16. Animation

Begriffsklärung:

(1) *Animation*: Gesamtheit der Methoden zur Erzeugung synthetischer Bewegtbilder

(2) *Animation*: die Computergrafik-Methoden, die der "Belebung" (Bewegung, Veränderung) dienen

(3) *Animation*: ein vermittels (1) oder (2) erzeugtes Produkt auf Bewegtbildmedien

Ursprung: *anima* (Lufthauch, Atem)

- *Bewegung* ist eine eigenständige Wahrnehmungsqualität des visuellen Systems
- "Höhere Eigenschaft" als nur die Summe einzelner Standbilder, basiert auf Erfahrung und Konsistenz der Einzelereignisse (best fit)
- neben Form und Farbe *dritte Grundgröße der visuellen Wahrnehmung*

Motivation zur Nutzung von Animation, Multimedia etc.:

- möglichst optimale Nutzung der menschlichen Wahrnehmungskanäle
- Aufmerksamkeit ist größer als bei starren Bildern (Grund: Bedeutung von Bewegung in der natürlichen Umgebung der stammesgeschichtlichen Vorfahren des Menschen)
- Berücksichtigung bestehender Sehgewohnheiten (dominierende Rolle des Fernsehens und der Computerspiele)

Auge und Gehirn interpretieren Folge von Bildern als kontinuierliche Bewegung ("persistence of motion")

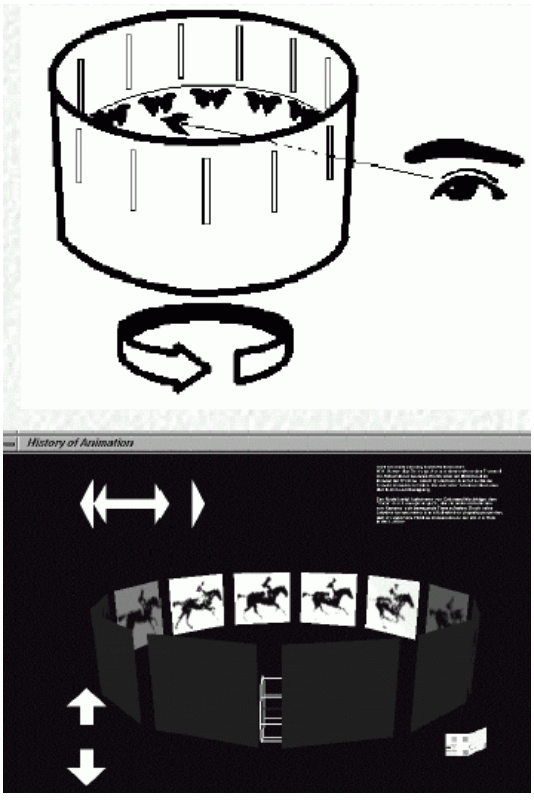
Bedingungen dafür:

- passende Geschwindigkeit der Abfolge (situations- und bildabhängig; Bildwiederholrate: Anzahl Bilder / s; Samplingrate: Anzahl verschiedener Bilder / s)
- entsprechendes Bildmaterial, d.h. genügend Überlappung bei aufeinanderfolgenden Bildern

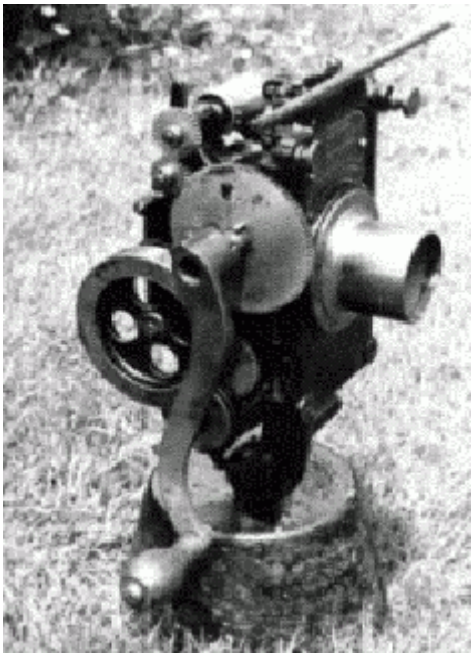
Geschichtlicher Überblick zur Animation
(nach Geiger 2000 und Krömker 2001)

1824: Mark P. Roget publiziert das Prinzip der "Persistence of Vision". Die Wahrnehmung eines visuellen Reizes erfolgt erst nach einer Latenzzeit und bleibt eine Zeit nach Verschwinden des Reizes bestehen.

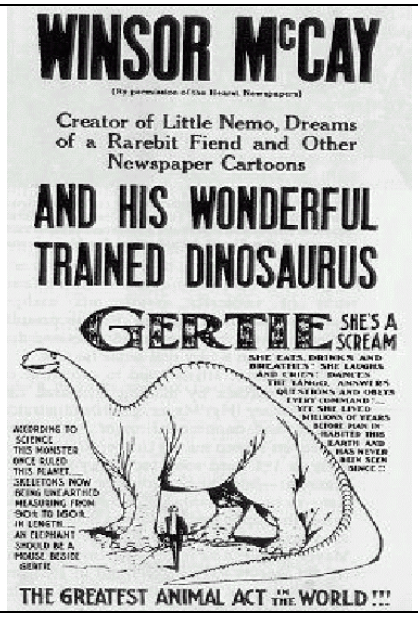
um 1830: verschiedene *Animationsmaschinen* werden entwickelt, ab 1850 werden darin die ersten Fotografien statt Zeichnungen eingesetzt

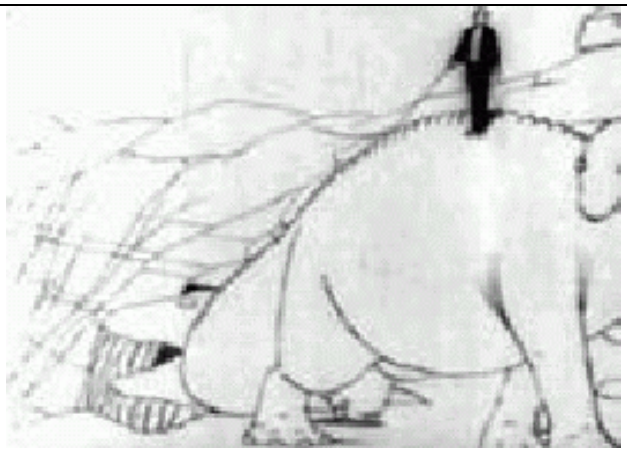


1870: E. Muybridge entwickelt die Chronophotographie zur Aufnahme von Bewegungen.
1880: Thomas A. Edison entwickelt einen Filmprojektor (Kinetoscope).
Die Entwicklung des Rollfilms und neue Projektortechniken ermöglichen realistische Filmaufnahmen, die zunächst die bisherigen Animationen verdrängen.

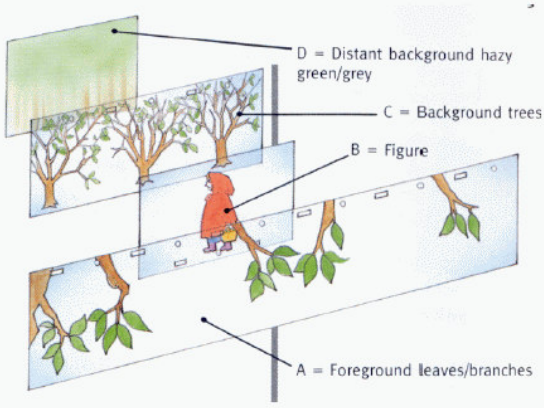


1899 entwickelt A. Melbourne-Cooper die erste Filmanimation "Matches: An Appeal".
1900 animiert J. S. Blackton den Rauch einer Lokomotive und produziert 1906 die erste Cartoonanimation.
1907-09: E. Cohl entwickelt ca. 40 Cartoons mit weißen Streichholzfiguren auf schwarzem Grund
1908: Winsor McCay entwickelt eine Animation von Little Nemo, dem ersten Comic.
1912 entsteht die 5-min. Animation "Gertie, the trained Dinosaur" aus 10 000 Einzelbildern, die innerhalb von 2 Jahren gezeichnet wurden.

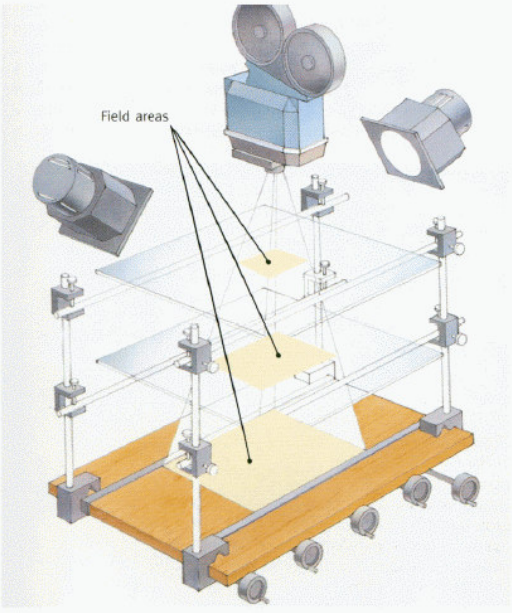




1915: Earl Hurd entwickelt die "Cel-Animation", die den Aufwand der Animationserzeugung drastisch verkürzt. Eine Szene wird mit Hilfe von transparenten Celluloid-Folien erzeugt, so dass nur die bewegten Elemente neu erstellt werden müssen.



Prinzip der Cel-Animation (früher Celluloid, heute Azetatfolie: i.d.R. hochtransparent; Hintergrund i.d.R. Papier)

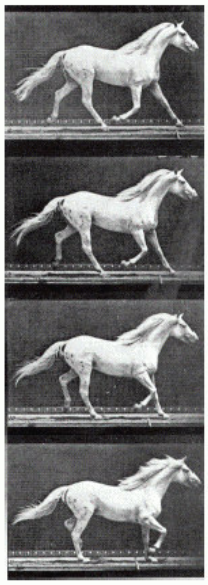


Möglichkeit zur Erzeugung von Tiefenschärfe

1921: *Walt Disney* beginnt mit den "Laugh-o-Grams" seine Trickfilmkarriere. 1922 Gründung der Disney Studios.

1928: "Steamboat Willy", der erste Zeichentrickfilm mit synchronisiertem Ton (Hauptdarsteller: Mickey Mouse). Höhepunkte: 1937-40 Fantasia, Pinocchio, Schneewittchen.

Traditioneller Zeichentrick



Eadweard Muybridge: *Animals in Motion , 1899*

Tom and Jerry, MGM

Prinzipien entwickelten sich in späten 20ern und 30ern Jahren:

„How it moves is more important than what moves.“

Norman McLaren

Reale Bewegungen zu imitieren ist nicht verstandene Animation (das kann der Lifefilm besser): **Animation lebt von der Variation und Verstärkung von Zeit, Geschwindigkeit und Form.**

Computeranimation unterstützt heute die 2D-Cartoonanimation beim Erzeugen der Bilder (Scanner, grafischer Editor, Erzeugen von Bewegung: "Inbetweens", Colorierung, Nachbearbeitung).

Achtung: Cartoonanimation ist keine eingeschränkte 3D-Animation!

- oft ist verzerrte Darstellung gefragt
- spezielle Werkzeuge
- typische Prinzipien

1932: Willis O'Brien produziert den Spielfilm "King Kong" und verwendet dabei die *Stop-Motion-Technik* für die Animation des Gorillas. Dabei wird das zu animierende Objekt gefilmt, anschließend das Objekt manipuliert, wieder abgefilmt usw.

Computer-Animation: war und ist angewiesen auf leistungs-fähige Hardware

1962: Fetter, Bernhard (Boeing) – Abfilmen geplotteter 3D-Zeichnungen

1963: erste Computeranimation auf dem "Whirlwind" (MIT) – Charly Adams stellt einen springenden Ball auf dem neuen Grafikdisplay als erste Anwendung dar

E. Zajac publiziert die Erzeugung bewegter Bilder zur Visualisierung eines Satelliten-Kontrollsystems und verwendet erstmals den Begriff "Computer Animation".

1965 Sketchpad (Ivan Sutherland): erstes Animationssystem, interaktiv, erste GUI.

Sketchpad

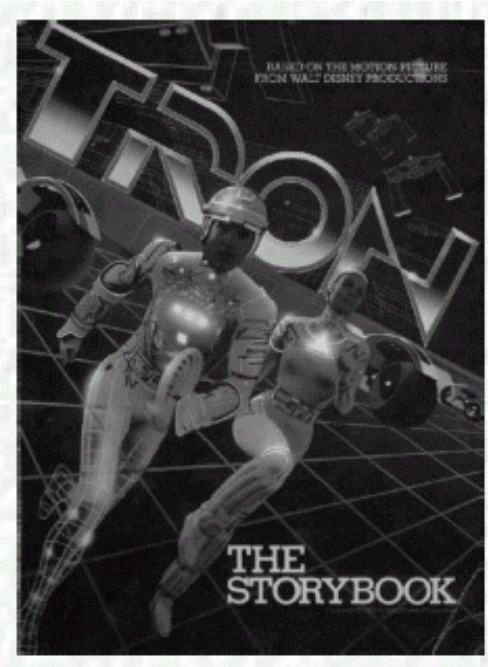
1965-1975 Entwicklung der grundlegenden Algorithmen zum Rendering (Shading, Texture Mapping)

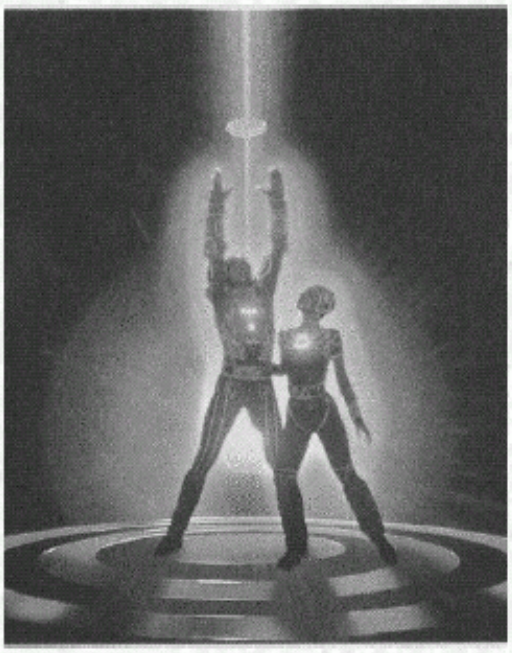
1971: N. Burtnik & M. Wein entwickeln ein keyframing-System, welches zwischen Schlüsselbildern mit wenigen grafischen Primitiven interpoliert. 1973 führen sie die skelettale Kontrolle in ihr System ein (komplexe 2D-Bilder werden interpoliert, wenn ein einfaches Skelett angegeben wird)
1974 gewinnt der von P. Foldes mit diesem System erzeugte Film "Hunger" einen Preis in Cannes:

1975-1985: Zeit der "großen" Produktionshäuser (eigene Software, teilweise spezielle Hardware)

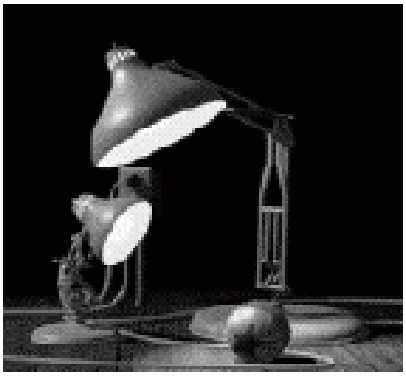
1982 "Dream Flight" von N. Magnenat-Thalmann und D. Thalmann: 13-min. Film, der als erster fiktionaler 3D-Film vollständig im Computer erzeugt wurde.

Disney produziert mit "Tron" den ersten Film, der 3D-Computergrafik zur Verfremdung der Schauspieler benutzt.





1986: Pixar Inc. produziert "Luxo Jr.", eine 3D-Animation zweier Schreibtischlampen, die durch Verwendung zahlreicher neuer Animationstechniken zum Meilenstein der Computeranimation wird. Der Film ist der erste computeranimierte Film mit einer Oscar-Nominierung.



1987: die Thalmanns produzieren "Rendez-vous à Montréal", wo eine virtuelle Marilyn Monroe einen virtuellen Humphrey Bogart trifft

weitere visuelle "Meilensteine": "Star Trek II: The Wrath of Khan" (Lucasfilm), Max Headroom, "Toy Story", "Jurassic Park" (Steven Spielberg)

Computerspiele als (weiterer) wichtiger Motor der Entwicklung!

1985-1995: Konzentration der Algorithmen und Technologie auf Optimierung, Qualität und einfachere Bedienbarkeit
Verbesserungen bei Radiosity, Gesichts-Animation, physikal. Simulationen, Fraktalen
führende Animationssysteme: ALIAS, Wavefront, Softimage

heute: High-end PCs und Workstations lösen die Spezial-systeme zunehmend ab

Aktuelle Forschungsaktivitäten:

- Human Animation, Facial Animation
- Artificial Life
- "Super Reality"
- Image Based Rendering
- Global Illlumination (Radiosity)
- Morphing
- neue Systemarchitekturen
- Netzwerkfähigkeit
- Spiele
- Benutzerorientierung: Einbeziehung von 3D-Animation in Benutzungsoberflächen; "Human Centered Computing"

Terminologie zum Film / zur Animation:

presentation, feature, film	Präsentationseinheit (kann, muss aber nicht Teil einer Serie sein)
Szene (scene)	gekennzeichnet durch Kontinuität im Ort, bei Figuren, Aktionen
Akt	Folge von Szenen, die eine größere Episode beschreiben
shot	zusammenhängende Serie von Einzelbildern, erzielt mit durchlaufender Kamera
sequence	subjektive Einheit, bestehend aus mehreren shots, die denselben Aspekt oder Moment einer Aktion darstellen
frame (auch: still)	Einzelbild *"an animator is a craftsman whose bricks are frames"* (J. Rosbush)
cue (mark, event)	ein frame, an dem ein Ereignis stattfindet

Übersicht Animationsmethoden im engeren Sinne:

klassische Techniken:

- Image interpolation / Transformation zwischen keyframes
  (in-betweening)
- Cel-Animation (Overlay von einzelnen Zellen: 2 1/2 D)

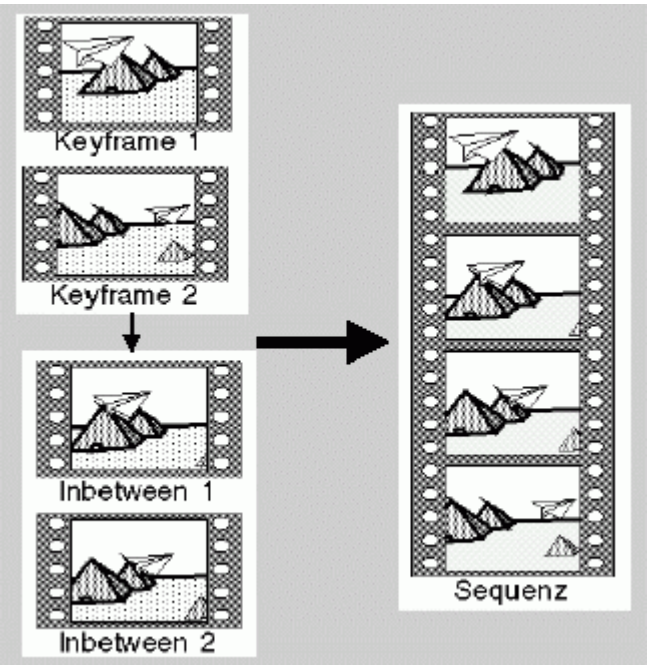
modernere Techniken:

- 3-D parametrisches Keyframing
- Kinematik und inverse Kinematik
- Dynamik, physikalische Simulation
- Skripting
- Artificial Life-Techniken

Keyframe-Animation

Computer-Version der traditionellen (Zeichentrick-) Keyframe-Animation

Keyframes: ausgewählte Standbilder, die vorgegeben (modelliert, akquiriert) sind
Zwischenbilder (*inbetweens*) werden konstruiert / berechnet.



2 Ansätze:

- bildbasiert, d.h. Interpolation der Pixel
- parametrisch, d.h. Interpolation im Modell (im Objektraum)

Bildbasierte Animation:

- Modifikation auf Pixelebene (2D)
- als Postproduction bei Filmen (z.B. Indiana Jones IV)
- Vorläufer aus der Biologie (D'Arcy Thompson: *On growth and form*, demonstriert morphologische Verwandtschaft zwischen verschiedenen Arten durch geometrische, stetige Transformation der Formen) und in alten Filmen (The Wolfman, 1941)
- Transformation von einem Objekt in ein anderes (Metamorphose), Ziel: möglichst stufenloser Übergang

"Morphing": Begriff gebräuchlich für 2D und 3D;
hier 2D

Morphing: Metamorphose von Objekt *A* zu Objekt *B*, d.h. Pixel ändert Position und Wert

Warping: Morphing von Objekt *A* zu verzerrtem Abbild *A'*, Verschieben von Pixeln bei gleichem Farbwert

Farbtransformation: Pixelwert-Veränderung bei gleicher Position, z.B. *Cross-Dissolve* (fade-in *B* + fade-out *A*)

Morphing von Bildern

Morphing= Warping+Farbtransformation

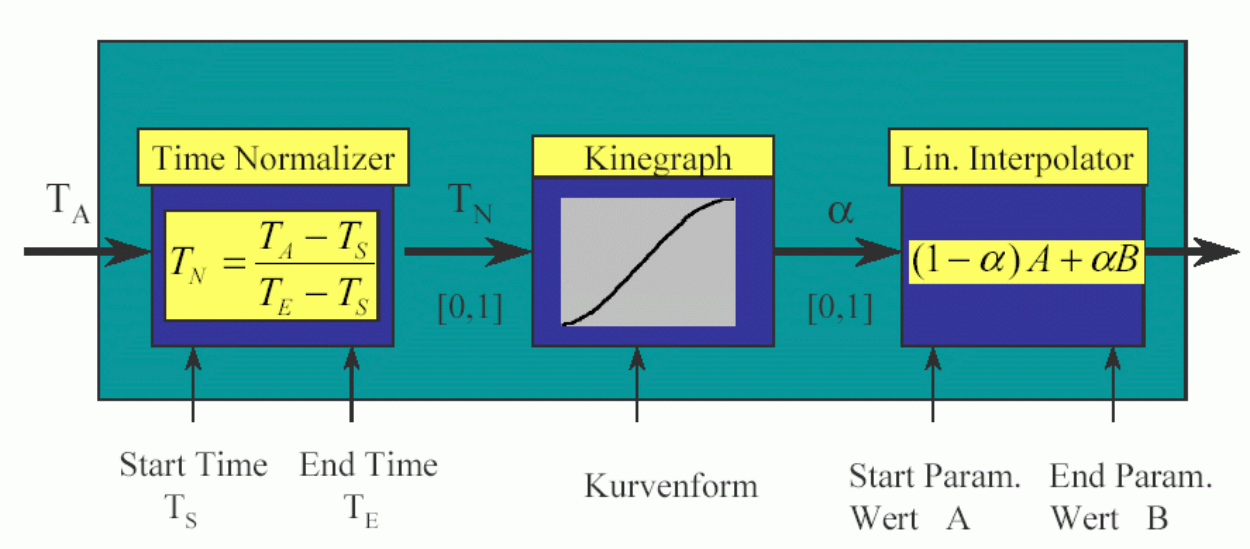
Sei C(x,y): Pixel der Farbe C an Position (x,y)

Morphing: S(x,y) -> T(x',y')
Warping: S=T und (x',y') = *warp*(x,y) und warp ist der benutzte Warpingalgorithmus (z.B. two spline mesh Verfahren, feature based morphing)
Farbtransformation: S(x,y) -> T(x,y), z. B. *cross-dissolve*: Pixel (x,y) hat Farbe I zum Zeitpunkt $t, t_{\min} \le t \le t_{\max}$

$$I(x,y,t) = S(x,y)(1 - \frac{t - t_{\min}}{t_{\max} - t_{\min}}) + T(x,y)(\frac{t - t_{\min}}{t_{\max} - t_{\min}})$$

die Gewichtungsfunktionen für $S(x, y)$ und $T(x, y)$ können verändert werden, z.B. langsamere Veränderung am Anfang, dann immer schneller!
"Kinegraph" legt den Verlauf der Gewichtsfunktion fest

Allgemeiner Parameter-Interpolator (auch für parametrische Interpolation im Objektraum):



Übliche Kinegraphen (Kurvenformen):

- hold (konstante Phasen, Plateau)
- linear
- ease in (slow in) – "weiches" Einsetzen
- ease out (slow out) – analog für die Endphase
- ease in ease out
- ...
- sketched (von Hand vorgegeben)
- splined
- Sprünge

Übliche Ease-Funktionen:

- quadratisch: $\alpha = 2T_N^2$ für $T_N = [0,0.5]$ ease in
  $\alpha = 2T_N - 2T_N^2$ für $T_N = (0.5,1]$ ease out
- sinusoidal: $\alpha = \sin(T_N \cdot \frac{\pi}{2})$
- diese Funktionen garantieren keine Kontinuität der Änderungsgeschwindigkeit an den Key-Punkten

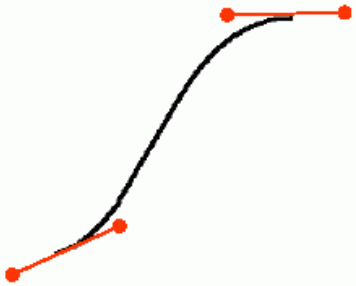
⇒ Verwendung von Splinefunktionen als Kinegraph

Basisgleichung: $\alpha = a \cdot T_N^{\,3} + b \cdot T_N^{\,2} + c \cdot T_N + d$

mit

$$a = m + n - 2$$
$$b = -2m - n + 3$$
$$c = m$$
$$d = 0$$

m: Steigung am Start-Key
n: Steigung am End-Key



übliche Kontrolltechniken für Spline-Kinegraphen:

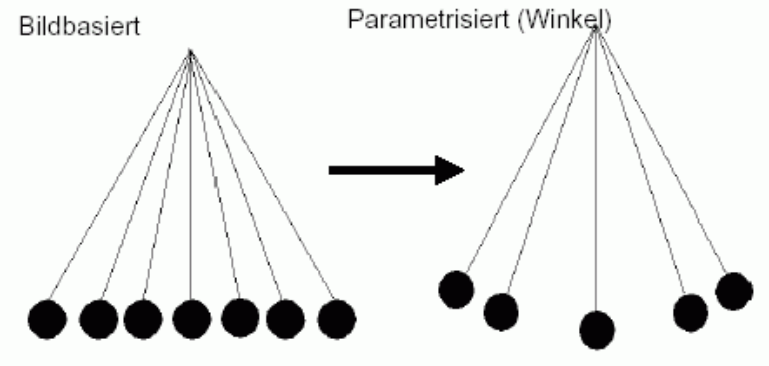
- Benutzer gibt Steigungswerte explizit an (numerisch oder über grafische Editoren)
- System macht Annahmen über Steigungswerte, z.B. Gerade zwischen vorhergehendem und nachfolgendem Key-Wert
- Benutzer gibt Zeitdauer (in Frames) für ease-in und ease-out an

3D-parametrisches Keyframing

- 3D-Szenenbeschreibung wird parametrisch dargestellt (im Objektraum)
- Parameterwerte dieser Beschreibung werden für Keyframes (ausgewählte Zeitpunkte) eingestellt
- Inbetweens (bzw. die entsprechenden Parameter) werden durch Interpolation ermittelt

Gründe für Übergang zu 3D-Parametrisierung und -Interpolation:

- Interpolation auf Pixelebene kann zu Schwierigkeiten bei 3D-Strukturen führen
- unnatürliche Effekte bei bildbasierter Interpolation:



Auswahl der zu interpolierende Parameter durch das Animationssystem oder durch den Benutzer

- potenziell kommt jedes Attribut der erstellten Modelle und der Szene in Frage
- z.B. Position, Orientierung, Materialeigenschaften, Gestalt, Textur, Umgebung...
- 1 Parameter für jeden Freiheitsgrad des Modells: können sehr viele sein, Über-Parametrisierung (= zu viel Aufwand)!
- potenziell für jeden Parameter andere key-Zeitpunkte
- nicht jedes Attribut ist wirklich geeignet zur Interpolation
- kein Animationssystem unterstützt / kennt alle Modellierungsverfahren
- Art der Interpolation ist wichtig, muss zum Typ des Parameters passen!

Beispiel: Interpolation von Positionen (3D-Vektoren)

Finde Weg zwischen zwei Positionen, oder genauer:
Finde Werte *P* zwischen zwei Stützstellen *X, Y* mit $P, X, Y \in \Re^m$

$$X = \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix}, Y = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix} \xrightarrow{???} P = \begin{pmatrix} p_1 \\ \vdots \\ p_m \end{pmatrix}$$

$$p(t) = (1-t)X + tY, t \in [0,1]$$

Was kontrolliert *t* ? Frameanzahl F_{end}-F_{start} oder Zeit T_{end}-T_{start}

$$t = \frac{T - T_{start}}{T_{end} - T_{start}} \qquad t = \frac{F - F_{start}}{F_{end} - F_{start}}$$

Lineare Interpolation

$$p(t) = (1-t)X + tY, t \in [0,1]$$

Animation von Werten (Beispiel Objektposition)

- Gegeben Startposition X, Endposition Y, Anfangszeit T_{start}, Endzeit T_{start} (analog auch Frames F, F_{end}, F_{start})
- Berechne t mittels Zeit T, T_{end}, T_{start} (oder F, F_{end}, F_{start})
- Berechne Objektposition P(x,y,z) mit t, X,Y
- Render Objekt an Position P(x,y,z)

Nicht-lineare Interpolation

- Bewegungen sind in der Realität stets nicht-linear und folgen dynamischen Gesetzmäßigkeiten
- Kinematische Gesetze beschreiben dies vereinfacht
- Approximation durch handhabbare mathematische Funktionen:

Trigonometrische Funktionen
Parametrische Überblendung (Quadratisch, Kubisch)
Hermite Überblendung
Parabolische Überblendung
Splines, insbesondere Catmull-Rom spline

Nichtlineare Funktionen:
häufig verwendet: Sinusfunktion ("Ein- und Ausschwingen")

$p(t) = (1-t)X + tY$ mit $t = \sin\alpha,\ 0° \le \alpha \le 180°$

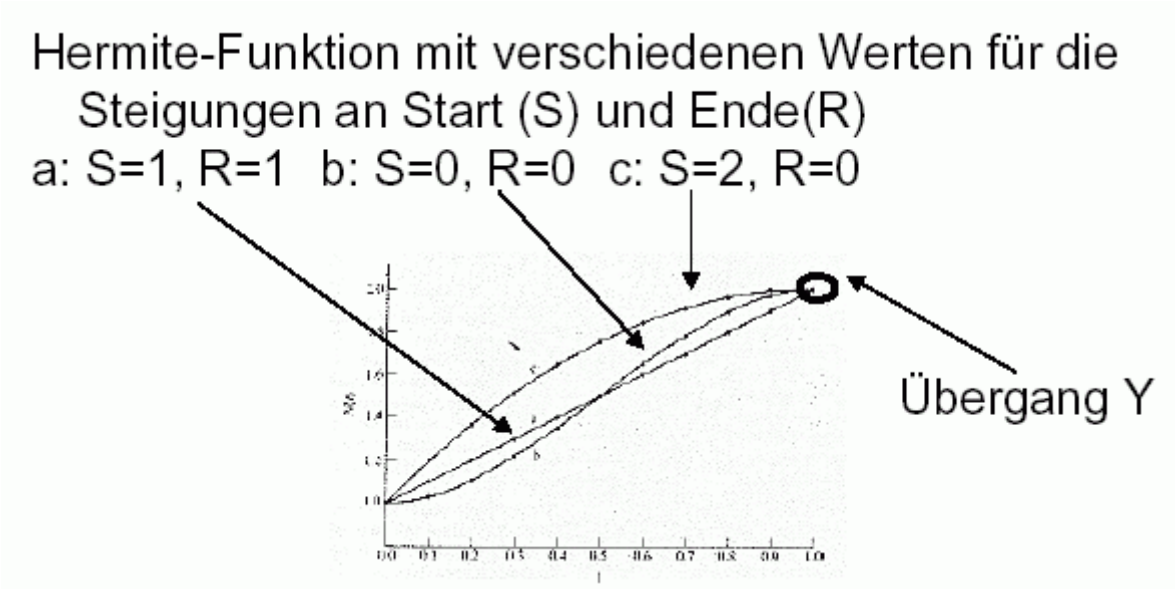
weitere Techniken:
Verwendung von Polynomen (quadratisches, kubisches Überblenden)

Hermite-Überblendung:

- Kontrolliere Start und Ende des Interpolationsvorgangs durch explizite Parameter S,R
- Benutze kubische Funktion für Überblendung
- Darstellung in Matrixform

$$p(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} * \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} X \\ Y \\ S \\ R \end{bmatrix}$$

Beispiel:



weitere Ansätze:

- Parabolische Überblendung (parabolic blending)
- Bézier-Kurven
- B-Splines
- andere Spline-Varianten

Problem:
Bei Positions-Interpolation entspricht die *Geschwindigkeit* der gewünschten Positionsveränderung oft nicht der Parametrisierung der gewählten Spline-Kurve

$\Rightarrow$ notwendiger Zwischenschritt:
Bogenlängenparametrisierung

- wenn parametrische Kurve $Q(u)$ gegeben: bestimme Funktion, die pro Parameter-Intervall du die zurückgelegte Weglänge (Bogenlänge) $s = A(u)$ auf der Splinekurve bestimmt
- dann Reparametrisierung der Splinefunktion zu $Q(s)$

Reparametrisierung notwendig, da erst die Splinekurve bekannt sein muss, damit die Bogenlänge als Parameter für die Splinebeschreibung dienen kann

- Problem: Bogenlängenfunktion $A(u)$ kann i.allg. nicht analytisch bestimmt werden

$\Rightarrow$ numerische Integration

aber das ist aufwändig

$\Rightarrow$ Approximation durch akkumulierte Bogensehnen (Verfahren der "Vorwärtsdifferenzierung") mit Stützstellen, die möglichst gleichmäßig auf der Kurve verteilt liegen

Interpolation von Rotationswerten

in 2D einfach, es genügt die Interpolation des Winkels

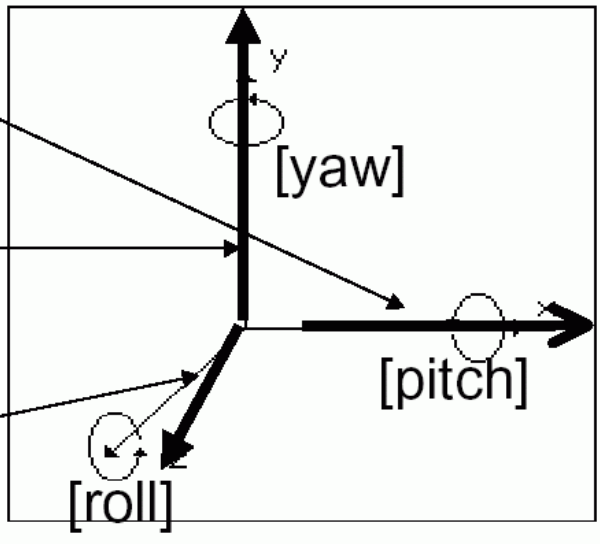
in 3D prinzipiell Kombination von Rotationen um 3 Achsen (x, y, z)
verschiedene Bezeichungsweisen der Rotationen um die Achsen:
X, Y, Z; pitch, yaw, roll (Navigation); H, L, U (Turtle-Geometry)
Orientierung der Achsen nach der Rechte-Hand-Regel

Darstellung in homogener Koordinatendarstellung

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(pitch) & -\sin(pitch) & 0 \\ 0 & \sin(pitch) & \cos(pitch) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(yaw) & 0 & \sin(yaw) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(yaw) & 0 & \cos(yaw) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(roll) & -\sin(roll) & 0 & 0 \\ \sin(roll) & \cos(roll) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$



Die entsprechenden Rotationswinkel um die 3 Grund-Achsen werden als "Euler-Winkel" bezeichnet
(Achtung: nicht für jede Bewegung eindeutig bestimmt!)

– es bietet sich an, diese für die Interpolation zu benutzen

Interpolation der Eulerwinkel

Eulerwinkelrotation

yaw: Rotation um Y-Achse
pitch: Rotation um X-Achse
roll: Rotation um Z-Achse

Für rechtshändiges Koordinatensystem:
positive Rotation ist, in Richtung des Ursprungs der betreffenden Achse betrachtet, stets gegen den Uhrzeigersinn

Folge von Rotationen ist nicht kommutativ

daher Festlegung, z. B. [Vince,92] roll->pitch->yaw :

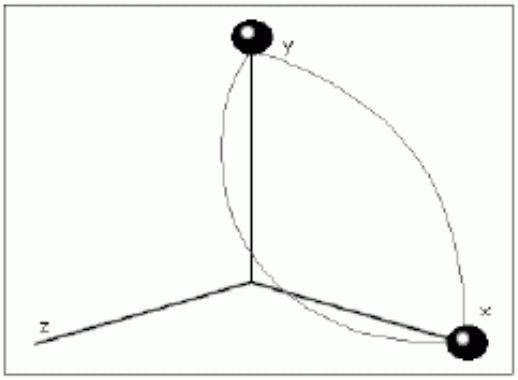
ergibt als Matrix:

$$[newPos] = [yaw][pitch][roll]\ [oldPos]$$

aber: diese Vorgehensweise ist oft zu abstrakt, nicht intuitiv

Probleme bei Eulerwinkel: Interpolation

- Weg ist nicht eindeutig vorhersagbar
- Nichtintuitive Bewegung bei Interpolation
- Schwierig vorhersagbar

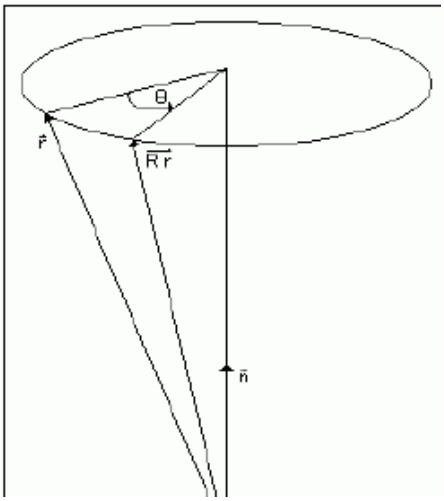


Alternative:
Charakterisierung der räumlichen Drehung durch Rotationsachse und Winkel
(so bei VRML und Java 3D)

Andere Darstellung aus Rotationsachse n und Winkel θ
Wie bei Eulerwinkel wird daraus Rotationsmatrix
berechnet, also statt R([roll],[pitch],[yaw]) -> $R(\theta,n)$
Drehung eines Vektors r zur
Position Rr um Winkel θ

Berechnung der Rotation
(vgl. Watt, S 359)

$$Rr = (\cos\theta)r + (1-\cos\theta)\, n(n\cdot r) + \sin(\theta)n\times r$$



- wenn Drehachse fest: Interpolation unproblematisch
- sonst erst Interpolation der Achsen, dann der Winkel
- auch hier kann es zu nichtintuitiven Ergebnissen kommen

weitere Alternative: *Quaternionen*

- Verallgemeinerung der komplexen Zahlen
  (hier in der Vorlesung nicht behandelt)
- können zur Charakterisierung von Rotationen in 3D herangezogen werden
- komponentenweise (lineare) Interpolation möglich

Zusammenfassung zum parametrischen 3D-Keyframing

Vorteile:

- universelles Animationsverfahren: anwendbar auf alle Modell-, Szenen- und Rendering-Parameter
- Parameter und Änderungsverhalten vollständig unter Benutzerkontrolle
- kombinierbar mit anderen Verfahren

Nachteile:

- nicht jeder Parameter eignet sich für eine Interpolation
- bei einigen Parametern zusätzliche Umformungen / Darstellungen nötig – Bsp. Rotationen (Euler-Winkel etc.); Farbspezifikationen (evtl. Wechsel des Farbmodells, um natürlichere Farbabstände zu erhalten)
- oft Schwierigkeit, verschiedene Parameter miteinander zu koordinieren: beachte Abhängigkeiten zwischen Parametern
- ggf. sehr viele Parameter zu kontrollieren: einige Hundert oder Tausend
- wünschenswert: Berücksichtigung von Einschränkungen, die sich z.B. durch physikalische Randbedingungen ergeben (dies ist zunächst nicht gewährleistet – sehr große Freiheiten!)

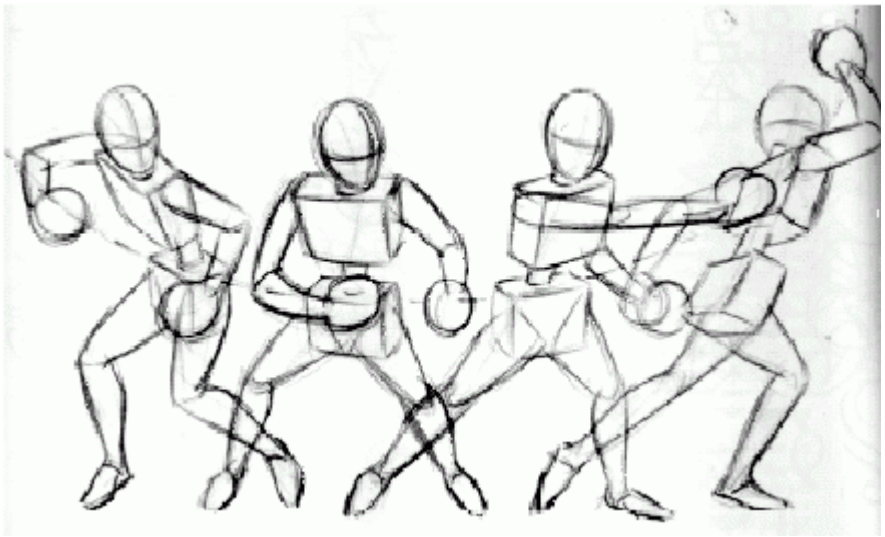
Constraints

- Constraints sind Bedingungen für Parameterwerte
- beschreiben Interaktionen und Randwerte
- einfache: Gleichheit, Max, Min... z.B.: Kamera soll immer auf Mittelpunkt eines bestimmten Objekts gerichtet sein
- komplexere: on-top, in-plane, on-path etc.
- u.U. nur iterativ / approximativ lösbar, oder gar nicht lösbar (Widersprüche)

Spezialfall:
mechanische Constraints

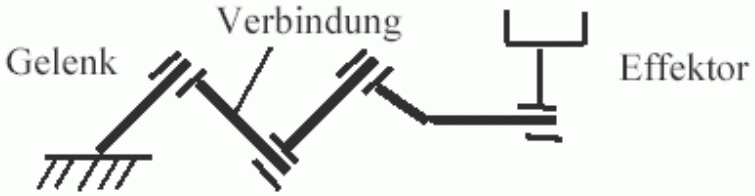
- Kollisionserkennung
- kinematische Modelle (Verbindung von Objekten an Gelenken)

Kinematik:
motiviert durch Animation von gegliederten Strukturen (*articulated structures*)



- Anwendung bei gelenkartig verbundenen Objekten, z.B. Tiermodellen, Menschmodellen
- Geometrisches Modell wird um “Skelett” erweitert: idealisiert eine

kinematische Kette



Vorwärtsrechnung

- Eingabe: Gelenkstellungen
- Ausgabe: Effektorstellung
- Berechnung der Endgliedstellung bei vorgegebenen Gelenkeinstellungen

Kinematik

Rückwärtsrechnung:

- Eingabe: Effektorstellung
- Ausgabe: Gelenkstellungen
- Berechnung der Gelenkstellungen bei vorgegebener Effektorstellung

Inverse Kinematik

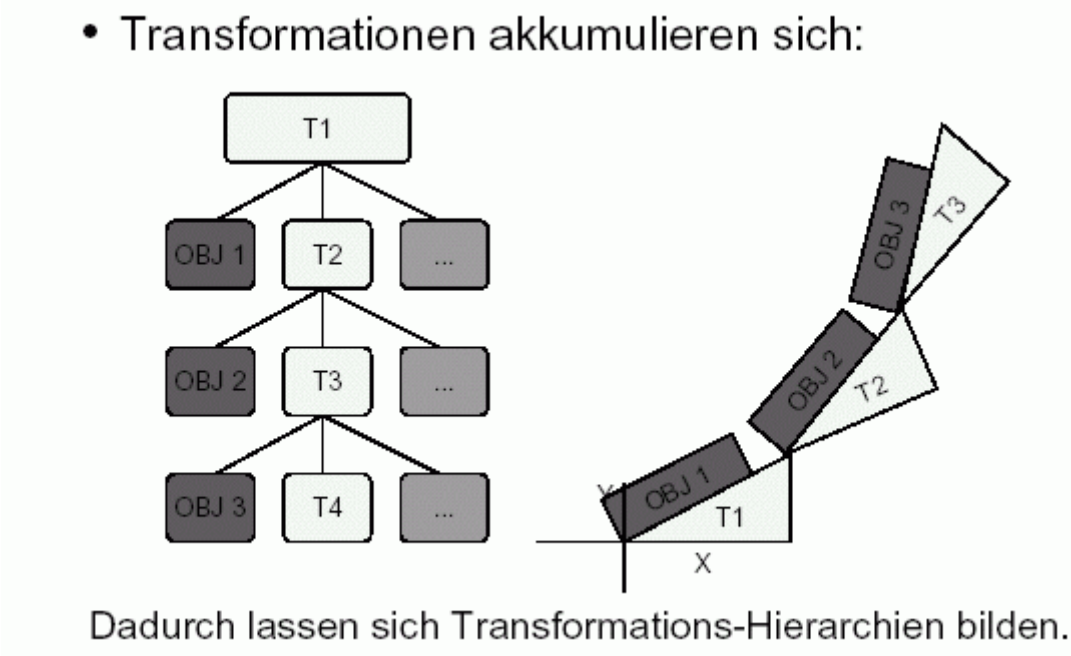
(Anwendung auch in der Robotik)

Vereinfachung durch Standardisierung nach Denavit / Hartenberg:

- es werden nur Schub- und Drehgelenke behandelt
- Beschreibung eines Gelenks durch konstante, konstruktionsbedingte Werte und durch Gelenkvariablen (aktuelle Stellung des Gelenks)

aus solchen Primitivgelenken lassen sich beliebige kinematische Ketten bilden

Vorwärtsrechnung: Produkt von Matrizen
Grundlage: Transformationen akkumulieren sich (vgl. Szenengraph!)



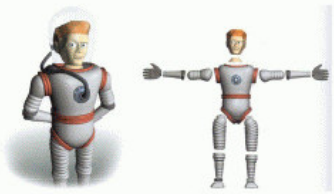
- mit diesem Ansatz mathematische Beschreibung größerer kinematischer Ketten möglich
- in der Praxis häufig gekoppelt mit "Skin models"
- "interessante" Ketten sind häufig unterbestimmt – weitere Constraints müssen eingeführt werden, um eindeutige Lösung zu gewährleisten

- Bewegungen trotzdem oft "unnatürlich"
- Problem, dass zunächst nur "Skelette" animiert werden: Übertragung auf 3D-Körper nichttrivial

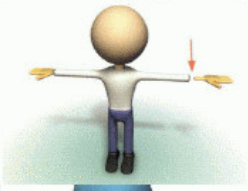
"Skinning":
bei skelettierten Objekten müssen geeignete Maßnahmen ergriffen werden, um die "Haut" der Bewegung anzupassen.

Methoden für segmentierte Objekte

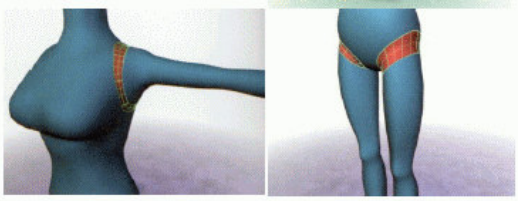
- „versteckte“ Kugelgelenke (filler objects)



- Kleidung



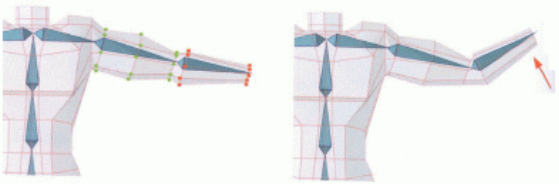
- Blend Objects

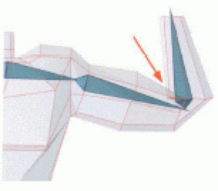


Skin surfaces:

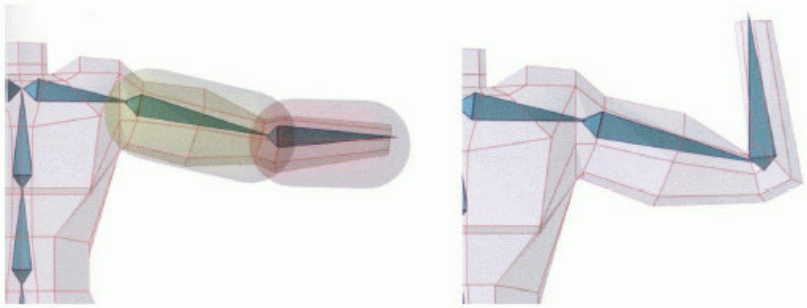
Die definierenden Eckpunkte sind relativ zu den Gelenken (Verbindungen) definiert:

- Direkte Zuordnung (direct assignment)





- Gewichtete Zuordnung (weighted assignment z.B. mit envelopes)



Kinematik vs. Dynamik

Kinematik:

- Bewegungslehre ohne Berücksichtigung der auftretenden Kräfte
- betrachtet Position, Geschwindigkeit, Beschleunigung...

Dynamik:

- Bewegungslehre unter Berücksichtigung von Kräften
- betrachtet Massen, Trägheitsmomente, Kräfte, Impuls, Energie...

Animationen mit Dynamik-Simulation:

- z.B. Feder-Masse-Dämpfer-Systeme
- häufig auf Felddynamik erweitert: Gravitation, Ladung...
- Beeinflussung vieler Objekte gleichzeitig: Partikel
- Spezialfall der allgemeinen Simulation
- oft schwierig zu benutzen, da visuelle Effekte schwer vorhersehbar!

Simulation

- basiert auf Repräsentation von Aspekten der realen Welt in einem abstrakten Modell
- Kinematik und Dynamik sind nur Spezialfälle, weitere Einsatzgebiete z.B.: Strömungslehre, Thermodynamik, Radiometrie (Radiosity), Elektrodynamik, biologisches Wachstum, chemische Reaktionen...
- potenziell sehr viele verschiedene Basis-Modelle, Sprachen und Methoden verfügbar, z.B. Petrinetze, Simula, HLA...
- methodisch oft eher Experimentieren als Konstruieren oder Gestalten!

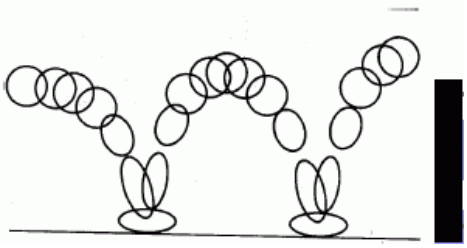
- sehr leistungsfähige Simulatoren vorhanden
- Simulation erlaubt Kontrolle von Animation auf höherem (abstrakterem) Niveau als Geometrie und visuelle Merkmale
- Grundlage der "interaktiven" Animation
- Verknüpfung mit anderen Animationsmethoden möglich, aber selten implementiert

Erweiterungen von Dynamik in der Animation:

Soft-object-Animation

Motivation und Bedeutung

- Generell: weitere Freiheiten für den Animator
- Ein wichtiges Prinzip aus der traditionellen (Walt Disney-) Animation:
  - **Squasch and Stretch**
  - **Stretch and Squeeze**



⇒ die traditionellen Grenzen zwischen Modellierung und Animation verwischen!

bei der Soft-object-Animation sind immer 2 Prozesse beteiligt:

- Methode zur Objektdeformation (shape change) (polygonale Modelle: Verschieben der Eckpunkte; parametrische Modelle: Verändern der Kontrollpunkte)
- Methode zur Animation des Deformationsprozesses

Deformation polygonaler Repräsentationen:

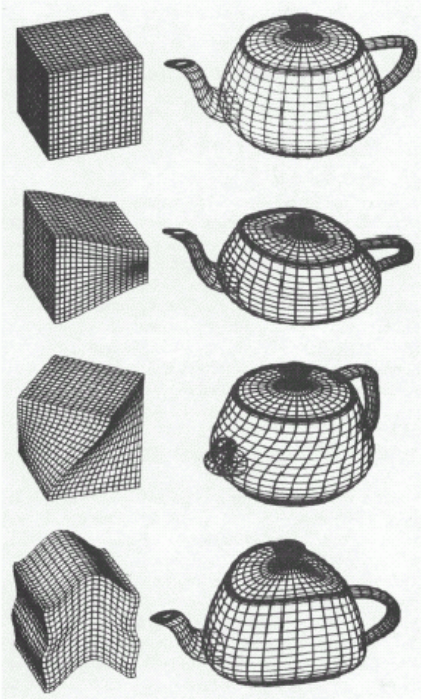
nur die Positionen der Eckpunkte werden verändert, Topologie bleibt erhalten

Deformation parametrischer Repräsentationen (z.B. bikubische Flächen)

Nichtlineare globale Deformation ("3D-Warping")
nach Barr 1984

- Beschreibung mit Hilfe von Transformationsmatrizen
- 3 Haupttransformationen: Tapering (verjüngen, anspitzen), Twisting (verdrehen), Bending (biegen, krümmen).

Nichtlineare globale Deformation Tapering



$(X,Y,Z) = F(x,y,z)$

(x,y,z) : Eckpunkt im nichtdeformierten Objekt

(X,Y,Z) : Eckpunkt im deformierten Objekt

Beispiel : Skalierung

Nichtdeformiert : $(X,Y,Z) = (s_x x, s_y y, s_z z)$

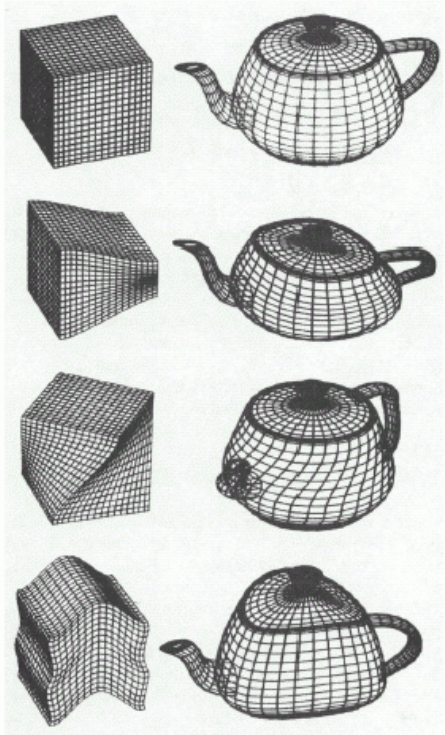
Tapering entlang der z - Achse :

$(X,Y,Z) = (rx, ry, z)$ mit

$r = f(z)$ als lineares oder

nichtlineares Tapering - Profil

Nichtlineare globale Deformation Twisting



basiert auf der Rotation,

z.B. um die z - Achse

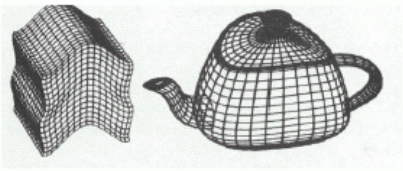
$$(X,Y,Z) = (x\cos\Theta - y\sin\Theta,\; x\sin\Theta + y\cos\Theta, z)$$

$$\Theta = f(z)$$

$f(z)$ repräsentiert die Verdrehungs - rate pro z - Einheit

Nichtlineare globale Deformation
Bending

Ein Bending ist definiert als eine zusammengesetzte Transformation (Rotation und Translation) in einer Region: z.B. entlang der y-Achse:



$y_{min} \le y \le y_{max}$

Krümmung : k^{-1}

Zentrum der Krümmung : y_0

Verdrehungswinkel : $\Theta = k(y'-y_0)$ mit

$$y' = \begin{cases} y_{min} & \text{für } y \le y_{min} \\ y & \text{für } y_{min} < y < y_{max} \\ y_{max} & \text{für } y \ge y_{max} \end{cases}$$

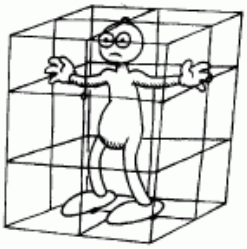
Die Deformationstransformation ist dann

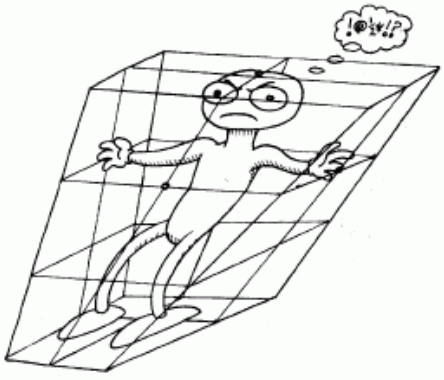
$X = x$

$$Y = \begin{cases} -\sin\Theta(z-k^{-1}) + y_0 & y_{min} \le y \le y_{max} \\ -\sin\Theta(z-k^{-1}) + y_0 + \cos\Theta(y-y_{min}) & y < y_{min} \\ -\sin\Theta(z-k^{-1}) + y_0 + \cos\Theta(y-y_{max}) & y > y_{max} \end{cases}$$

$$Z = \begin{cases} \cos\Theta(z-k^{-1}) + k^{-1} & y_{min} \le y \le y_{max} \\ \cos\Theta(z-k^{-1}) + k^{-1} + \sin\Theta(y-y_{min}) & y < y_{min} \\ \cos\Theta(z-k^{-1}) + k^{-1} + \sin\Theta(y-y_{max}) & y > y_{max} \end{cases}$$

Freiform-Deformation (Gitter-Deformation) nach [Sederburg 86]





Idee: 1. Man faßt ein Objekt in ein umgebendes Gitter (einen Raum) ein.

2. Man deformiert das Gitter (den Raum) durch Modeling-Transformationen

3. Deformationen des Gitters (des Raumes) werden auf das Objekt übertragen.

Modellvorstellung z.B. : Objekte sind durch Federn in dem Gitter gehalten

benutzbar für polygonale und parametrische Objekte:
Eckpunkte ←→ Kontrollpunkte

(vgl. 2D-Morphing)

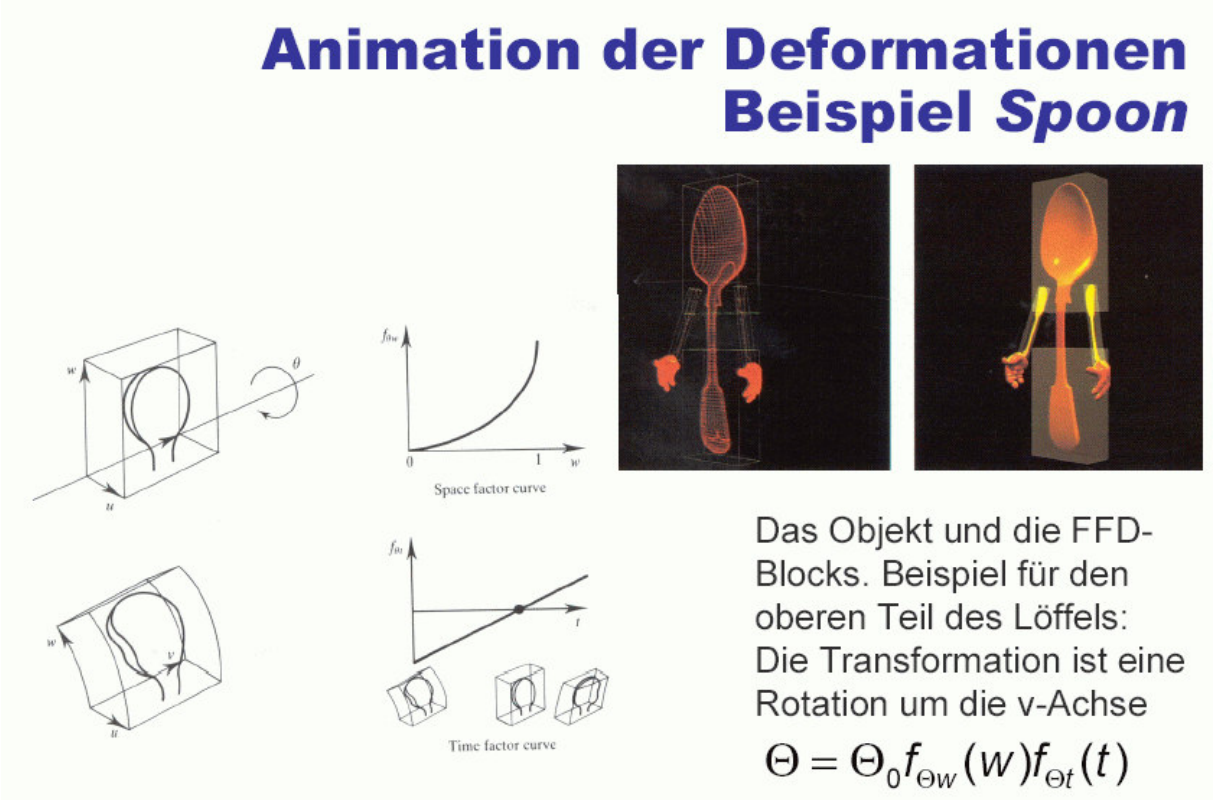
Animation der Deformationen:

Transformationen sind Funktionen des Ortes (sog. Faktorkurven) – diese werden um eine zeitliche Komponente erweitert

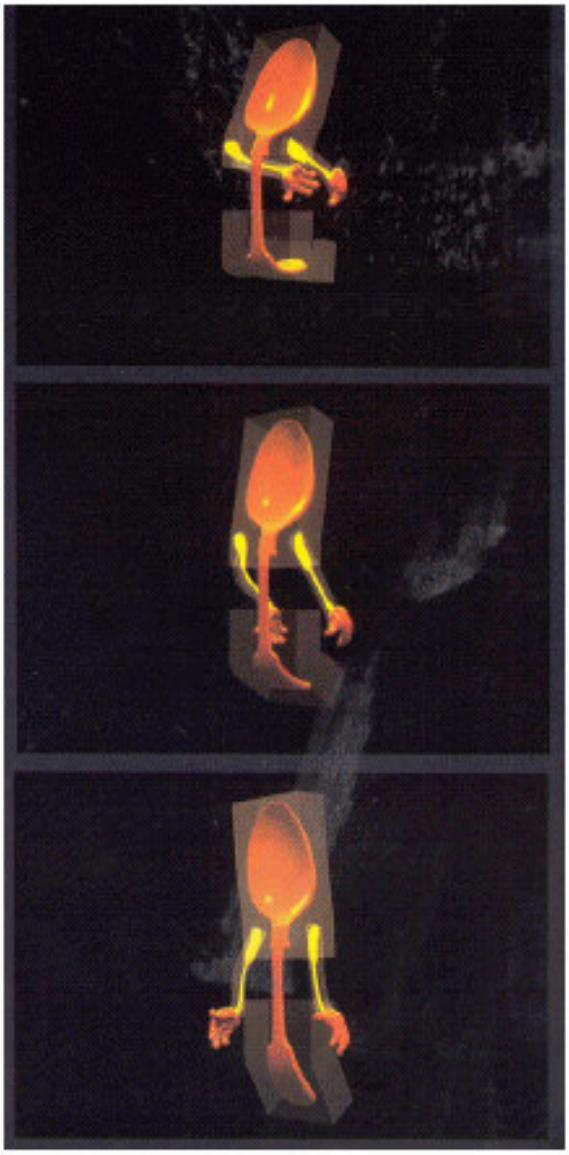
Der Benutzer zerlegt die Deformation in 2 Komponenten:
einen Satz von Transformationen, die den Gesamtumfang der Deformation beschreiben, zusammen mit den entsprechenden Parametern,
einen Satz von Faktorkurven in Raum und Zeit, die die Veränderungen der Parameter (wo und wann) beschreiben

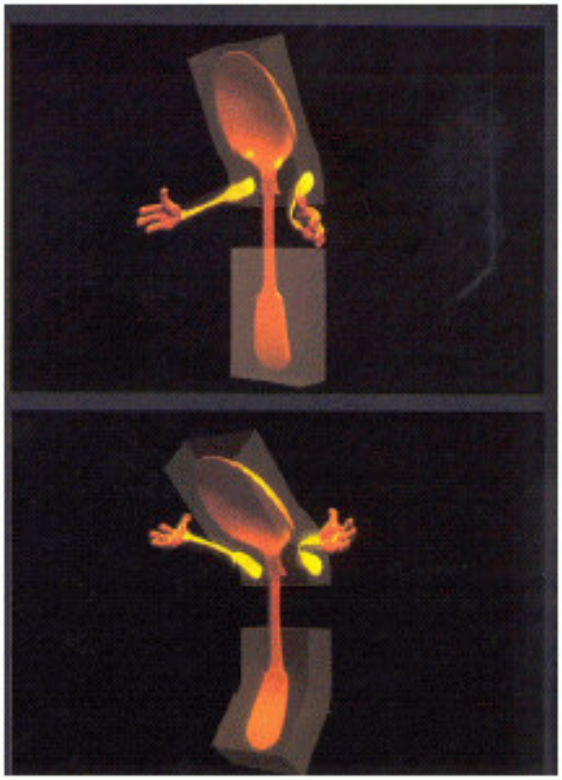
Faktorkurven: oft als Bézierkurven spezifiziert (vgl. Kinegraph)

Beispiel:



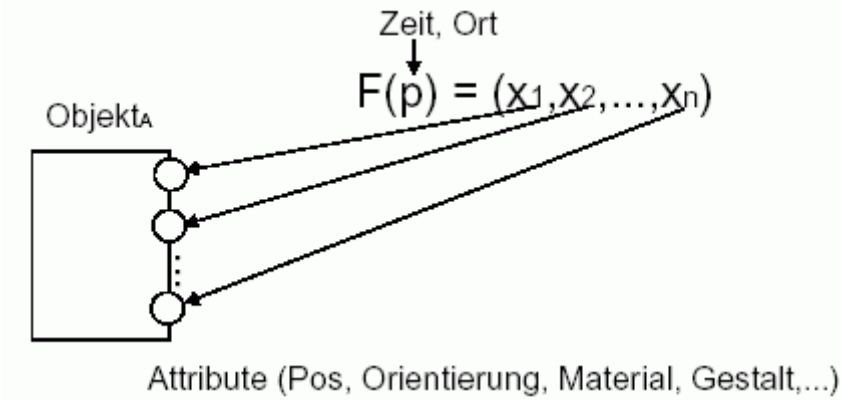
(FFD = Freiformdeformation, 3D-Warping)





Prozedurale Animation

Idee: Modifizieren der Objekte (und Erzeugen, Löschen) mittels algorithmischer Vorschriften
(vgl. prozedurale Modellierung – Fraktale)



Art des verwendeten Algorithmus: sehr unterschiedliche Ansätze

- stochastische Animation (Partikelsysteme)
- verhaltensbasierte Animation (Fische, Vögel, Insekten)
- wissensbasierte Ansätze, z.B. genetische Algorithmen

typisch: emergentes globales Verhalten durch lokale Regeln

Anwendungen z.B.:

Partikelsysteme
: Pflanzen, Feuer, Wassertropfen

verhaltensbasierte Ansätze
: Schwärme, agentenorientierte Modelle, Artificial Life

Modellierung von Strukturen
: Textilien, elastische Oberflächen

Partikelsysteme

Partikel: größere Anzahl an betrachteten Einheiten mit Attributen, die Aussehen festlegen
oft grafische Primitive wie Punkte, Linien, einfache 3D-Körper

- stochastische Komponente: Kontrolleinheit, die die Partikel beeinflusst
- nichtdeterministische Modifikation der Attribute (z.B. Farbe, Geschwindigkeit), meist innerhalb vordefinierter Grenzen
- Verwendung zuerst von Reeves in "Star Trek II" zur Animation der Entwicklung eines Planeten

5 Schritte für die Berechnung eines Frames:
(1) erzeuge neue Partikel im System
(2) Zuordnung von initialen Attributwerten für jedes neue Partikel
(3) jedes Partikel, das seine maximale Lebensdauer erreicht hat, wird gelöscht
(4) aktuelle Partikel werden gemäß der Berechnungsvorschrift bewegt
(5) Alle Partikel innerhalb vordefinierter Grenzen werden gerendert

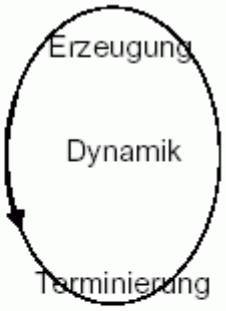
```
while (time < duration)
    {
    generate_particles
    update_particles
    remove_dead_particles
    draw_particles
    time++
    }
```

Vorteile des Ansatzes:

- komplexe Objekte mit nicht eindeutig definierbaren Konturen und komplexer Bewegung können mit wenig Aufwand modelliert werden
- Algorithmen können mit anderen Systemen gekoppelt werden, z.B. Simulationen
- Umfang der Parametrisierung erlaubt Kontrolle über den Aufwand in Relation zur Wirksamkeit

Attribute eines Partikel:

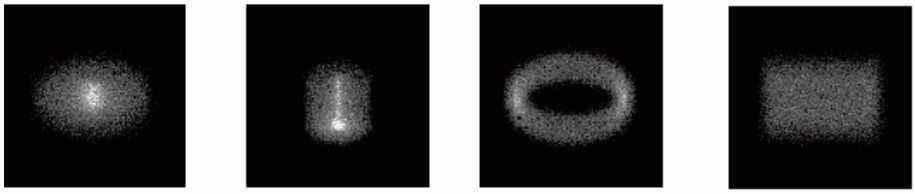
Position
Velocity (speed and direction)
Color
Lifetime
Age
Shape
Size
Transparency



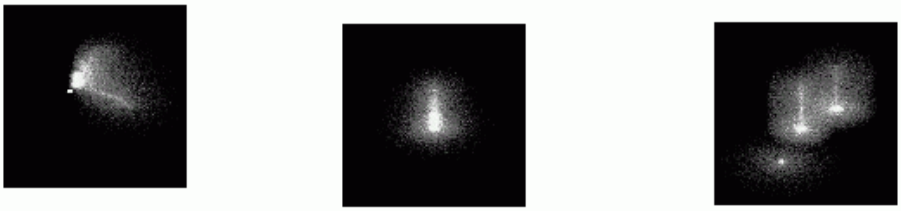
- Darstellung als Punkt, Linie, Fläche, 3D-Objekt
- Partikel kollidieren nicht mit anderen Partikeln
- Partikel werfen keine Schatten
- Partikel reflektieren kein Licht, sondern sind als Punktlichtquellen modelliert

Erzeugung der Partikel:

Ort der Erzeugung (Generation shape) kann variieren



Initialwerte der Partikel können variieren
DefaultValue(i,t)=MeanValue(i,t)+Rand()*Var(i,t) mit t als Zeit und i=(Position, Orientierung, Größe, Farbe, Transparenz, Geschwindigkeit,...)



Partikel werden durch einen globalen, kontrollierten stochastischen Prozeß erzeugt,
Zeitabhängigkeit und Varianz können Ausdehnung der Wolke steuern

$$N(t) = averageN(t) + rand(r) * \text{variance}(t)$$

2. Methode: Bildschirmgröße als Parameter verwenden

$$N(t) = (averageN(t) + rand(r) * \text{variance}(t)) * \text{screenSize}$$

Dynamik:

- lokale Skripte können die Partikelattribute modifizieren
- kinematische oder dynamische Gesetze können auf diese Weise eingebaut werden

z.B. Explosion: von Kern aus nach oben, dann aufgrund der Schwerkraft nach unten, dabei wechselt die Farbe

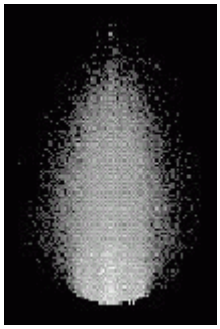
Termination von Partikeln:
Lebensalter und maximale Lebensdauer als zentrale Attribute
Partikel werden gelöscht, wenn

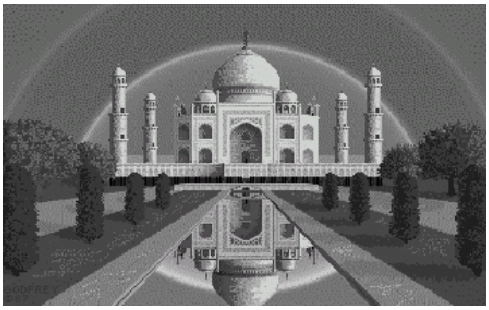
- ihre max. Lebensdauer erreicht ist
- sie sich aus dem relevanten Bereich entfernt haben und nicht zurückkehren können
- Attributwerte bestimmte Schwellenwerte überschritten haben (z.B. Farbwert nahe Schwarz oder transparent, Partikel zu klein etc.)
- besondere Ereignisse eintreffen (z.B. Kollision mit Szenenobjekt)

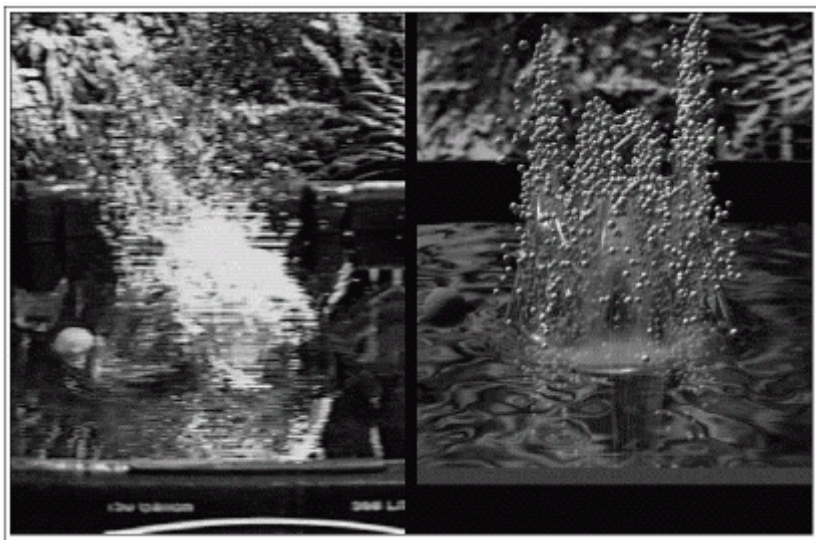
Rendern von Partikeln:
unterschiedliche Ansätze

- geometriebasiert: kleine polygonale 3D-Modelle für individuelle Partikel (aufwändig!)
- bildbasiert: Partikel werden direkt als Pixel dargestellt, Farbwerte werden addiert (schnelle Berechnung der Partikel, separates Rendering, Komposition mit restlicher Szene)

Beispiele:







Vergleich von Realität (links) und Simulation mit Partikelsystem (rechts), nach O'Brien et al.: Balls in a pond