1), 1998

Kollisionserkennung

- Raumunterteilung:
  - Z.B. Uniforme Raumaufteilung



  - Zwei Objekte können sich nur überlappen, wenn sie in einer gemeinsamen Region liegen
  - Die Zahl paarweiser Tests lässt sich damit reduzieren
  - Wahl der Raumaufteilung wichtiger Parameter
  - Alternative Unterteilungsmöglichkeiten: Quad-/Octrees

Anwendungen

- Z.B. Kollissionserkennung in Computerspielen und Simulationen

http://www.flightsimworld.com/gallery/displayimage.php?pid=757
http://www.tagesspiegel.de/mobil/lkw-simulator-von-sifat-road-safety-alarmfahrt-auf-probe/10114080.html

Medizinische Anwendung

- z.B. Bewegung von starren Instrumenten bei OP-Simulation
- Kollisionserkennung wichtig: Force-Feedback!

https://www.oh-tech.org/blog/surgery_simulation_research_garners_london_praise#.VjJF8msmkUF

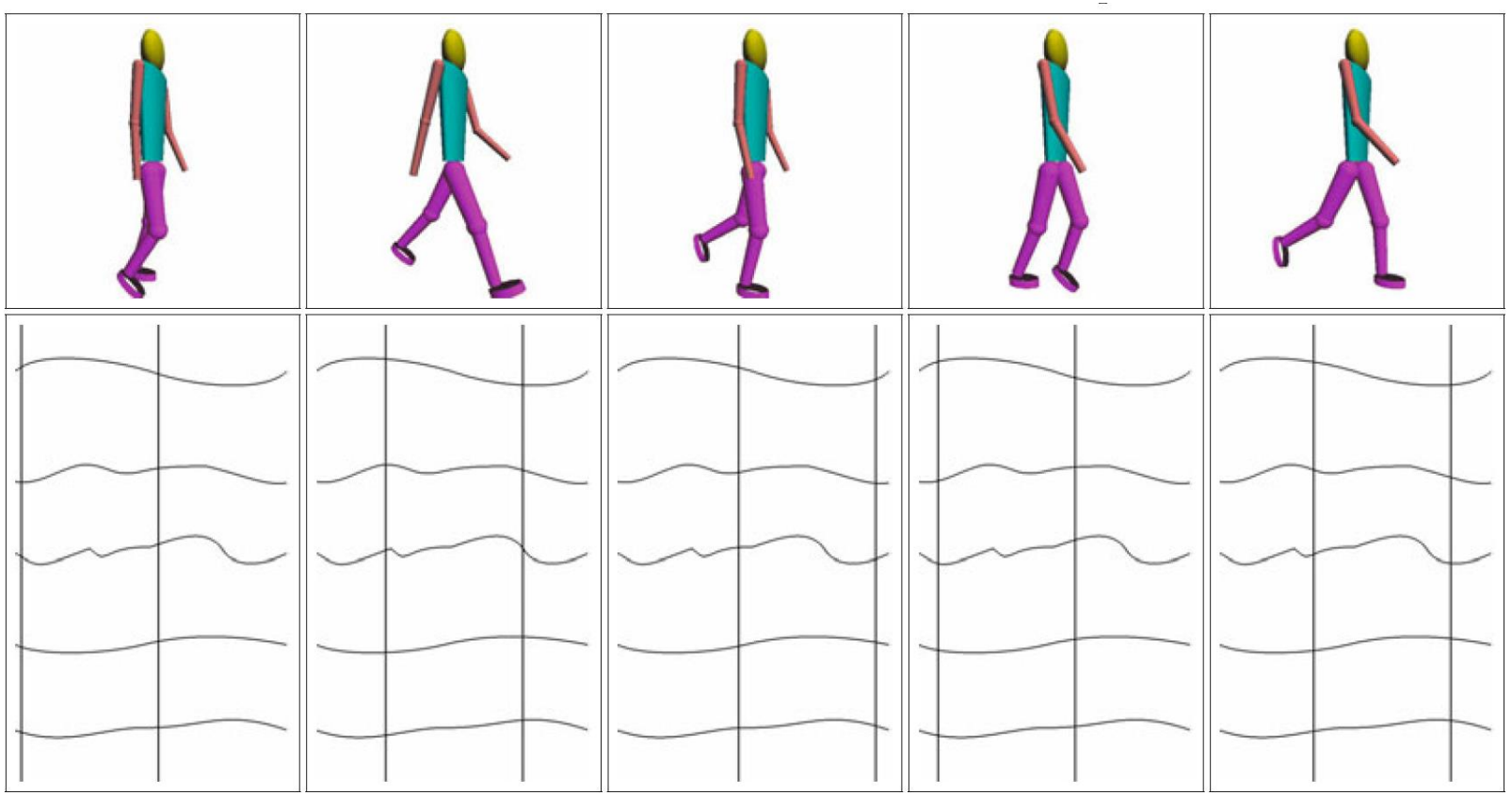
9.2 ARTIKULATION

Artikulation

- Bewegung von Gelenken eines Objekts, Teilobjekte sind starr
- Beschreibung der Gelenkwinkel als Kurve über der Zeit
  (= „Bewegungssatz“)

Artikulation

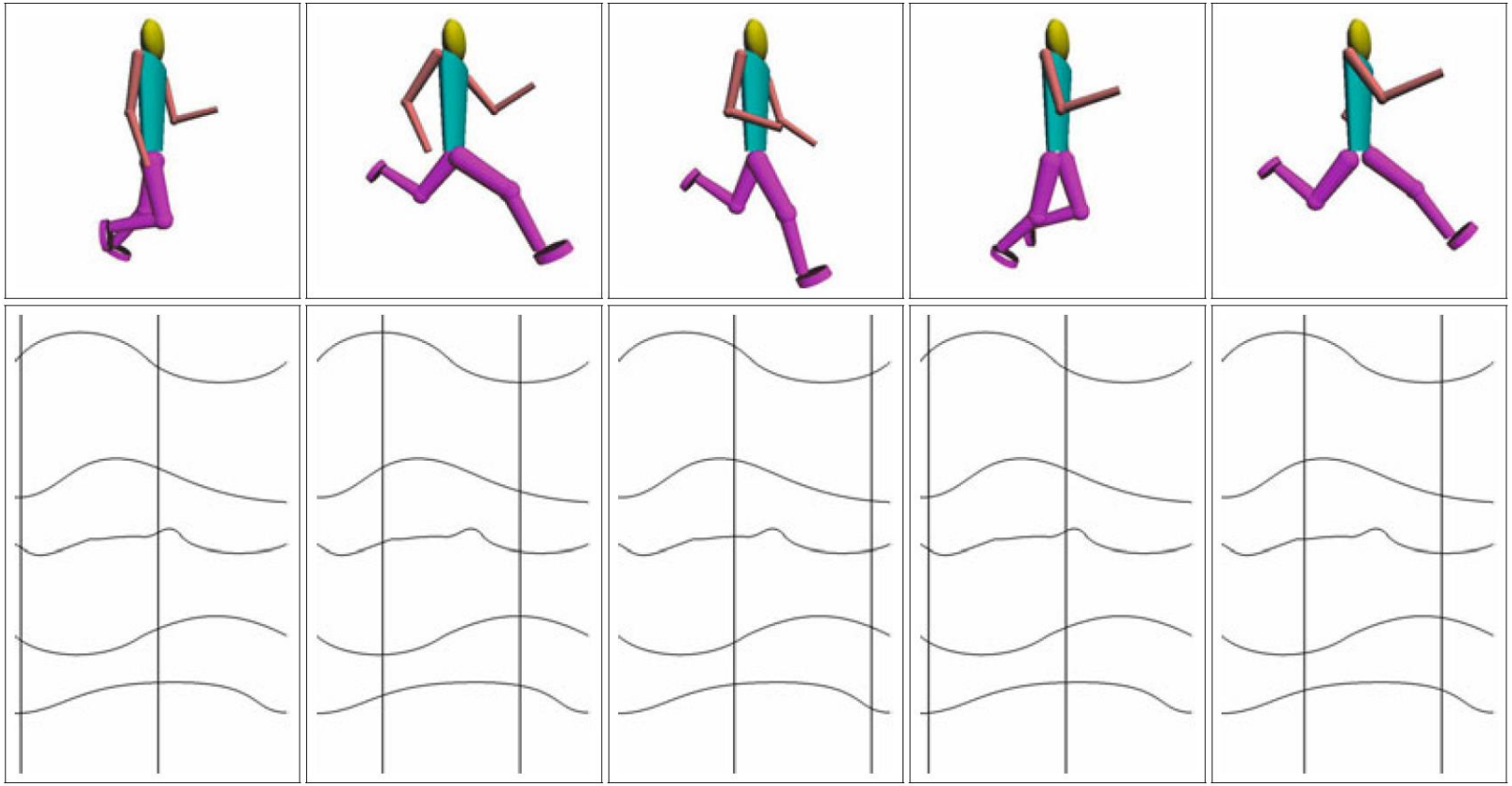
- Beispiel: Laufbewegung (10 Gelenke)



A. Nischwitz et al. „Computergrafik und Bildverarbeitung“, Band 1: Computergrafik“

Artikulation

- Änderung der Bewegung durch Austausch des Bewegungssatzes



A. Nischwitz et al. „Computergrafik und Bildverarbeitung“, Band 1: Computergrafik“

Artikulation

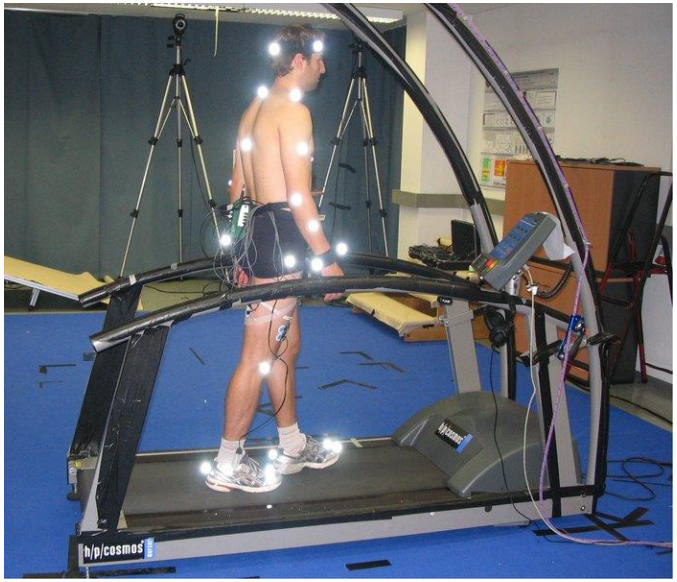
- Statt kontinuierlicher Kurven kann auch Key-Frame-Technik eingesetzt werden:
  - Diskreter Winkel zu einem bestimmten Zeitpunkt
  - Interpolation des Winkels
  - Glattheit der Bewegung bestimmt durch Anzahl der diskreten Winkel und Interpolationsmethode.

- Übergänge zwischen zwei Bewegungsmustern durch Interpolation der Bewegungssätze

- Oft umgekehrtes Problem: Optimale Bewegung für ein Szenario herausfinden
  - Z.B. Berechnen des Bewegungsablaufs zum Ergreifen eines Gegenstandes
  - Hochdimensionales Optimierungsproblem: „Inverse Kinematik“

Artikulation

- Realistischere Personenmodelle für vorgegebene Bewegungen durch Motion Capture
  - Z.B. Bewegungsanalyse → Schätzung der Gelenkwinkel



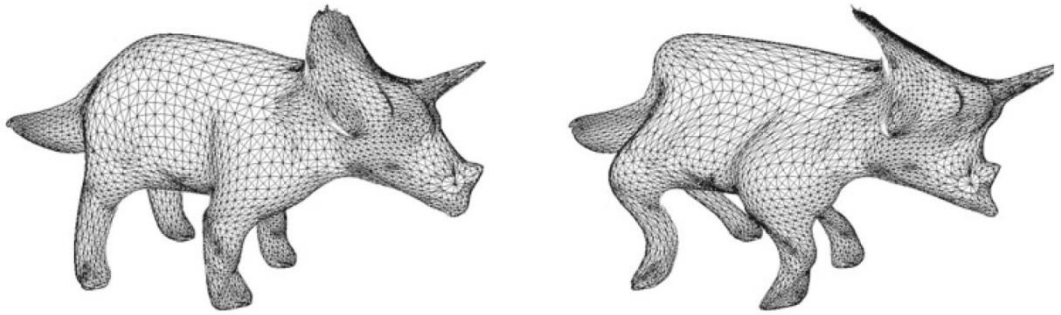
http://www.cin.uni-tuebingen.de/mission-methods/methods/medical-technical-applications.html
https://www.yahoo.com/movies/a-brief-history-of-motion-capture-in-the-movies-91372735717.html

9.3 MORPHING

Morphing

- Morphing = Verformung von Oberflächen
  - Objekt wird nicht mehr als starrer Körper betrachtet
  - Objekt nicht nur an Gelenken beweglich

- Direkte Verschiebung der Vertex-Positionen
  - Rechenintensive Operationen, da meist Glattheitseigenschaften berücksichtigt werden
  - Normalen der Oberflächen müssen in jedem Verformungsschritt neu berechnet werden



A. Nischwitz et al. „Computergrafik und Bildverarbeitung“, Band 1: Computergrafik“

Morphing

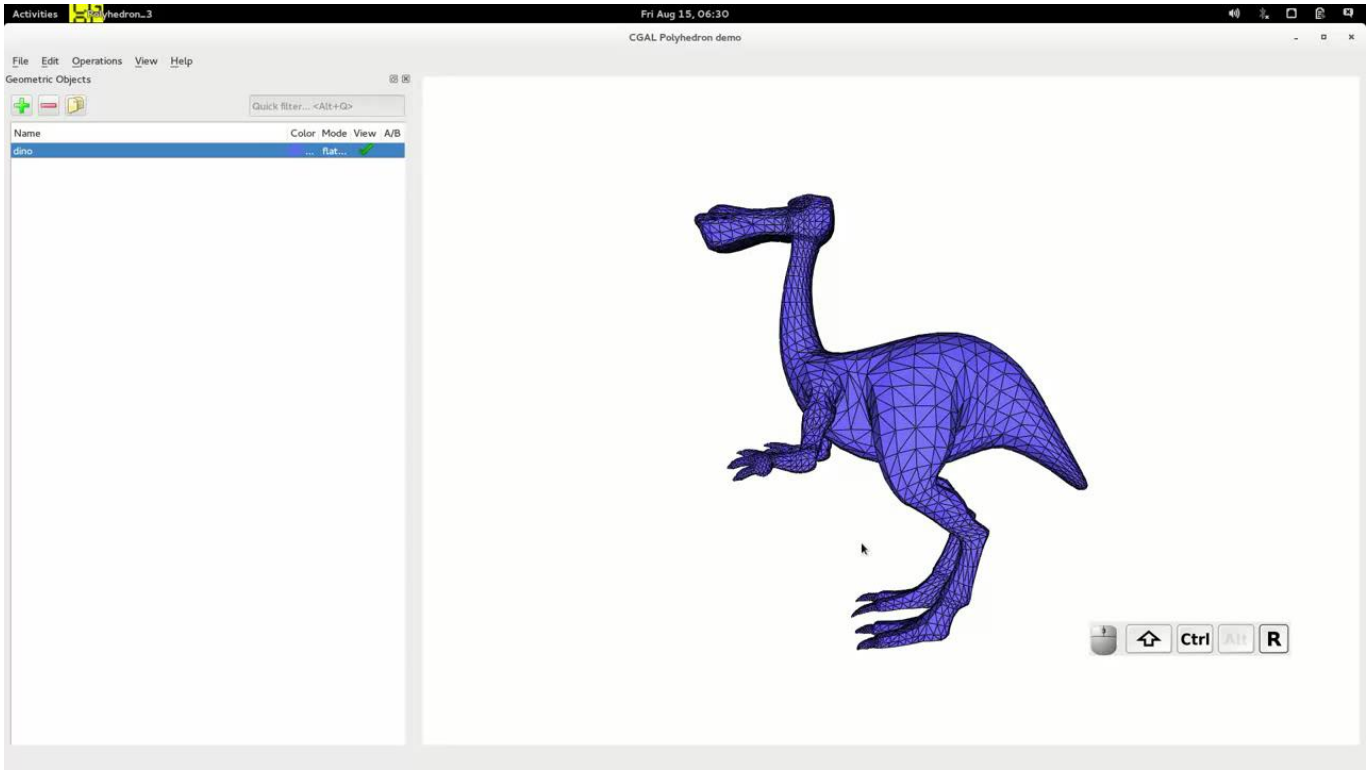
- Ähnlich der Starrkörper-Bewegung, hier aber Definition einer Bewegungsbahn pro Vertex
  - Key-Frame Technik kann eingesetzt werden
  - Topologie des Polygonnetzes bleibt i.d.R. gleich, nur die Vertex-Positionen werden verschoben.

- Bewegungsbahn durch explizite Definition, physikalische Berechnung etc.

- Kollisionserkennung
  - Kollisionen zwischen mehreren Objekten: in jedem Morphing-Schritt bestimmen
  - Eigenkollision möglich!

Morphing

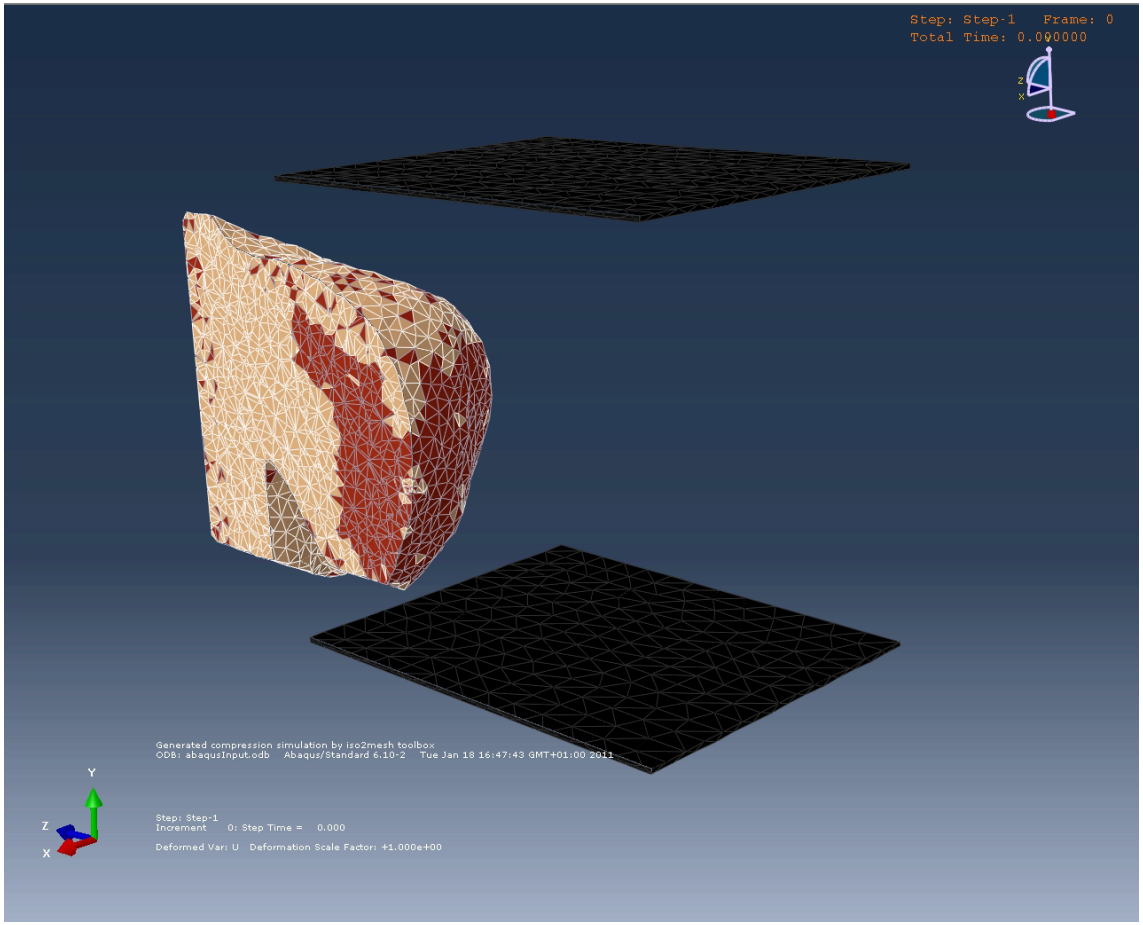
- Beispiel: Methoden von CGAL



http://doc.cgal.org/latest/Surface_modeling/index.html

Morphing

- Beispiel aus der Medizin: physikalische Simulation



Simulation und Visualisierung mit ABAQUS

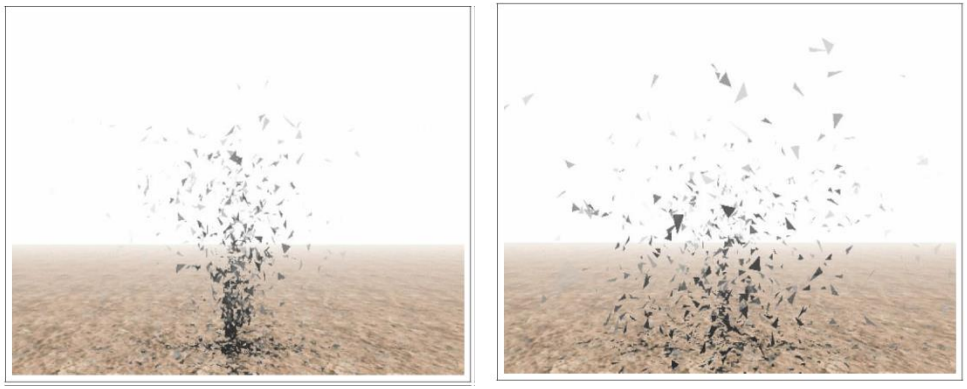
T.Hopp: „Multimodal Registration of X-Ray Mammograms with 3D Volume Datasets", PhD Thesis, University of Mannheim, 2012

9.4 SCHWÄRME UND PARTIKELSYSTEME

Partikelsysteme

- Ansammlung von einfachen Grafikobjekten (Partikel) bildet zusammen ein großes Objekt
- Verwendet zur Darstellung von natürlichen Phänomenen
  - … deren Form nicht eindeutig definierbar ist
  - … die keine eindeutige glatte Oberfläche besitzen

- Beispiele: Feuer, Wasser, Wolken, Bäume, Gräser, Dampf/Rauch, Textilien

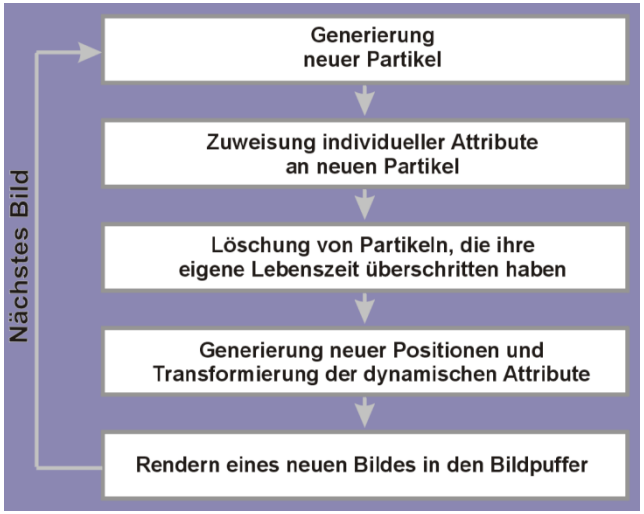


A. Nischwitz et al. „Computergrafik und Bildverarbeitung“, Band 1: Computergrafik“

Partikelsysteme

- Objekt wird durch Häufung von Partikeln beschrieben
- Partikelsystem beschrieben durch zwei Komponenten:
  - Partikel = Elemente des Systems
  - Emitter = Kontrolle über Partikel

- Attribute der Partikel ändern sich über die Zeit, z.B.
  - ***Position***: Punkt im Raum, an dem sich der Partikel zu einem Zeitpunkt befindet
  - ***Geschwindigkeit*** und ***Richtung***
  - ***Größe***
  - ***Farbe***
  - ***Transparenz***
  - ***Form/Aussehen***
  - ***Lebensdauer***



I. Sejdic, D. Erdmann, D. Wilhelm, C. Kurz: „Partikelsysteme & Schwärme“, Seminararbeit, Universität Kassel

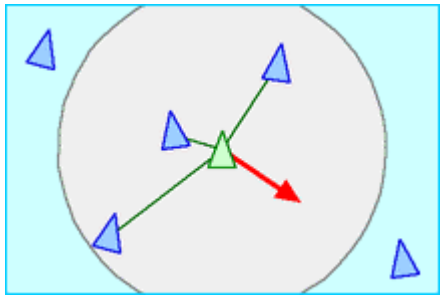
Partikelsysteme

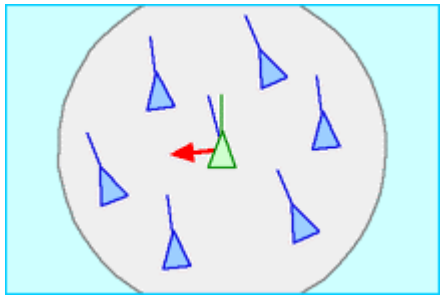
- Stochastische Partikelsysteme
  - Verhalten der Partikel unabhängig. (Beispiel: Feuer)
- Strukturierte Partikelsysteme
  - Verhalten der Partikel in Abhängigkeit von z.B. Nachbarn. (Beispiel: Grasanimation)

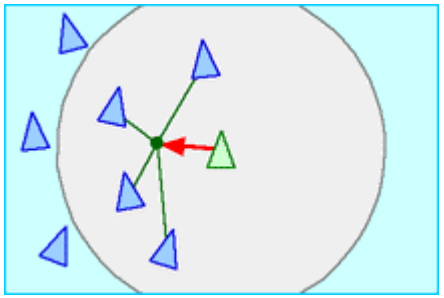
https://www.youtube.com/watch?v=pd0jSmExgB0

Schwärme

- Partikelsystemen sehr ähnlich
- Dennoch kleine Unterschiede:
  - Schwarmelemente bestehen so lange wie der Schwarm selbst
  - Dynamischer Eindruck (z.B. Vogelschwarm-Animation)
  - Schwarmelemente komplexer als Partikel
  - Schwarmelemente greifen aktiv in Geschehen ein. Komplexe Abhängigkeiten können entstehen:
    - Berechnung der Kollisionsvermeidung (z.B. durch Simulation von Kraftfeldern), Variation der Geschwindigkeit
    - Regeln für Schwarmverhalten (z.B. gemeinsame Ausrichtung, Zentrierung)







http://www.red3d.com/cwr/boids/

ZUSAMMENFASSUNG

Zusammenfassung

- Animation = jegliche Veränderung einer Szene über die Zeit
  - **Pfadanimation**: Bewegung starrer Objekte entlang einer räumlichen Bahn
    - Meist Vorberechnung der Bahn an diskreten Stellen + Interpolation (Key Frame Technik)
    - Kollisionserkennung wichtig: einfache Objekte, Hüllkörper, Raumunterteilung, Hierarchieunterteilung
  - **Artikulation**: Bewegung innerer Freiheitsgrade eines Objektes (z.B. Gelenk)
    - Bewegungskurve pro Gelenk: Winkel eines Gelenks
  - **Morphing**: elastische oder plastische Verformung eines Objektes
  - **Partikelsysteme**: gleichartige Bewegung einer Objektgruppe

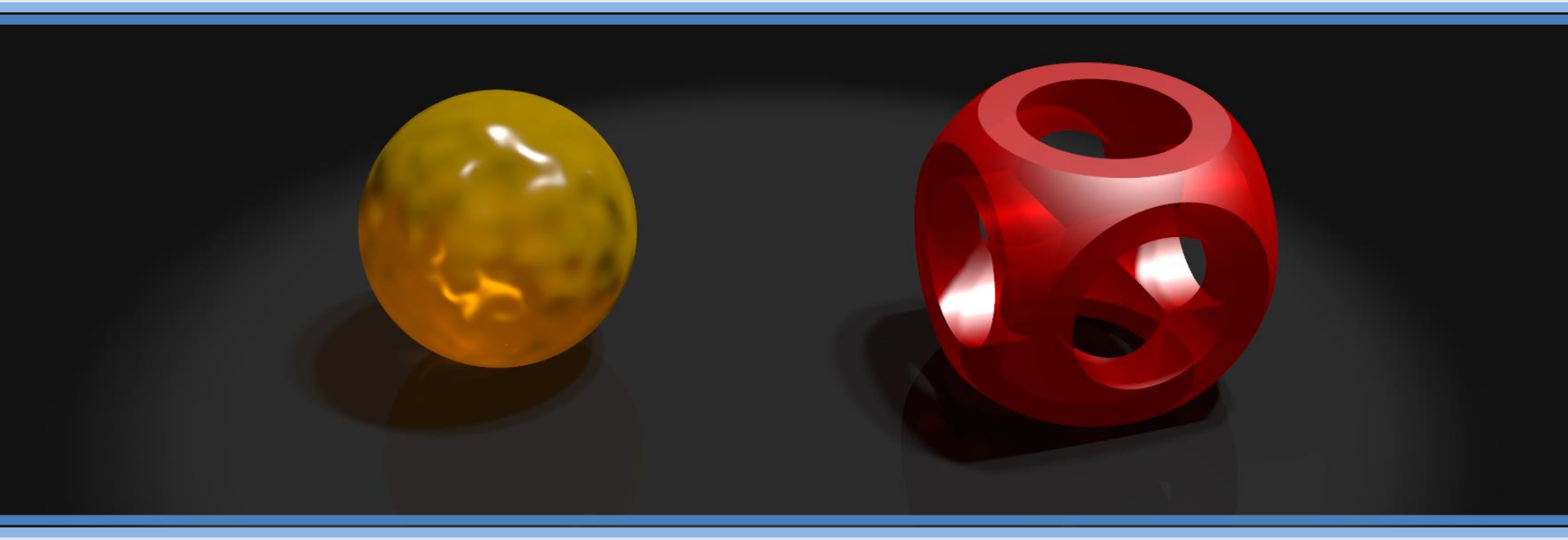
ÜBUNGSAUFGABEN

Übungsfragen Kapitel 9

- Was ist die Key-Frame-Technik?
- Beschreiben Sie die Funktion von Hüllkörpern für die Kollisionserkennung
- Was ist der Unterschied zwischen Partikelsystem und Schwarm?

Computergrafik

T. Hopp



Themenübersicht

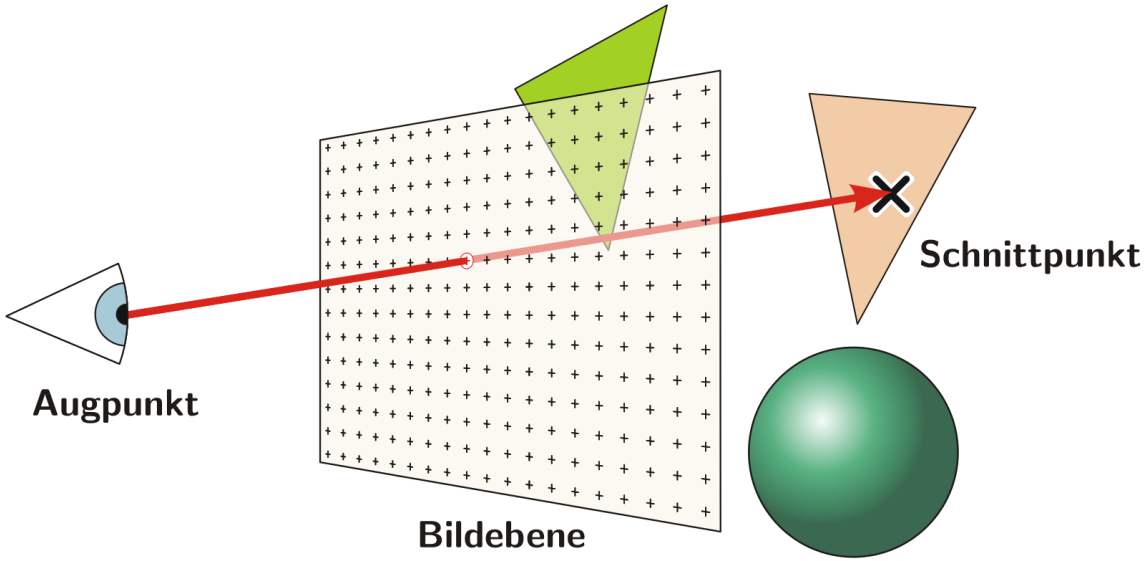
1. Einführung
2. Programmierbibliotheken / OpenGL
3. Geometrische Repräsentation von Objekten
4. Koordinatensysteme und Transformationen
5. Zeichenalgorithmen
6. Buffer-Konzepte
7. Farbe, Beleuchtung und Schattierung
8. Texturen
9. Animationen
10. **Raytracing**
11. Volumenvisualisierung

Raytracing

- Berechnung von (Mehrfach-)Reflexionen und Transmission von Licht → globale Illumination!
- Raytracing arbeitet im Objektraum: Prinzip der Strahlverfolgung in die Szene für jedes Pixel.
- Vorteile: das Raytracing-Modell berechnet durch Strahlverfolgung
  - Verdeckung
  - Transparenz
  - Direkte Beleuchtung
  - Indirekte Beleuchtung
  - Schatten
- Nachteile:
  - Extremer Aufwand!
  - Bei Streulicht an diffusen Oberflächen versagt „klassisches“ Raytracing → z.B. Radiosity
  - Bündeln von Lichtstrahlen (=Kaustik) benötigt ebenfalls Erweiterungen (Photon Mapping, Path Tracing)

Verdeckungsberechnung

- Grundsätzlicher Raytracing-Algorithmus führt eine Verdeckungsberechnung durch:
  - Sichtstrahl vom Augpunkt durch jedes Pixel der *near clipping plane*
  - Berechnung des ersten Schnittpunktes mit einem Objekt. (Schnittpunkt mit geringstem Abstand zum Augpunkt)
  - Farbberechnung durch Beleuchtungsrechnung, Schattierungsalgorithmus



https://de.wikipedia.org/wiki/Raytracing

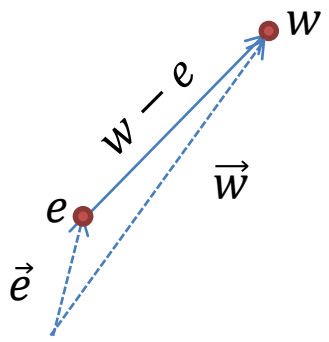
Berechnung von Sichtstrahlen

- Parameterdarstellung einer Geradengleichung vom Augpunkt e zu einem Pixel w in der Bildschirmebene:

$$p(t) = e + t(w - e)$$

- Durch jedes t ist ein Punkt auf der Geraden beschrieben:
  - $t = 0$: Punkt e
  - $t = 1$: Punkt w
  - $t_1 < t_2$: t_1 ist näher am Augpunkt als t_2.

- Bestimmung von w:
  - Augpunkt-Transformation der Szene
  - Diskretisierung der near clipping plane.



Verdeckungsberechnung

- Pseudocode für ein einfaches Ray Tracing, das Ergebnisse ähnlich dem z-Buffer liefert

```
Funktion Bild_Rendern
    Strahl.Ursprung := Augpunkt
    Für jedes (x,y)-Pixel der Rastergrafik
        Strahl.Richtung := [3D-Koordinaten des Pixels der Bildebene] – Augpunkt
        Farbe des (x,y)-Pixels := Farbe_aus_Richtung(Strahl)

Funktion Farbe_aus_Richtung(Strahl)
    Schnittpunkt := Nächster_Schnittpunkt(Strahl)
    Wenn Schnittpunkt.Gewinner = (kein) dann
        Farbe_aus_Richtung := Hintergrundfarbe
    sonst
        Farbe_aus_Richtung := Farbe_am_Schnittpunkt(Strahl, Schnittpunkt)

Funktion Nächster_Schnittpunkt(Strahl)
    MaxDistanz := ∞
    Schnittpunkt.Gewinner := (kein)
    Für jedes Primitiv der Szene
        Schnittpunkt := Teste_Primitiv(Primitiv, Strahl)
        Wenn Schnittpunkt.Distanz < MaxDistanz dann
            MaxDistanz := Schnittpunkt.Distanz
            Schnittpunkt.Gewinner := Primitiv
    Nächster_Schnittpunkt := Schnittpunkt
```

Beschleunigung der Schnittpunktberechnung

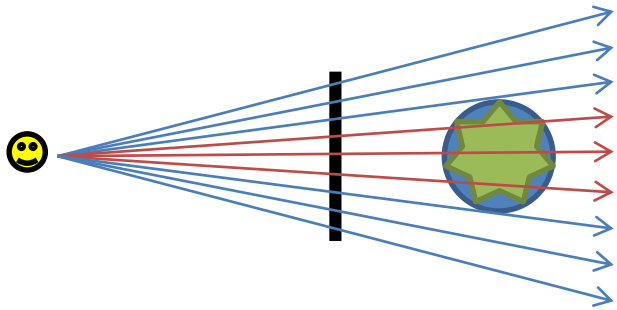
- Implementierung schneidet jeden Sichtstrahl mit jedem Objekt: extrem aufwändig!
  - Auflösung 1024 x 768 bei 100 Objekten: ca. 78 Mio. Schnittpunkberechnungen!
  - Ca. 75-95% der Rechenzeit wird für Schnittpunktberechnung benötigt

- Beschleunigungs-Strategien: sehr ähnlich zu Kollisionsdetektion

  - Z-Sort:
    1. Transformation der Strahlen, so dass sie entlang der z-Achse verlaufen
    2. Transformation der Objekte und Sortierung nach z-Werten → Schnittpunkt mit vorderstem Objekt
    3. Zurücktransformation der Schnittpunkte für Schattierungsberechnung

Beschleunigung der Schnittpunktberechnung

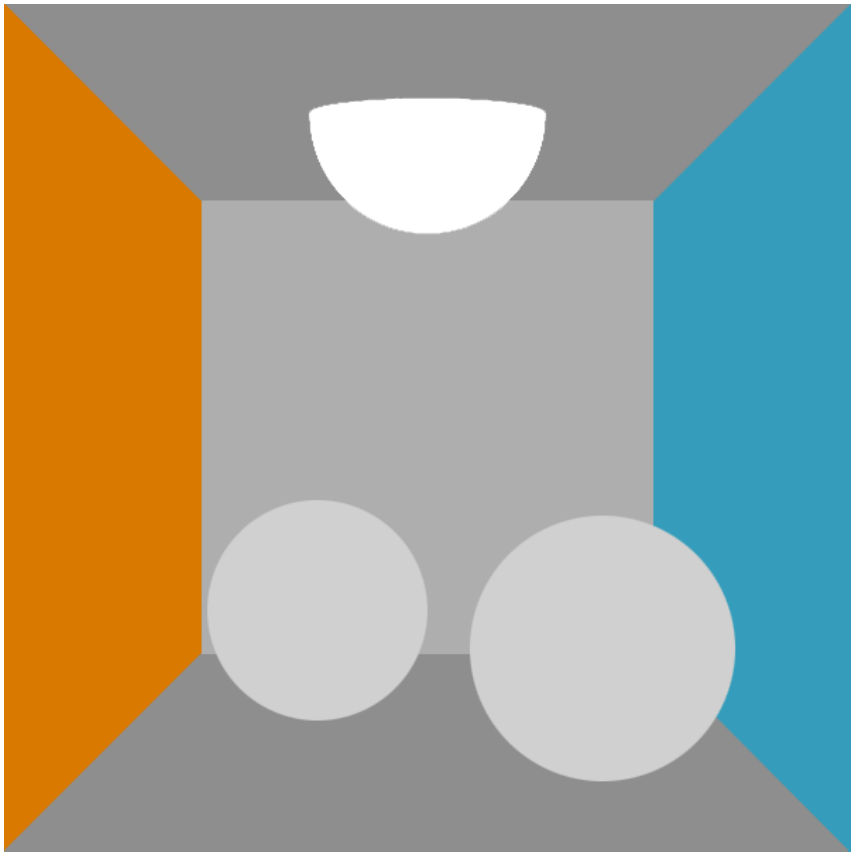
- Begrenzende Volumen:
  - Komplexe Objekte mit aufwändiger Schnittpunktberechnung durch umhüllenden Körper, z.B. Kugel oder Quader umgeben
  - Testen des komplexen Objektes nur wenn der umhüllende Körper vom Sichtstrahl geschnitten wird



- Hierarchien:
  - Organisation von begrenzenden Volumen in verschachtelten Hierarchien
  - Baumartiger Aufbau mit komplexen Objekten als Blätter
  - Test der Objekte eines Baumzweiges kann entfallen wenn Vaterknoten schon nicht geschnitten wird.
- Bereichsunterteilung
  - Unterteilung der gesamten Szene in ein regelmäßiges Gitter
  - Zuweisung der Objekte zu Bereichen
  - Schnittberechnung nur mit Objekten wenn ein Strahl den zugehörigen Bereich durchläuft.

Verdeckungsberechnung

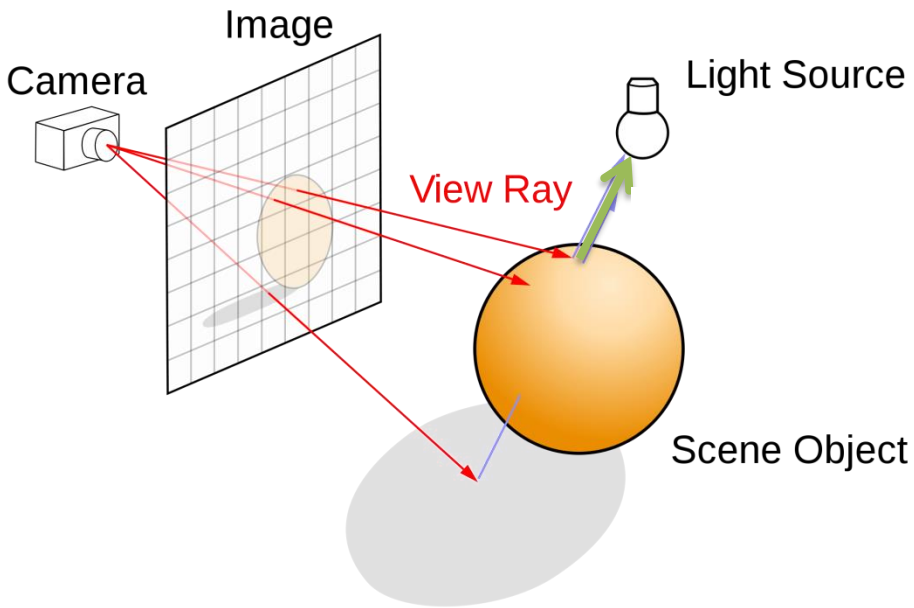
- Bisher:



https://de.wikipedia.org/wiki/Raytracing

Beleuchtung von Objekten

- Die Beleuchtung von Objekten erfolgt analog zu den bisher besprochenen Beleuchtungsmodellen:
  - Winkelberechnung zwischen eintreffendem Sichtstrahl zur Oberflächennormale
  - Winkelberechnung zwischen Oberflächennormale und Gerade von Schnittpunkt zu Lichtquelle
  - Immer noch lokale Beleuchtung!

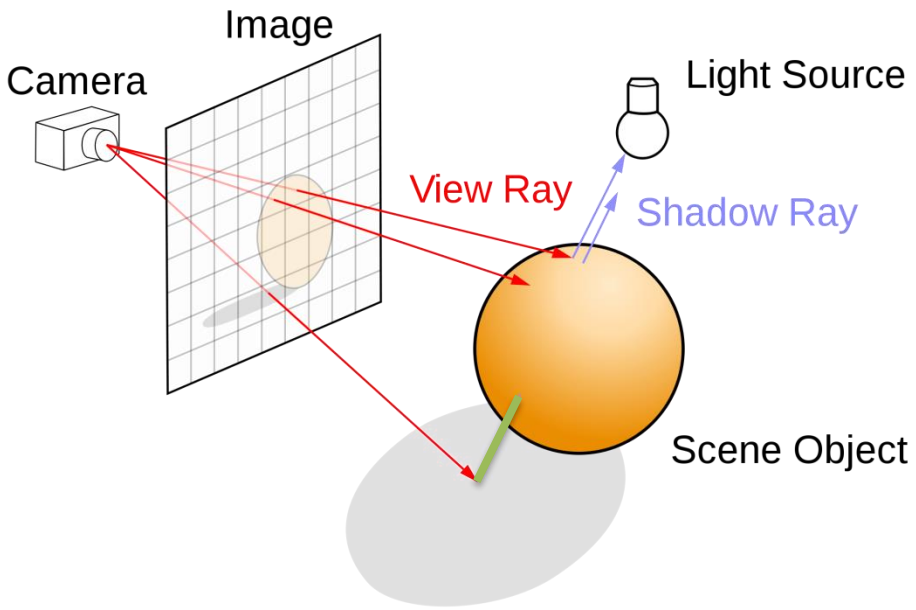


  - Bei mehreren Lichtquellen: Beleuchtungsberechnung für alle Lichtquellen!

https://en.wikipedia.org/wiki/Ray_tracing_%28graphics%29

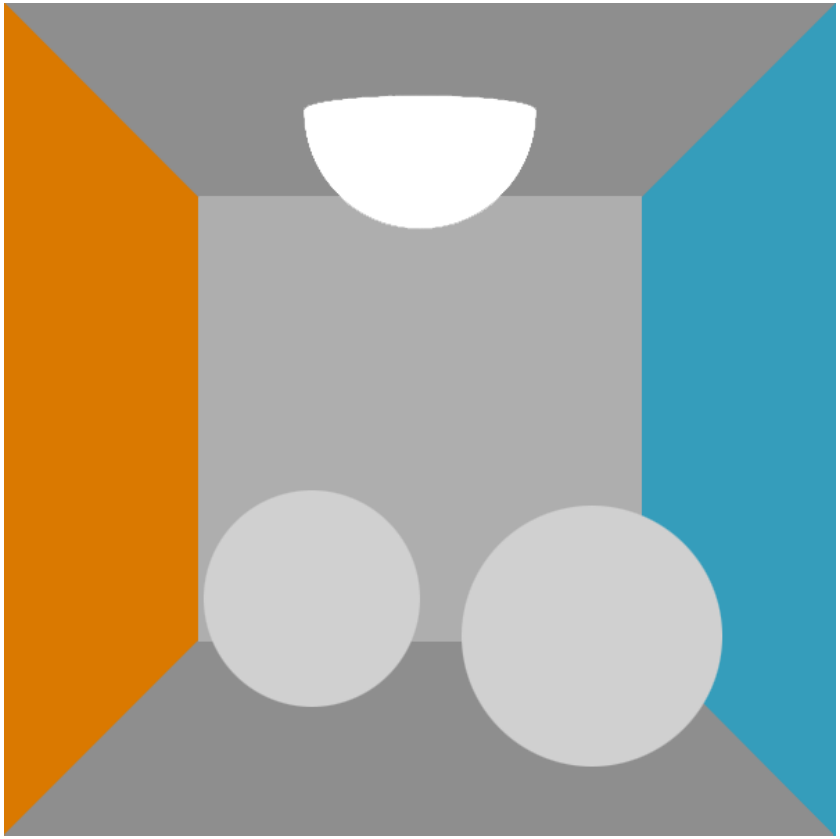
Schattenberechnung

- Analog können auch Schatten berechnet werden:
  - Schnittpunkte auf Oberflächen, die durch Verdeckung durch andere Objekte die Lichtquelle nicht sehen können.
  - Schnittpunkt-Test für den Strahl vom Schnittpunkt zur Lichtquelle.
  - Wenn ein Schnittpunkt existiert: Farbwahl = Schattenfarbe (meist schwarz)

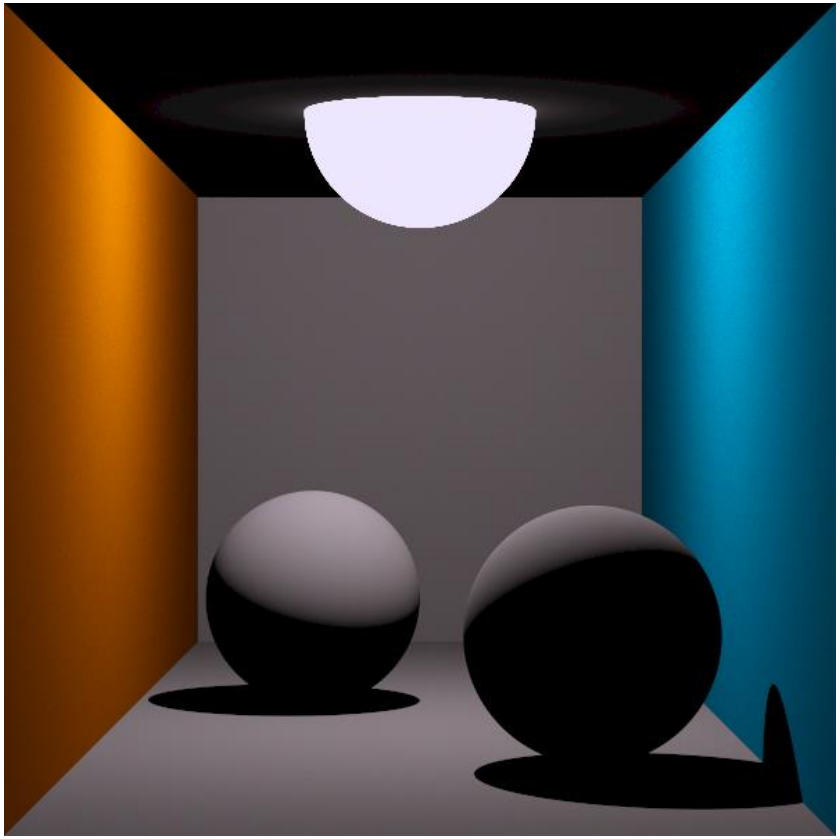


https://en.wikipedia.org/wiki/Ray_tracing_%28graphics%29

Beleuchtung & Schatten



Verdeckungsberechnung



Raytracing mit Beleuchtungsberechnung und Schatten

https://de.wikipedia.org/wiki/Raytracing

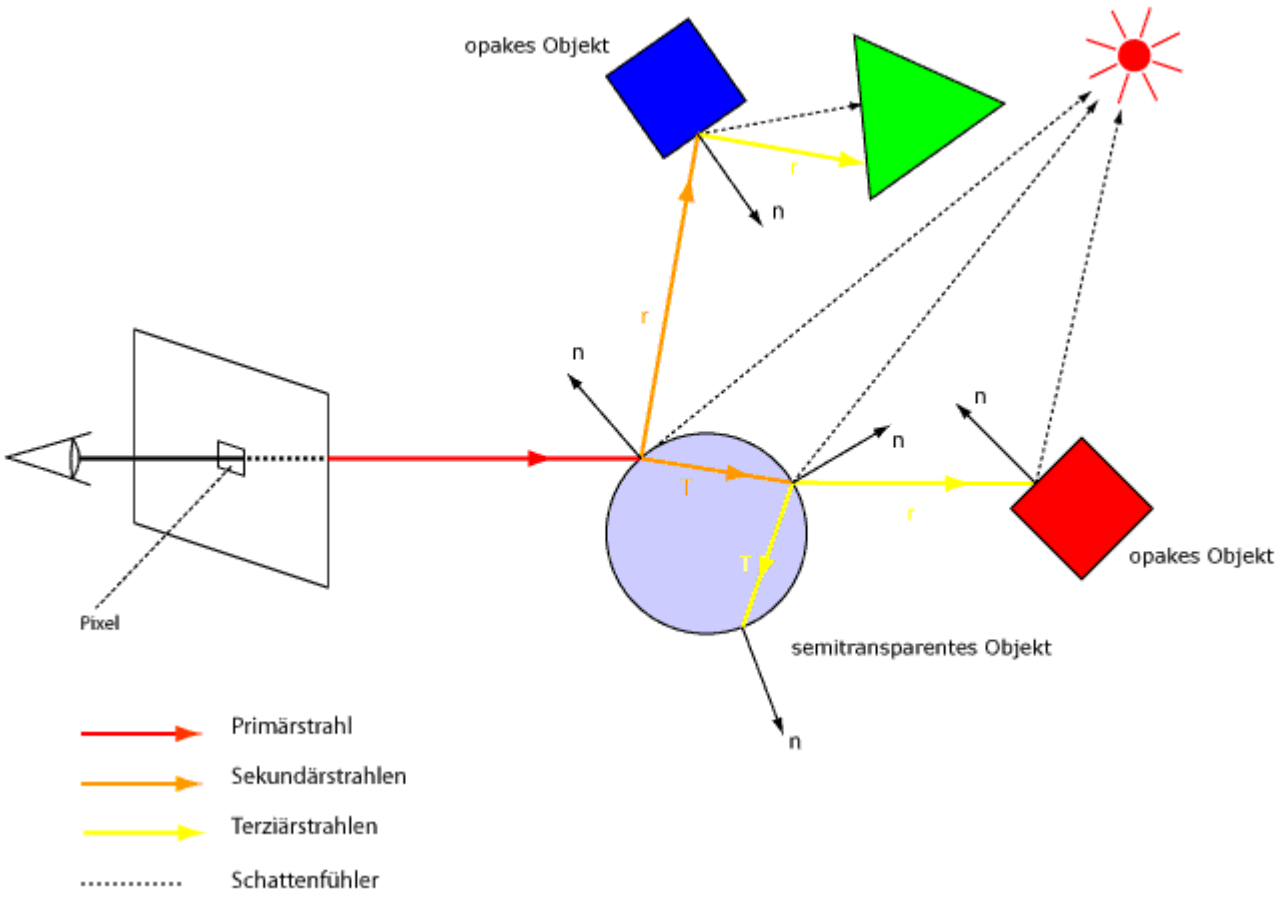
Rekursives Raytracing

- Verallgemeinerung: Verfolgung von Sekundärstrahlen durch Spiegelung und Brechung des Sichtstrahls.
  - Berechnung des Reflexionsvektor r durch Reflexionsgesetz: $r = l + 2(l \cdot N) \cdot N$
  - Für transparente Objekte Berechnung des Transmissionsvektors t durch Brechungsgesetz: $t = \frac{n_1(l - N(l \cdot N))}{n_2} - N\sqrt{\frac{n_1(1-(l \cdot N)^2)}{{n_2}^2}}$
  - Beachtung des Fresnel-Effektes in Abhängigkeit des Auftreffwinkels eines Sichtstrahls auf die Oberfläche: Gewichtung der Farbintensitäten der Sekundärstrahlen

- Entscheidungskriterium für Weiterverfolgung: Abbruch bei erreichter Rekursionstiefe oder wenn Strahl kein Objekt trifft.

- Pixelfarbe ergibt sich durch Integration über die Primär- und Sekundärstrahlen (Beleuchtungsberechnung!)

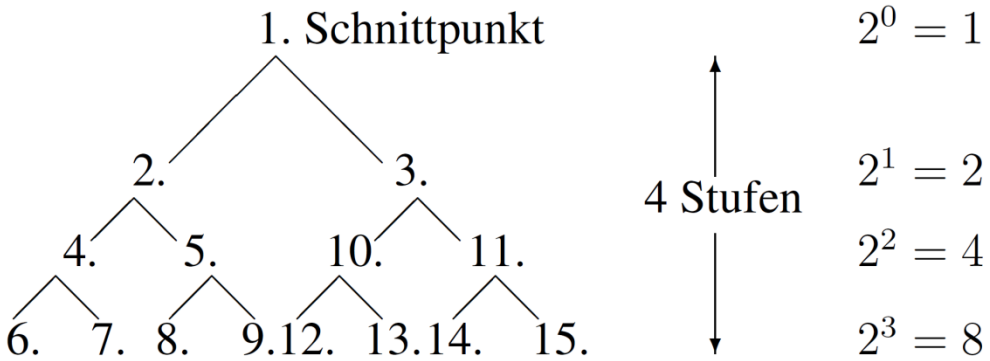
Rekursives Raytracing



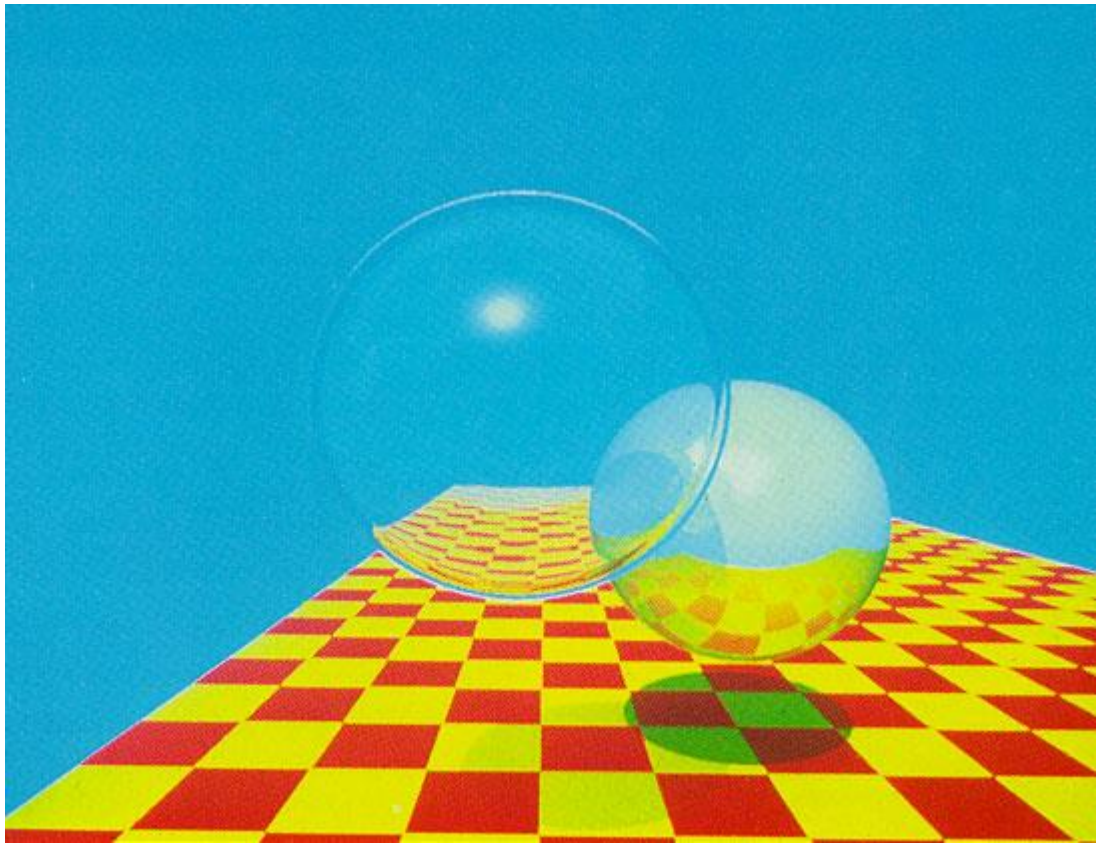
http://www.dma.ufg.ac.at/app/link/Grundlagen%3A3D-Grafik/module/8653?step=all

Rekursives Raytracing

- Implementierung: Binärer Baum
  - Bei jedem Auftreffen auf eine Oberfläche, entsteht ein reflektierter und gebrochener Strahl
  - Jeder dieser Strahlen schneidet erneut ein Objekt usw.
  - Es entsteht ein binärer Baum: bei Schachtelungstiefe n weist er eine Anzahl von $\sum_{i=0}^{n-1} 2^i = 2^n - 1$ Schnittpunkten auf.

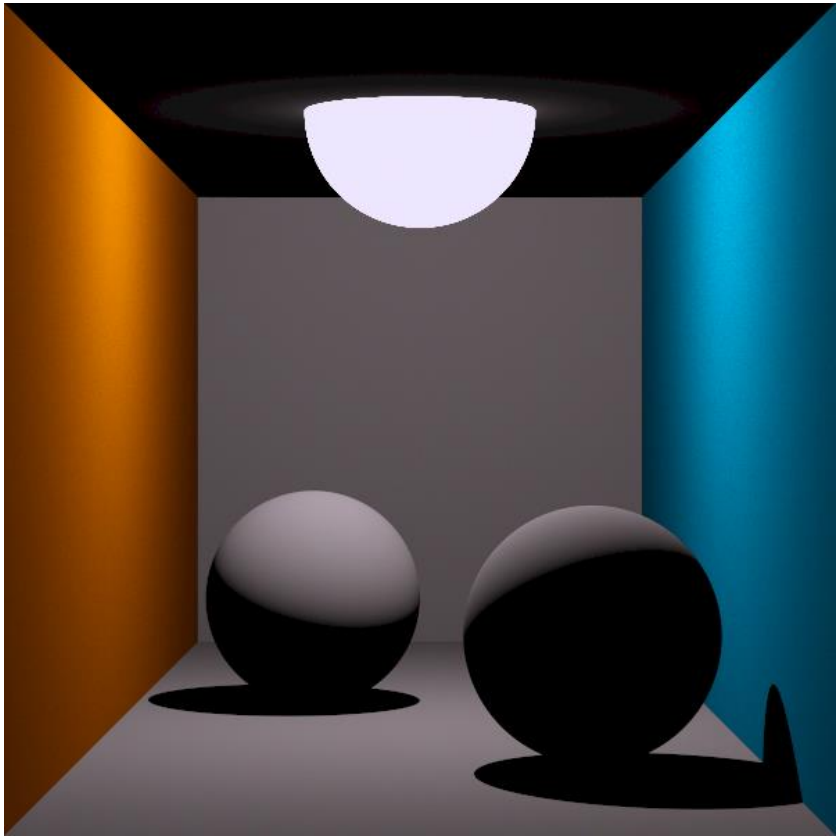


Rekursives Raytracing

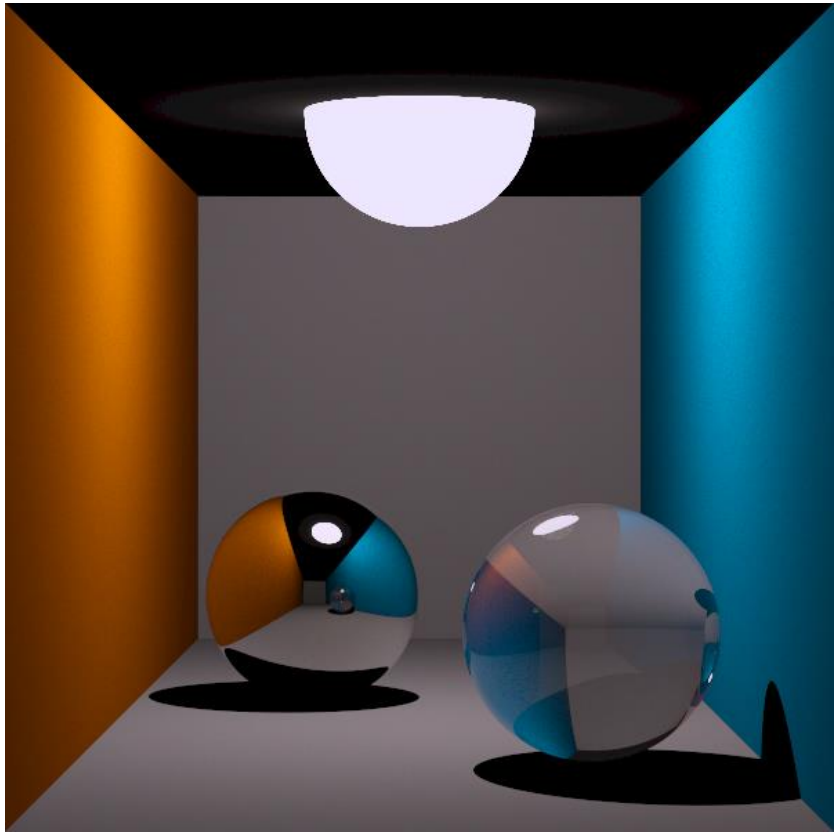


https://de.wikipedia.org/wiki/Raytracing

Rekursives Raytracing



Raytracing mit Beleuchtungsberechnung und Schatten

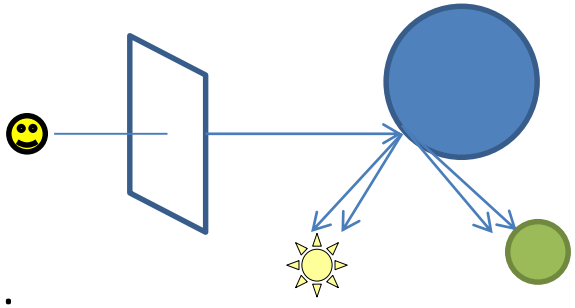


Rekursives Raytracing

https://de.wikipedia.org/wiki/Raytracing

Diffuses Raytracing

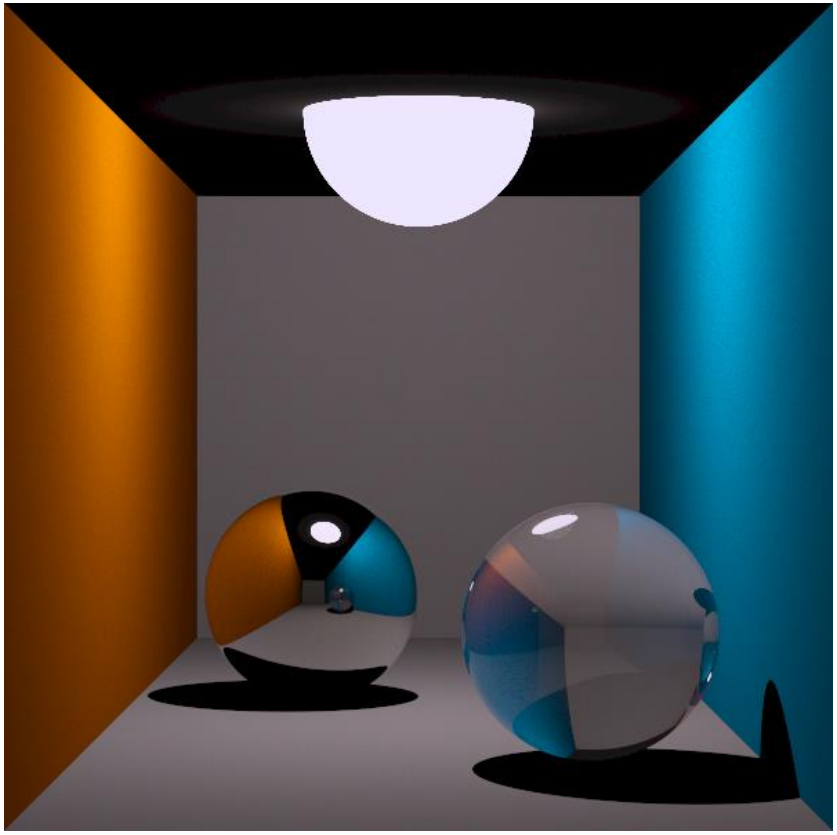
- Grundidee: stochastische Verteilung mehrerer Strahlen statt nur einem gebrochenem/reflektierten Strahl
  - Simulation von weichen bzw. verschwommenen Eigenschaften



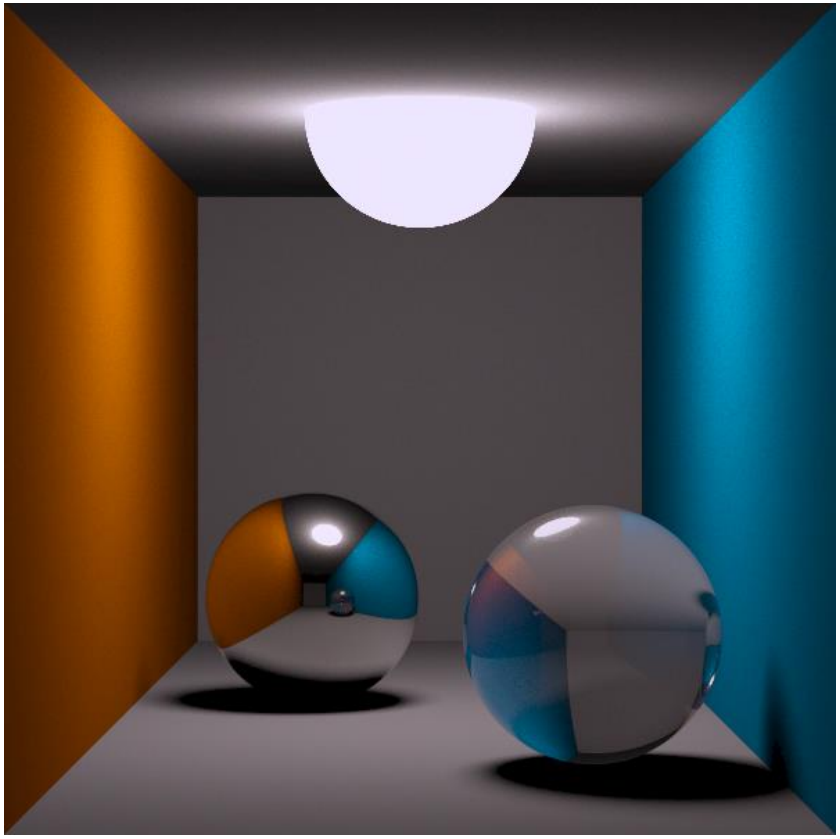
- Erzielbare Effekte:
  - Echte, weiche Lichtschatten (Abtastung der sichtbaren Oberfläche der Lichtquellen)
  - Verschwommene Lichtreflexionen auf glänzenden Oberflächen
  - Tiefenunschärfe (Abtastung der Linsenoberfläche)
  - Bewegungsunschärfe (Aufintegration der Strahlen über die Zeit)
  - Antialiasing (Mittelwert mehrerer Strahlen pro Pixel)

- Deutlich höherer Aufwand!
- Erlaubt noch keine Berechnung globaler Beleuchtung
  - Keine Sekundärstrahlen bei diffus streuenden Objekten

Diffuses Raytracing



Rekursives Ray Tracing



Diffuses Ray Tracing

https://de.wikipedia.org/wiki/Raytracing

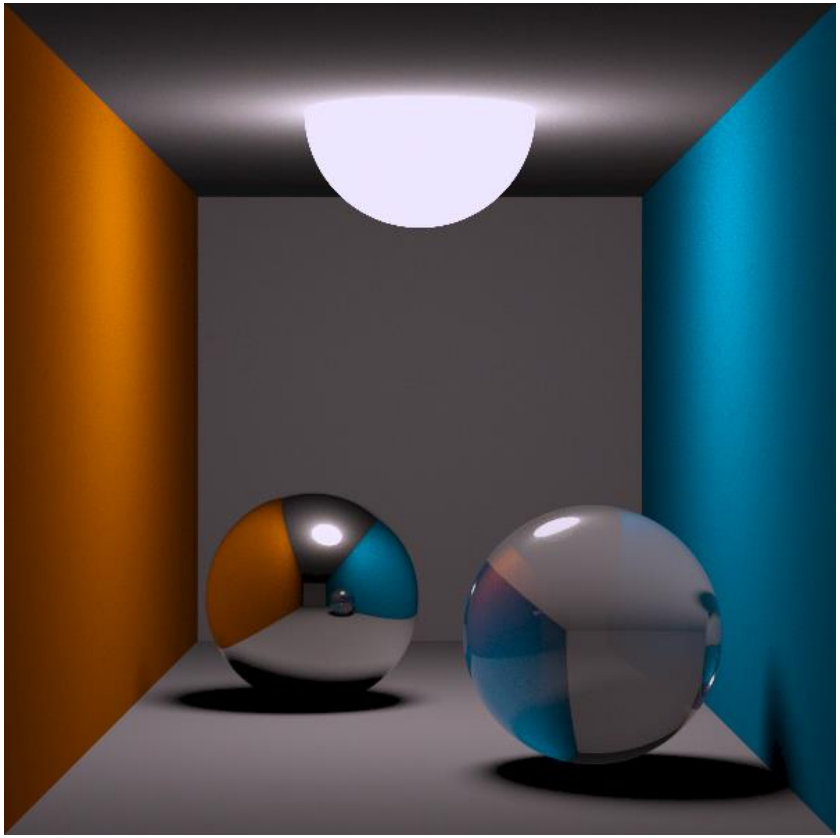
Path Tracing

- Diffuses Raytracing löst die Rendergleichung näherungsweise für spiegelnde Objekte (Strahlverfolgung nur in Reflexions- und Transmissionrichtung, Beleuchtungsrechnung).
- Grundidee von Path Tracing: „echte“ Verfolgung von zufälligen Strahlen.
- Integration über viele zufällige Strahlen pro Pixel in verschiedene Richtungen.
- Erlaubt Berechnung von Kaustiken
  - =Bündelung von Lichtstrahlen durch Objekte

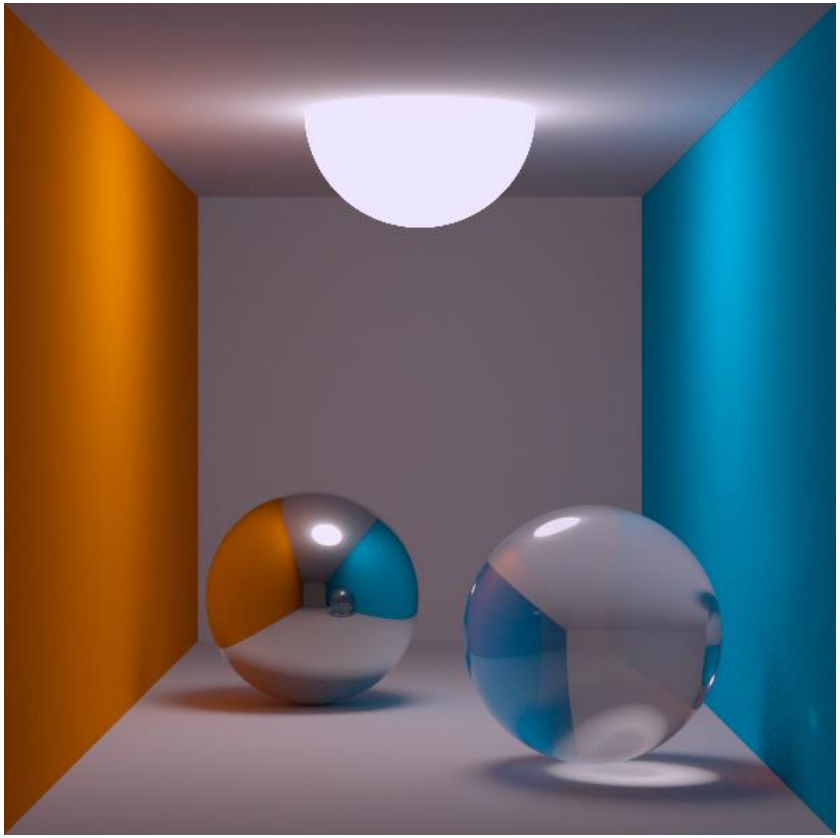
- Hoher Aufwand
- Rauschen bei zu niedriger Strahlanzahl

- Seltene Erweiterung: Light Raytracing:
  - Umkehrung der Tracing-Richtung: Strahlen ausgehend von der Lichtquelle

Path Tracing



Diffuses Raytracing



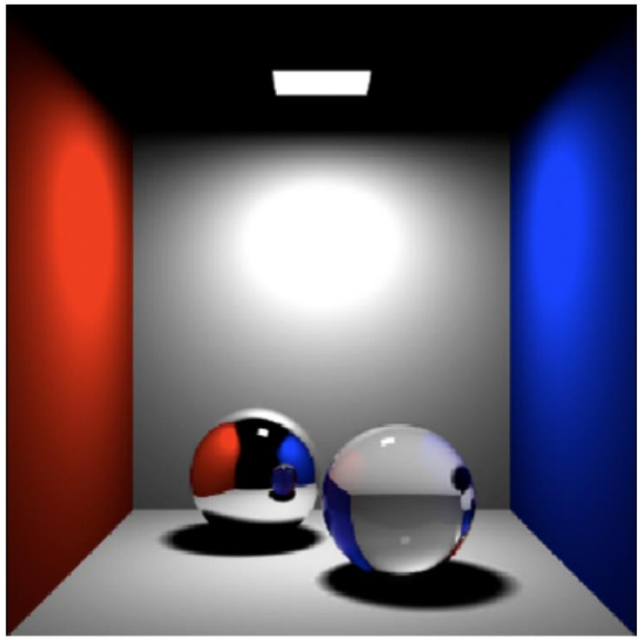
Path Tracing

https://de.wikipedia.org/wiki/Raytracing

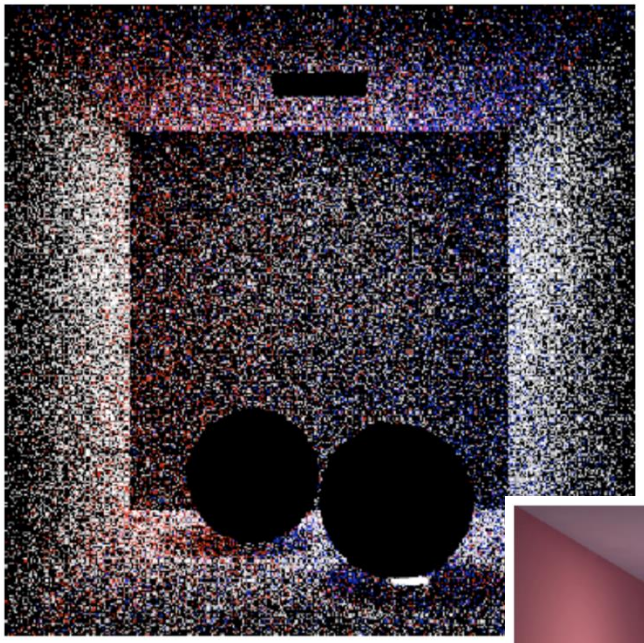
Photon Mapping

- Grundidee: Ermitteln der indirekten Beleuchtung durch umgekehrte Tracing-Richtung:
  - Strahlen werden ausgehend von Lichtquelle in die Szene „geschossen“
  - Reflexion, Brechung, Streuung, Absorption
  - Speicherung in Photon Map wenn Strahl auf diffus reflektierende Oberfläche trifft
    - Spezielle Datenstruktur, meist dreidimensionaler Baum
- Kombination mit „Vorwärtsrichtung“ des Raytracings
  - Wenn ein Strahl auf eine diffuse Oberfläche trifft, wird die indirekte Beleuchtung durch die Photon Map bestimmt.
  - Addition von indirekter Beleuchtung (Photon Map) und direkter Beleuchtung (Diffuses Raytracing, Path Tracing) ergibt globale Beleuchtung
- Erweiterung um Kaustik Map: Beschleunigung von Path Tracing

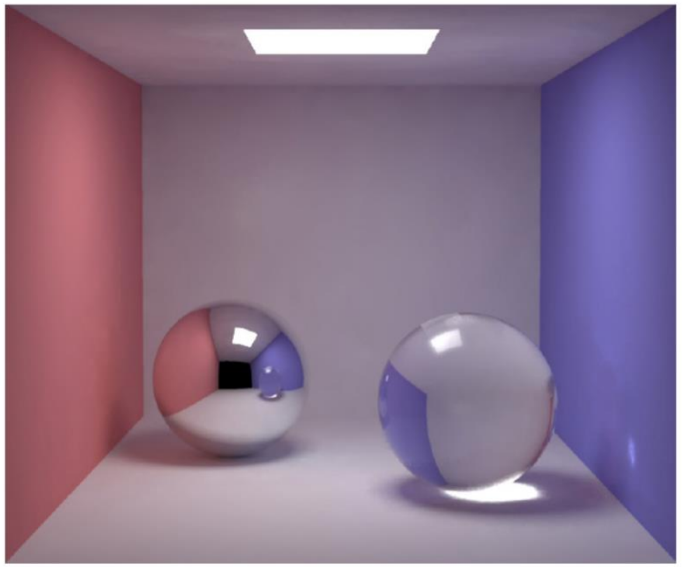
Photon Mapping



Diffuses Ray Tracing



Photon Map



Kombination: Photon Mapping

S. Krömker: Computergrafik I, Universität Heidelberg

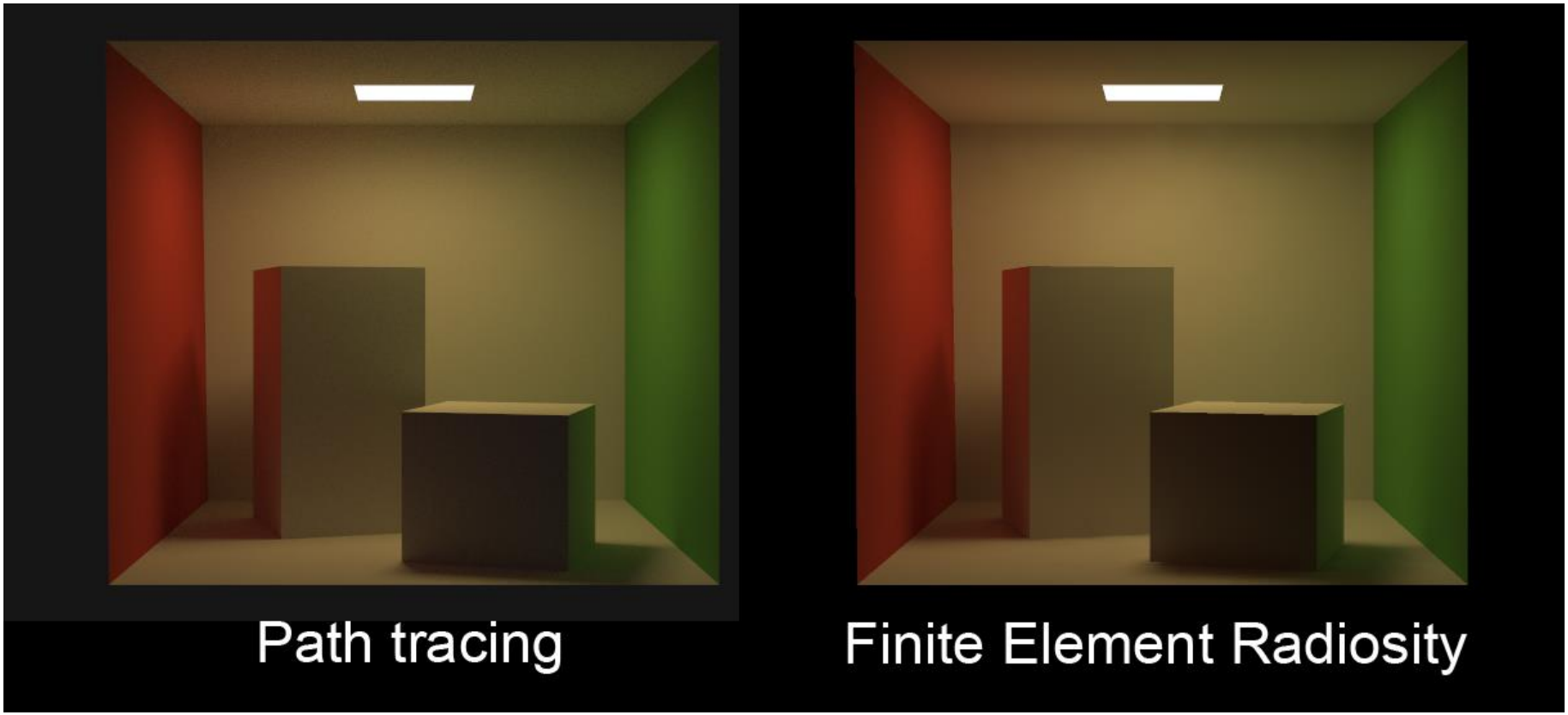
Photon Mapping

https://rosgar.wordpress.com/tag/yafaray/

Radiosity

- Alternative zu Raytracing-Verfahren für globale Illumination
- Beruht auf Energieerhaltungssatz: Alles Licht, das auf eine Fläche fällt und von dieser nicht absorbiert wird, wird von ihr zurückgeworfen.
- Nicht vom Blickpunkt abhängig: Beleuchtung für die gesamte Szene unabhängig von der Position des Betrachters berechnet.
  - Blickpunktabhängige Verdeckungsberechnung in unabhängigen Schritt.
- Alle Oberflächen ideal diffuse Reflektoren
- Löst die aus der Rendergleichung abgeleitete Radiosity-Gleichung
- Sehr aufwendige Berechnung, heute Tendenz eher zu Path Tracing, Photon Mapping um etwa gleiche Ergebnisse zu erreichen

Radiosity



http://www.deluxerender.com/2017/01/comparing-two-global-illumination-methods/

Beispiel: Brigade Rendering Engine

https://www.youtube.com/watch?v=FbGm66DCWok

ZUSAMMENFASSUNG

Zusammenfassung

- Berechnung von Reflexionen und Transmission von Licht → globale Illumination!
- Standard-Raytracing:
  - Prinzip der Strahlverfolgung vom Betrachter in die Szene
  - Verdeckungs-, Beleuchtungs-, Schattenberechnung
- Rekursives Raytracing:
  - Verallgemeinerung: Mehrfachreflexionen/-brechung
- Diffuses Raytracing:
  - stochastische Verteilung mehrerer Strahlen → weiche Darstellung
- Path Tracing:
  - Lösung der Rendergleichung für diffus streuende Objekte → indirekte Beleuchtung
- Photon Mapping:
  - Ermitteln der indirekten Beleuchtung durch umgekehrte Tracing-Richtung → Alternative und/oder Beschleunigung
- Radiosity:
  - Alternatives Verfahren zur globalen Beleuchtung
  - Lösung der diskreten Radiosity-Gleichung

ÜBUNGSAUFGABEN

Übungsfragen Kapitel 10

- Erläutern Sie kurz das Prinzip des rekursiven Raytracings
- Erläutern Sie den Unterschied zwischen Standard- (rekursivem) Raytracing und diffusem Raytracing? Welche Effekte lassen sich dadurch erzielen?