Simulation von Festkörpern mit orientierten Partikeln

basierend auf dem Paper „Solid Simulation with Oriented Particles” von Matthias Müller & Nuttapong Chentanez (2012)

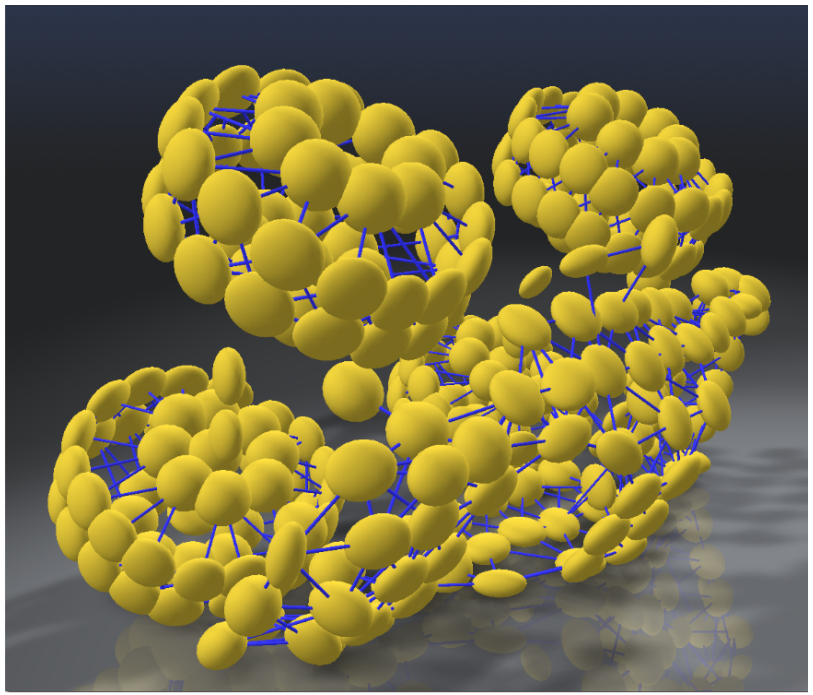
Michael Wolff, Institut für Informatik, Georg-August-Universitat Göttingen

28. Januar 2014

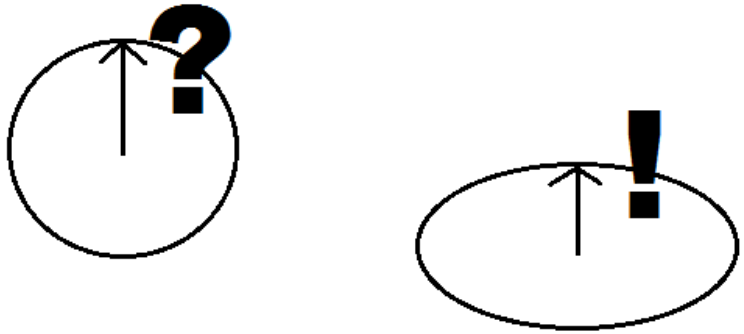
Inhaltsverzeichnis

Einführung

Worum geht's?



Einführung

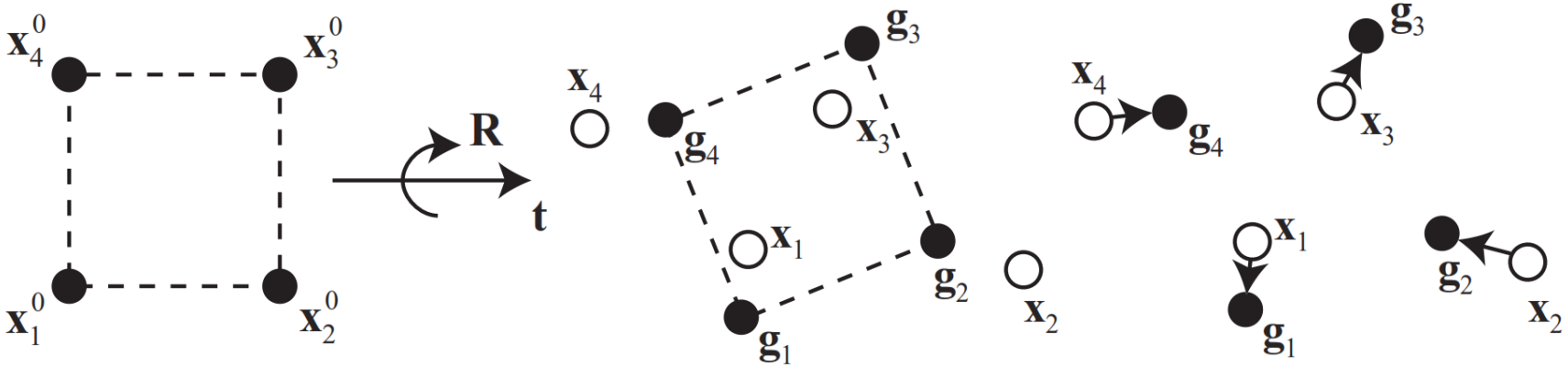


Was bringt die zusätzliche Information?

1. Partikel mit anisotropischen Formen approximieren Oberflächen besser als Sphären
2. Shape Matching (SM) auch für dünne Strukturen wie Ketten robust
3. Transformationen, die die Partikel speichern können für robustes Skinning und zur Rücktransformation in den Ruhezustand verwendet werden

Verallgemeinertes Shape Matching

Was ist Shape Matching?



1. Passe Originalform $\mathbf{x}_i^0$ an deformierte Form $\mathbf{x}_i$ an
2. Verschiebe deformierte Punkte $\mathbf{x}_i$ in Richtung der angepassten Form $\mathbf{g}_i$
   ($\rightarrow$ goal positions)

Verallgemeinertes Shape Matching

Gegeben n Partikel mit ...

- Ruheposition $\overline{\mathbf{x}}_\mathrm{i}$
- Aktueller Position $\mathbf{x}_\mathrm{i}$
- Masse m_i

... suche nach ...

- Globaler Translation $\mathbf{t}$
- Rotation $\mathbf{R}$

... welche sich optimal an die aktuellen Positionen annähern

Verallgemeinertes Shape Matching

Berechne dafür Matrix $\mathbf{A}$ der Form:

$$\mathbf{A} = \sum_i m_i(\mathbf{x}_i - \mathbf{c})(\overline{\mathbf{x}}_i - \overline{\mathbf{c}})^T \in \mathbb{R}^{3\times 3} \quad (1)$$

... mit den Massezentren:

$$\mathbf{c} = \sum_i m_i \mathbf{x}_i / \sum_i m_i$$

$$\overline{\mathbf{c}} = \sum_i m_i \overline{\mathbf{x}}_i / \sum_i m_i$$

$\overline{\mathbf{x}}_\mathrm{i} \rightarrow$ Ruheposition
$\mathbf{x}_\mathrm{i} \rightarrow$ Akt. Position
$m_\mathrm{i} \rightarrow$ Masse
$\mathbf{t} \rightarrow$ Translation
$\mathbf{R} \rightarrow$ Rotation

Verallgemeinertes Shape Matching

Die Polarzerlegung $\mathbf{A} = \mathbf{RS}$ ergibt die optimale Rotation ...
... und für die Translation gilt $\mathbf{t} = \mathbf{c} - \overline{\mathbf{c}}$ ...
... woraus sich die Zielposition für jeden Partikel berechnet:

$$\mathbf{g}_i = \mathbf{R}(\overline{\mathbf{x}}_i - \overline{\mathbf{c}}) + \mathbf{c}$$

Problem: Wenn Partikel nahezu koplanar/kolinaer sind, so ist $\mathbf{A}$ schlecht konditioniert oder singulär
$\rightarrow \mathbf{R}$ ist **nicht** wohldefiniert!
$\rightarrow$ Nutze Orientierungsinformationen!

$\overline{\mathbf{x}}_\mathrm{i} \rightarrow$ Ruheposition
$\mathbf{x}_\mathrm{i} \rightarrow$ Akt. Position
$m_\mathrm{i} \rightarrow$ Masse
$\mathbf{c} \rightarrow$ Massezentrum
$\mathbf{t} \rightarrow$ Translation
$\mathbf{R} \rightarrow$ Rotation

Verallgemeinertes Shape Matching

Annahme: Zwei Gruppen von Partikeln mit Matrizen $\mathbf{A}_1$, $\mathbf{A}_2$

Für Vereinigung der Gruppen reformuliere (1) zu:

$$\mathbf{A} = \sum_i m_i \mathbf{x}_i \overline{\mathbf{x}}_i^T - M \mathbf{c} \overline{\mathbf{c}}^T \qquad (2)$$

Definiere für einen einzelnen Partikel Matrix $\mathbf{A}_\mathrm{i}$

Nutze (2), um das Massezentrum jedes $\mathbf{A}_\mathrm{i}$ zu einem Globalen zu verschieben ...

$$\mathbf{A} = \sum_i (\mathbf{A}_i + m_i \mathbf{x}_i \overline{\mathbf{x}}_i^T - m_i \mathbf{c} \overline{\mathbf{c}}^T)$$

$\overline{\mathbf{x}}_\mathrm{i} \rightarrow$ Ruheposition
$\mathbf{x}_\mathrm{i} \rightarrow$ Akt. Position
$m_\mathrm{i} \rightarrow$ Masse
$M = \sum_i m_i$
$\mathbf{c} \rightarrow$ Massezentrum
$\mathbf{R} \rightarrow$ Rotation

Verallgemeinertes Shape Matching

... und (1) verallgemeinert zu:

$$\mathbf{A} = \sum_i (\mathbf{A}_i + m_i \mathbf{x}_i \overline{\mathbf{x}}_i^T) - M \mathbf{c} \overline{\mathbf{c}}^T$$

$\overline{\mathbf{x}}_\mathrm{i} \rightarrow$ Ruheposition
$\mathbf{x}_\mathrm{i} \rightarrow$ Akt. Position
$m_\mathrm{i} \rightarrow$ Masse
$M = \sum_i m_i$
$\mathbf{c} \rightarrow$ Massezentrum
$\mathbf{R} \rightarrow$ Rotation

Verallgemeinertes Shape Matching

Zur Berechnung der Matrix eines Partikels mit orthonormaler Orientierungsmatrix $\mathbf{R}$ integriere (1) über das Volumen des Partikels:

$$\mathbf{A}_{\text{sphere}} = \ldots = \frac{1}{5} m r^2 \mathbf{R}$$

mit Radius r

$$\mathbf{A}_{\text{ellipsoid}} = \frac{1}{5} m \begin{bmatrix} a^2 & 0 & 0 \\ 0 & b^2 & 0 \\ 0 & 0 & c^2 \end{bmatrix} \mathbf{R}$$

mit Radien a, b, c

$\overline{\mathbf{x}}_i \rightarrow$ Ruheposition
$\mathbf{x}_i \rightarrow$ Akt. Position
$m_i \rightarrow$ Masse
$M = \sum_i m_i$
$\mathbf{c} \rightarrow$ Massezentrum
$\mathbf{R} \rightarrow$ Rotation

Verallgemeinerte positionsbasierte Dynamiken

Worum geht's?
→ Die zeitliche Entwicklung der Partikel!

Grundlegender Ablauf von PBD:

1. **Prognosephase**: Berechne voraussichtliche Position $\mathbf{x}_\mathrm{p}$ für jeden Partikel mittels explizitem Euler-Verfahren

$$\mathbf{x}_p \leftarrow \mathbf{x} + \mathbf{v}\Delta t$$

2. **Korrekturphase**: Der *Solver* korrigiert vorausgesagte Positionen, sodass sie eine Anzahl Bedingungen erfüllen
3. **Integrationsphase**: Aktualisiere Zustandsvariablen

$$\mathbf{v} \leftarrow (\mathbf{x}_p - \mathbf{x})/\Delta t$$

$$\mathbf{x} \leftarrow \mathbf{x}_p$$

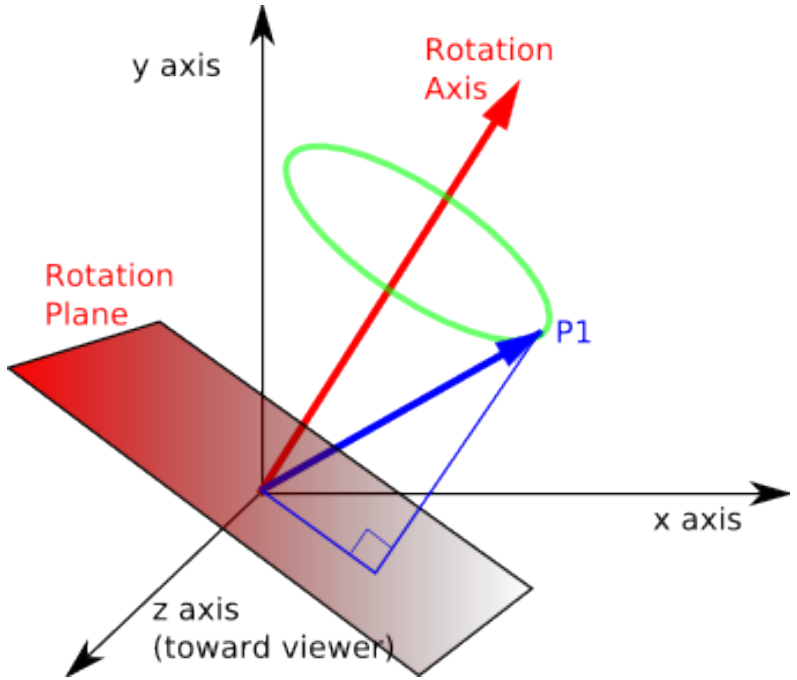
$\mathbf{x} \rightarrow$ Akt. Position
$\mathbf{x}_\mathrm{p} \rightarrow$ Prog. Posi.
$\mathbf{v} \rightarrow$ Geschwindigkeit

→ 3 Phasen/Zeitschritt

Integration

Partikel besitzen zusätzlich zu Position und Geschwindigkeit noch ...

- ein **Orientierungsquaternion** $\mathbf{q}$
- eine **Rotationsgeschwindigkeit** ω



Integration

→ Erweiterung der **Prognosephase**:

$$\mathbf{x}_p \leftarrow \mathbf{x} + \mathbf{v}\Delta t$$

$$\boxed{\mathbf{q}_p \leftarrow \left[\frac{\omega}{|\omega|}\sin(\frac{|\omega|\Delta t}{2}), \cos(\frac{|\omega|\Delta t}{2})\right]\mathbf{q}}$$

mit

$$\mathbf{q}_p \leftarrow \mathbf{q} \text{ für } |\omega| < \epsilon$$

$\mathbf{x} \rightarrow$ Akt. Position
$\mathbf{x}_p \rightarrow$ Prog. Posi.
$\mathbf{v} \rightarrow$ Geschwindigkeit
$\mathbf{q} \rightarrow$ Orient. Quat.
$\omega \rightarrow$ Rot.-Geschwin.

Integration

→ Erweiterung der **Integrationsphase**:

$$\mathbf{v} \leftarrow (\mathbf{x}_p - \mathbf{x})/\Delta t$$

$$\mathbf{x} \leftarrow \mathbf{x}_p$$

$$\boxed{\omega \leftarrow \text{axis}(\mathbf{q}_p\mathbf{q}^{-1}) \cdot \text{angle}(\mathbf{q}_p\mathbf{q}^{-1})/\Delta t}$$

$$\boxed{\mathbf{q} \leftarrow \mathbf{q}_p}$$

mit axis() → normierte Richtung eines Quaternion

angle() → Winkel eines Quaternion

$\mathbf{x} \rightarrow$ Akt. Position
$\mathbf{x}_p \rightarrow$ Prog. Posi.
$\mathbf{v} \rightarrow$ Geschwindigkeit
$\mathbf{q} \rightarrow$ Orient. Quat.
$\omega \rightarrow$ Rot.-Geschwin.

Reibung

Reibung? $\rightarrow$ Skaliere lineare Geschwindigkeit um konstanten Faktor s nach unten!

Bei Kollision modifiziere lineare und Rotationsgeschwindigkeit:

$$\mathbf{v} \leftarrow \mathbf{v} + (\mathbf{v}_s - \mathbf{v})\perp_{\mathbf{n}} \cdot s_{lin}$$

$$\omega \leftarrow \omega + \frac{\mathbf{r}}{|\mathbf{r}|^2} \times (\mathbf{v}_s - \mathbf{v} - \omega \times \mathbf{r}) \cdot s_{rot}$$

mit $\mathbf{v}_s \rightarrow$ Geschwindigkeit und

$\mathbf{n} \rightarrow$ Normale des Objekts am Punkt des Auftreffens

$$s_{lin} \in [0...1], s_{rot} \in [0...1]$$

$\mathbf{v} \rightarrow$ Geschwindigkeit
$\omega \rightarrow$ Rot.-Geschwin.
$\mathbf{r} = r\mathbf{n}$
$r \rightarrow$ Radius
$s \rightarrow$ Reibungsfaktor

Reibung

Ganz ähnlich bei Kollision von zwei Partikeln:

$$\mathbf{v}_1 \leftarrow \mathbf{v}_1 + \left(\frac{\mathbf{v}_1 + \mathbf{v}_2}{2} - \mathbf{v}_1\right)\perp_{\mathbf{n}} \cdot s_{lin}$$

$$\mathbf{v}_2 \leftarrow \mathbf{v}_2 + \left(\frac{\mathbf{v}_1 + \mathbf{v}_2}{2} - \mathbf{v}_2\right)\perp_{\mathbf{n}} \cdot s_{lin}$$

$$\omega_1 \leftarrow \omega_1 + \frac{\mathbf{r}_1}{|\mathbf{r}_1|^2} \times \left(\mathbf{v}_{avg} - \mathbf{v}_1 - \omega_1 \times \mathbf{r}_1\right) \cdot s_{rot}$$

$$\omega_2 \leftarrow \omega_2 + \frac{\mathbf{r}_2}{|\mathbf{r}_2|^2} \times \left(\mathbf{v}_{avg} - \mathbf{v}_2 - \omega_2 \times \mathbf{r}_2\right) \cdot s_{rot}$$

$$\text{mit } \mathbf{n} = (\mathbf{x}_2 - \mathbf{x}_1)/|\mathbf{x}_2 - \mathbf{x}_1|$$

$$\mathbf{r}_1 = r\mathbf{n}, \mathbf{r}_2 = -r\mathbf{n}$$

$\mathbf{v} \rightarrow$ Geschwindigkeit
$\omega \rightarrow$ Rot.-Geschwin.
$r \rightarrow$ Radius
$s \rightarrow$ Reibungsfaktor

Implizites Shape Matching

- Definiert eine Shape Matching Gruppe pro Partikel
- Gruppen beinhalten:
  - Korrespondierenden Partikel
  - Alle Partikel, die über einzelne Kante verbunden sind
- Nach Prognosephase iteriert *Solver* über alle SM-Bedingungen
- Berechne Zielpositionen
- Bewege alle Partikel der Gruppe Richtung ihrer Zielposition um einen Anteil in Abhängigkeit von $s_{\text{stiffness}}$
  ($\rightarrow$ Starrheit/Festigkeit, pro Partikel spezifizierbar)
- Aktualisiere Orientierung des Hauptpartikels

Streckung und Biegung

- SM modelliert Streckung & Biegung zugleich
- Auch separat spezifizierbar über Entfernungsbedingungen von PBD
- Bsp.: Zur Erhöhung der Biegsamkeit verringere Starrheit und aktiviere Entfernungsbedingung

Explizites Shape Matching

- Nutzer kann explizite SM-Gruppen spezifizieren
  $\rightarrow$ Kann beliebige Untermengen an Partikel umfassen
- Implizites SM nicht für explizite Gruppen
- **Alle** teilnehmenden Partikel erfahren gleiche SM-Rotation
  ($\rightarrow$ Starre Komponenten)
- Partikel in mehr als einer expliziten Gruppe gelten als nicht-orientiert
  ($\rightarrow$ Ermöglicht Gelenke)

Plastische Deformation

- Explizite SM-Gruppen auch deformierbar
- Deformation, sobald einer der Gruppenpartikel an Kollision mit bestimmter Geschwindigkeit beteiligt
  $\rightarrow$ Deaktiviere kurzzeitig explizite Gruppe
  $\rightarrow$ Implizites SM

Torsionswiderstand

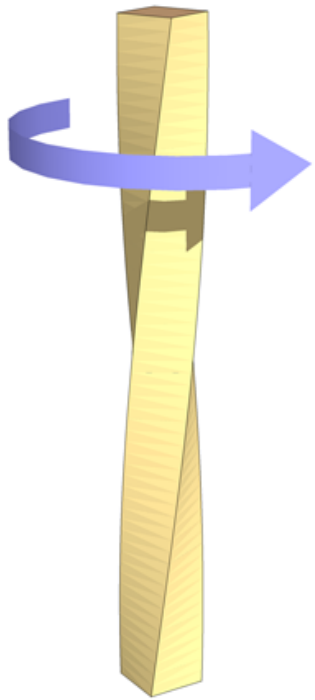
$\rightarrow$ Iteriere über alle Kanten und aktualisiere Rotationen benachbarter Partikel:

$$\mathbf{q}_1 \leftarrow \text{slerp}(\mathbf{q}_1, \mathbf{q}_2, \frac{1}{2} s_{torsion})$$

$$\mathbf{q}_2 \leftarrow \text{slerp}(\mathbf{q}_2, \mathbf{q}_1, \frac{1}{2} s_{torsion})$$

und $\text{slerp}(\mathbf{q}_1, \mathbf{q}_2, s)$ liefert $\mathbf{q}_1$ wenn $s = 0, \mathbf{q}_2$ wenn $s = 1$

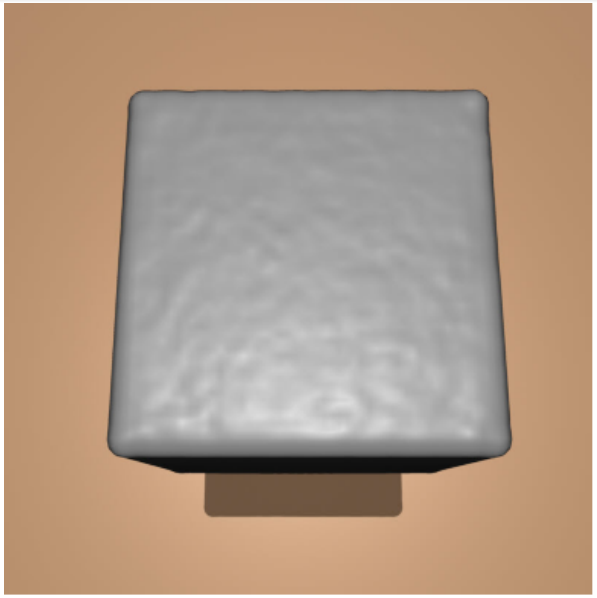
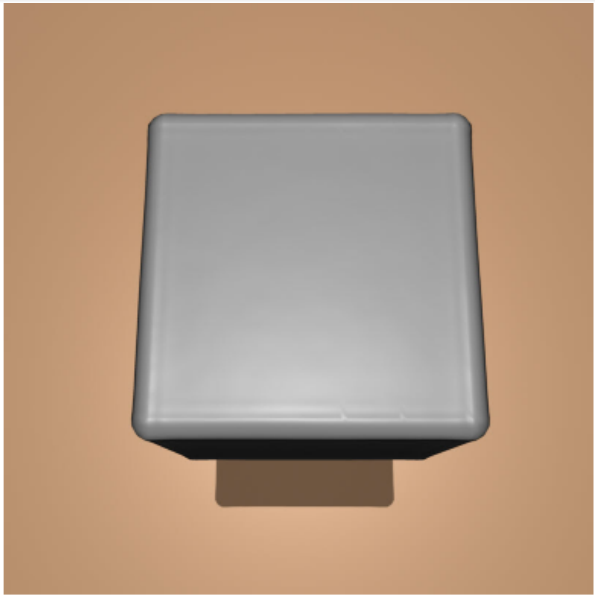
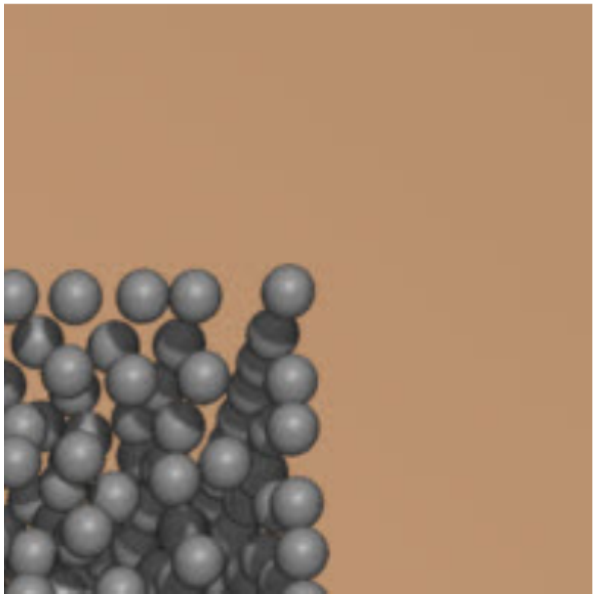
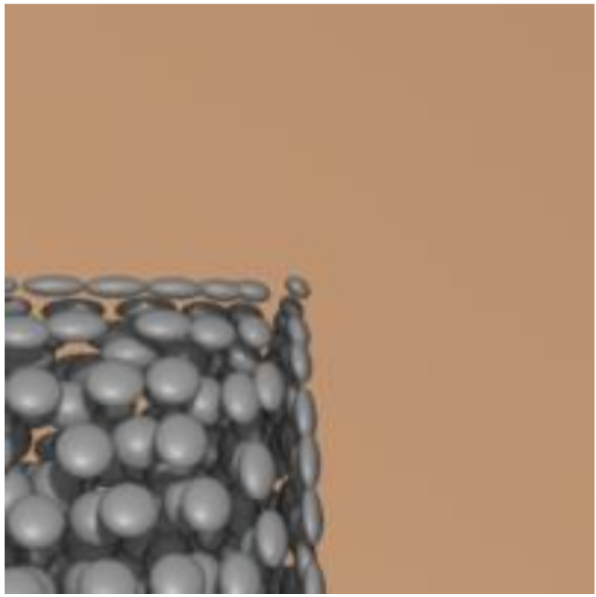
sonst sphärische Interpolation der beiden



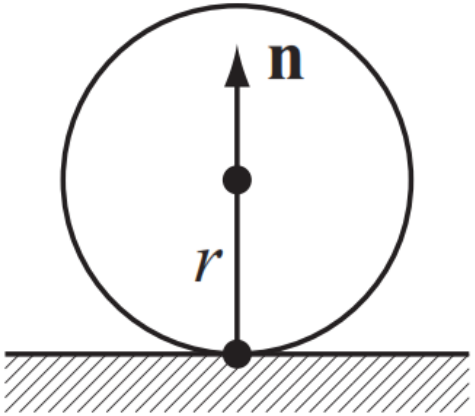
Kollisionsbewältigung

Sphären
vs.
Ellipsoide

Unebene Fläche
$\rightarrow$ falsche Reibung
$\rightarrow$ falsche Kollision

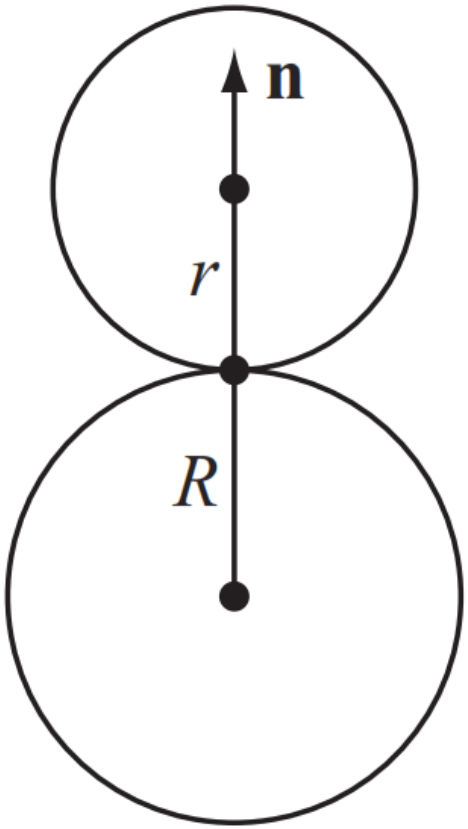





Ellipsoidkollision - Sphäre

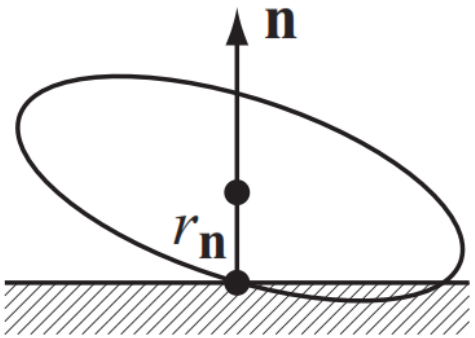


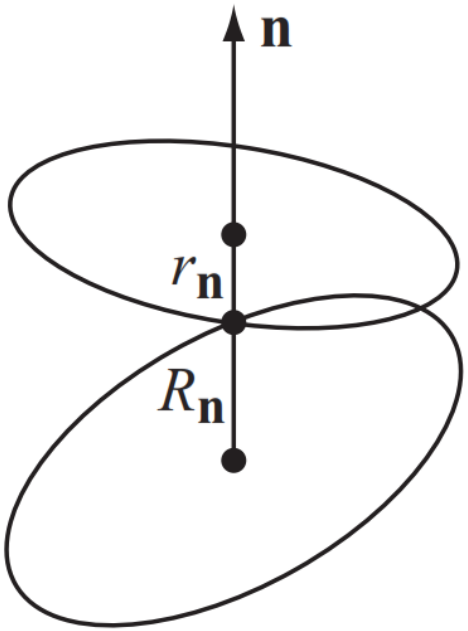
Kollision, wenn Partikel näher an Ebene/Sphäre als sein Radius r

$\rightarrow$ Verschiebung des Partikels entlang der Ebenen/Sphärennormale $\mathbf{n}$



Ellipsoidkollision - Ellipsoid approximiert

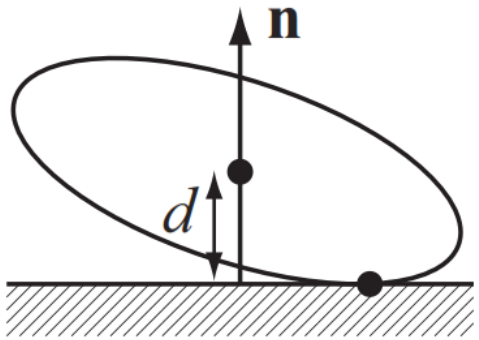


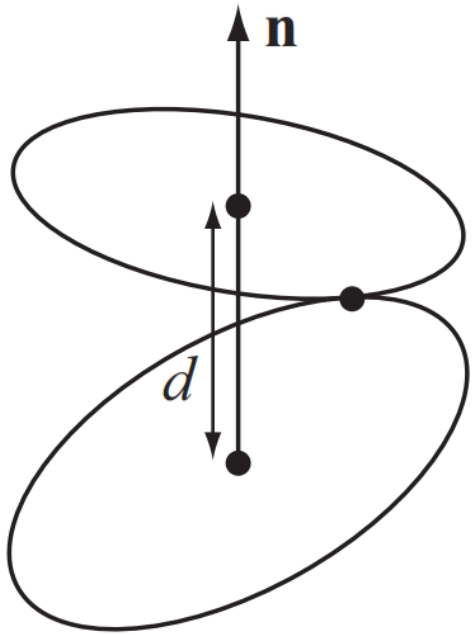


Kollision, wenn Partikel näher an Ebene/Sphäre als sein Radius in Richtung der Normalen $r_{\mathbf{n}}$

$\rightarrow$ Kollision nur korrekt, wenn $\mathbf{n}$ an Hauptachse des Ellipsoid ausgerichtet ist

Ellipsoidkollision - Ellipsoid korrekt





Kollisionsbewältigung mit korrekter Distanz d

$\rightarrow$ Zur Berechnung von d siehe Paper

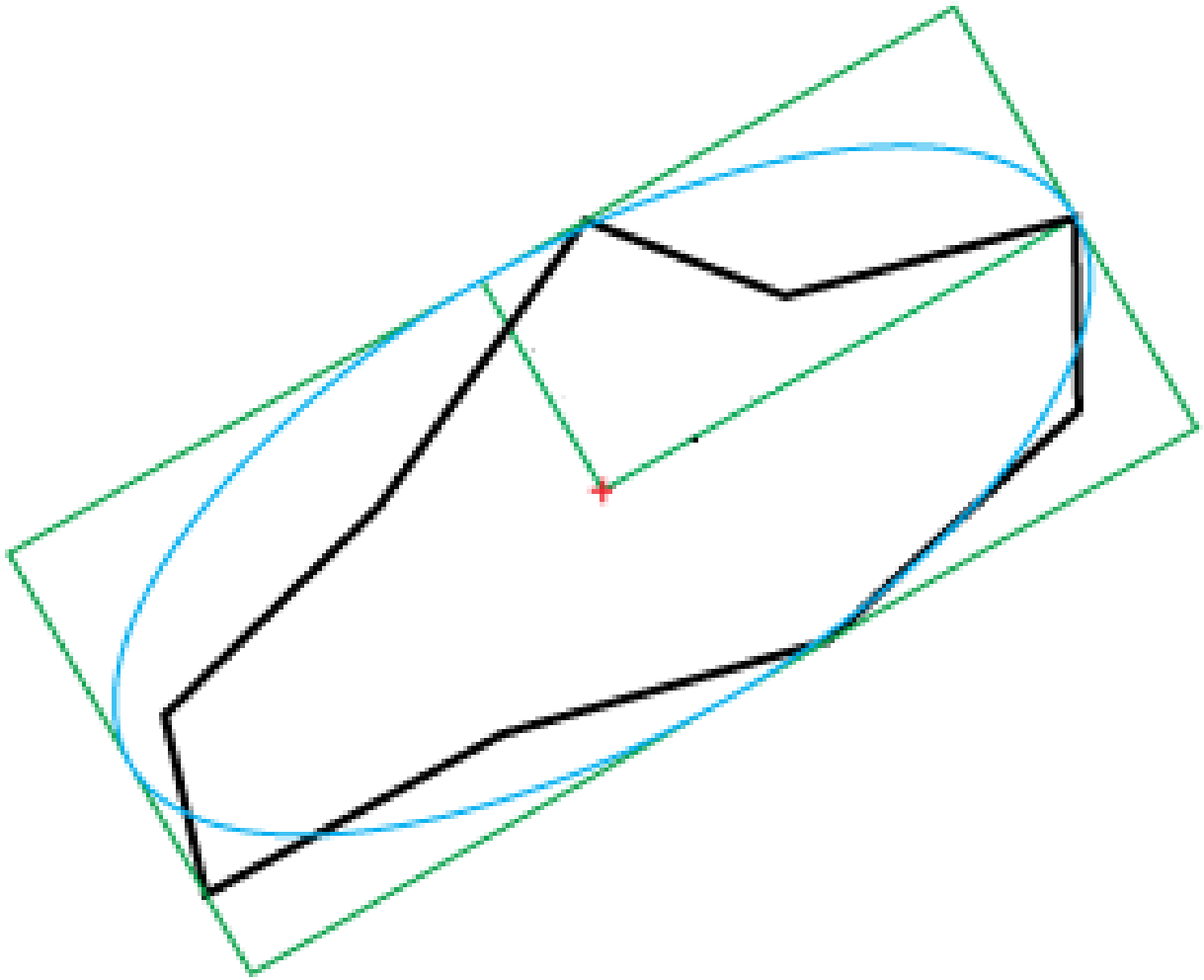
Repräsentation von Objekten durch Ellipsoide

Ausgangspunkt: Nutzer hat Simulationspartikel platziert

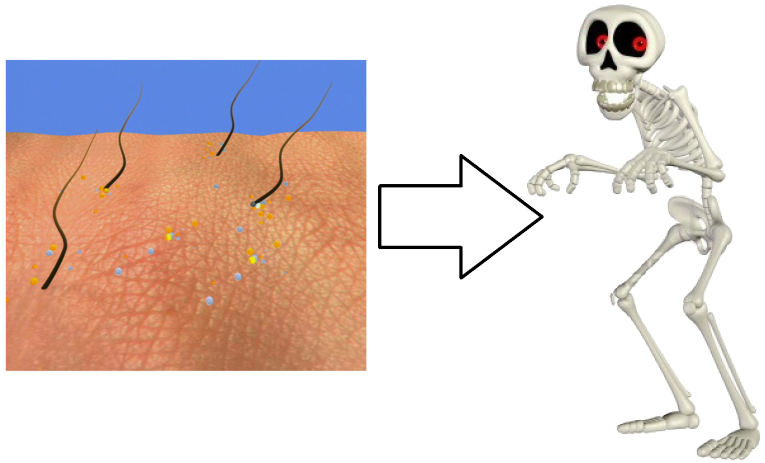
- Nutze räumliches Hashing zur Findung überlappender Partikel
- Berechne **Hauptachsenrichtungen** ...
  - Finde alle Vertices innerhalb des Radius eines Partikels
  - Berechne Kovarianzmatrix der Vertexwolke
  - Nutze Polarzerlegung zur Bestimmung der Orientierung
    $\sim$ Gewichtete Hauptkomponentenanalyse (WPCA)
- und **Radien**
  - Nutze Größe der Oriented Bounding Box (OOB) der Vertices

Repräsentation von Objekten durch Ellipsoide

Anschauliches Beispiel:



Skinning des visuellen Gitternetzes



Was ist Skinning?

→ Zuordnung des visuellen
Gitternetzes zum zugrundeliegenden physikalischen Skelett
(→ Ellipsoide + Kanten)

→ Definiert die **sichtbare** Verformung eines Objekts

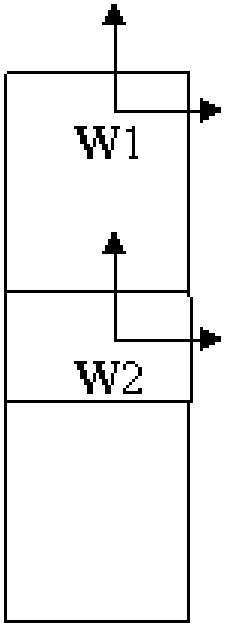
Skinning des visuellen Gitternetzes

→ **Linear Blend Skinning** (LBS)

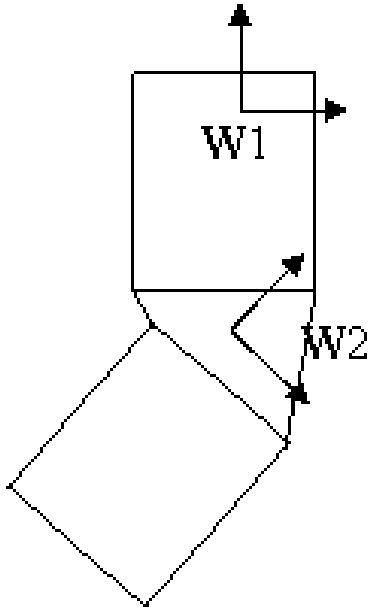
- Verbindungen & Gewichte zwischen Vertices und naheliegenden Partikeln
- Vertextransformation durch gewichteten Einfluss verbundener Partikel (→ gewichtetes Mittel)
- IdR ein bis zwei Verbindungen zu Partikeln
- Mehr als 4 Verbindungen selten sinnvoll

Skinning des visuellen Gitternetzes

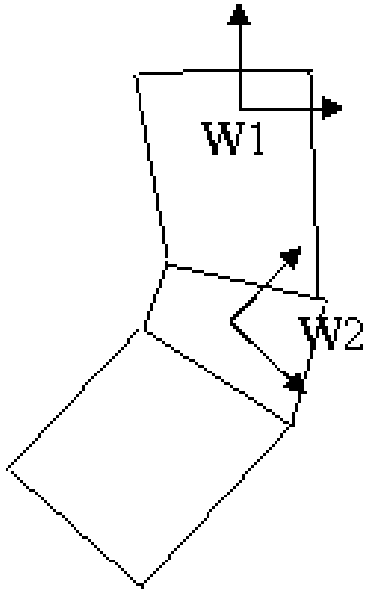
Einfaches Beispiel:



Ungebeugtes Knie, 2 Gelenke



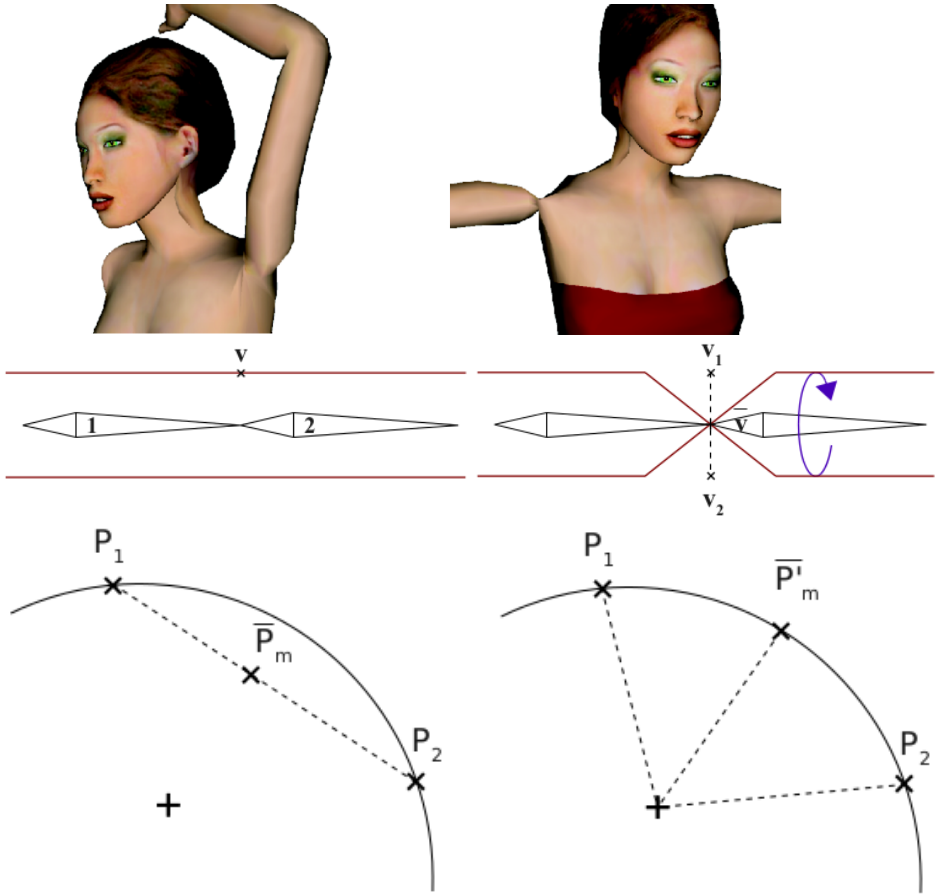
Gebeugtes Knie, jeder Vertex mit **einem** Gelenk verbunden



Gebeugtes Knie, mehr als eine Verbindung pro Vertex

Skinning des visuellen Gitternetzes

Mögliche Alternative → **Dual Quaternion Skinning** (DQS)



Candywrapper-Artefakt

Candywrapper schematisch

Links: Mittelung der Punkte (LBS)
Rechts: Mittelung der Winkel (DQS)

Simulationsframework

Designtool für physikalische Repräsentation von grafischem Gitternetz:

1. Simulationspartikel durch Spraytool
   (auch einzeln → Präzision)
2. Kanten mit Spraytool
   (auch einzeln, kürzer als spezifische Distanz)
3. Skinninggewichte automatisch

Simulationsframework

Weitere Tools:

- Partikel- & Torsionsfestigkeit
- Explizite SM-Gruppen
- Skinninggewichte anpassen
- Ellipsoide automatisch einfügen

Ergebnisse

Für Simulationen verwendete Hardware:

- 1 Kern von Intel Core2 @ 2.4 GHz
- NVIDIA Quadro FX 5800

Ergebnisse - Tintenfisch

Hier war Teil 1 von Video:

`http://www.youtube.com/watch?v=LRHqs4GJuCA`

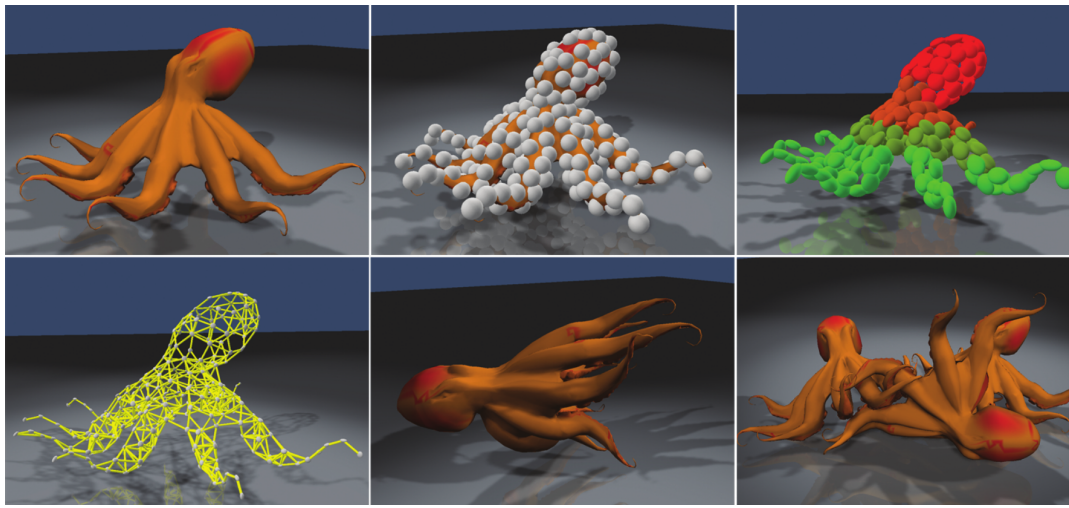
Ergebnisse - Tintenfisch

Physikalisches Gitternetz:

- 300 Partikel
- 750 Kanten

Visuelles Gitternetz:

- 5k Vertices
- 10k Dreiecke



Zeitaufwand:

- Simulation des Modells mit Selbstkollision $\rightarrow$ 4 ms
- Skinning & Rendering $\rightarrow$ 6 ms
- 3 interagierende Tintenfische (Solver) $\rightarrow$ 10 ms

Ergebnisse - Feuerfisch

Hier war Teil 2 von Video:

`http://www.youtube.com/watch?v=LRHqs4GJuCA`

Ergebnisse - Feuerfisch

Physikalisches Gitternetz:

- 1k Partikel
- 3.5k Kanten

Visuelles Gitternetz:

- 30k Vertices
- 48k Dreiecke

Zeitaufwand:

- Skinning $\rightarrow$ 9 ms
- Solver $\rightarrow$ 10 ms

Ergebnisse - Feuerfisch

Hier waren Ausschnitte von Video:

`http://www.youtube.com/watch?v=AUyCCRODkAA`

Ergebnisse - Monstertruck

Hier war Teil 3 von Video:

`http://www.youtube.com/watch?v=LRHqs4GJuCA`

Ergebnisse - Monstertruck

Physikalisches Gitternetz:

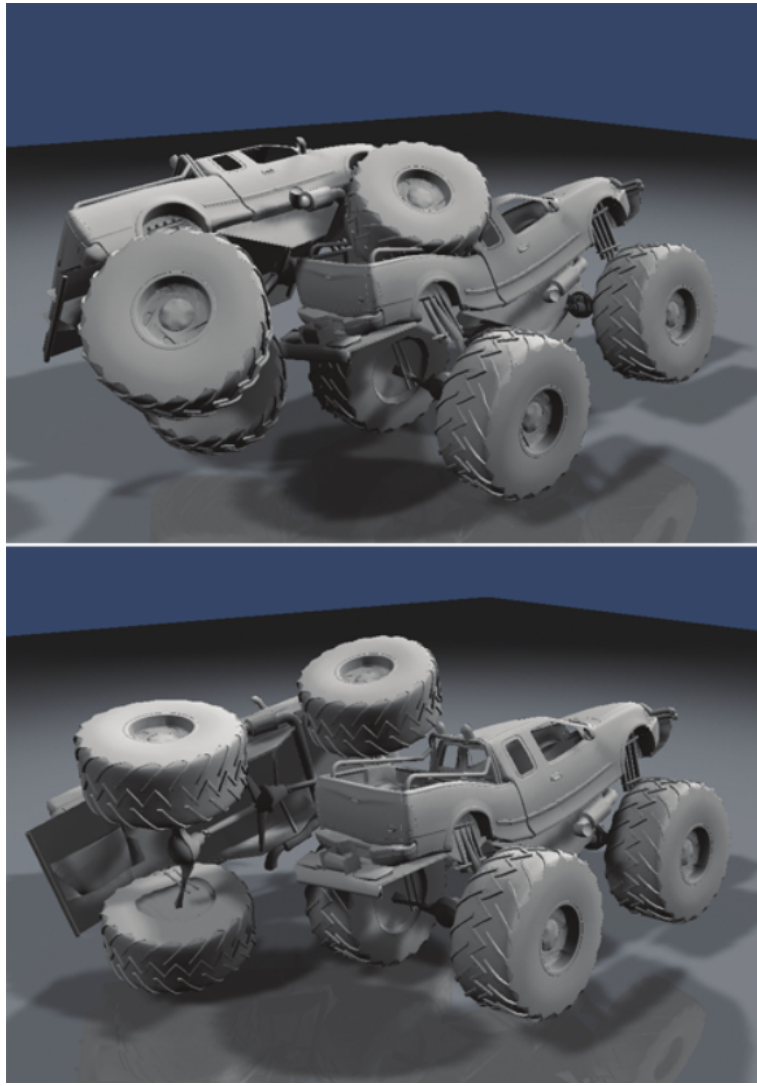
- 300 Partikel
- 800 Kanten

Visuelles Gitternetz:

- 63k Vertices
- 100k Dreiecke

Zeitaufwand:

- Skinning $\rightarrow$ 30 ms
- 2 interagierende Trucks (Solver) $\rightarrow$ 8 ms

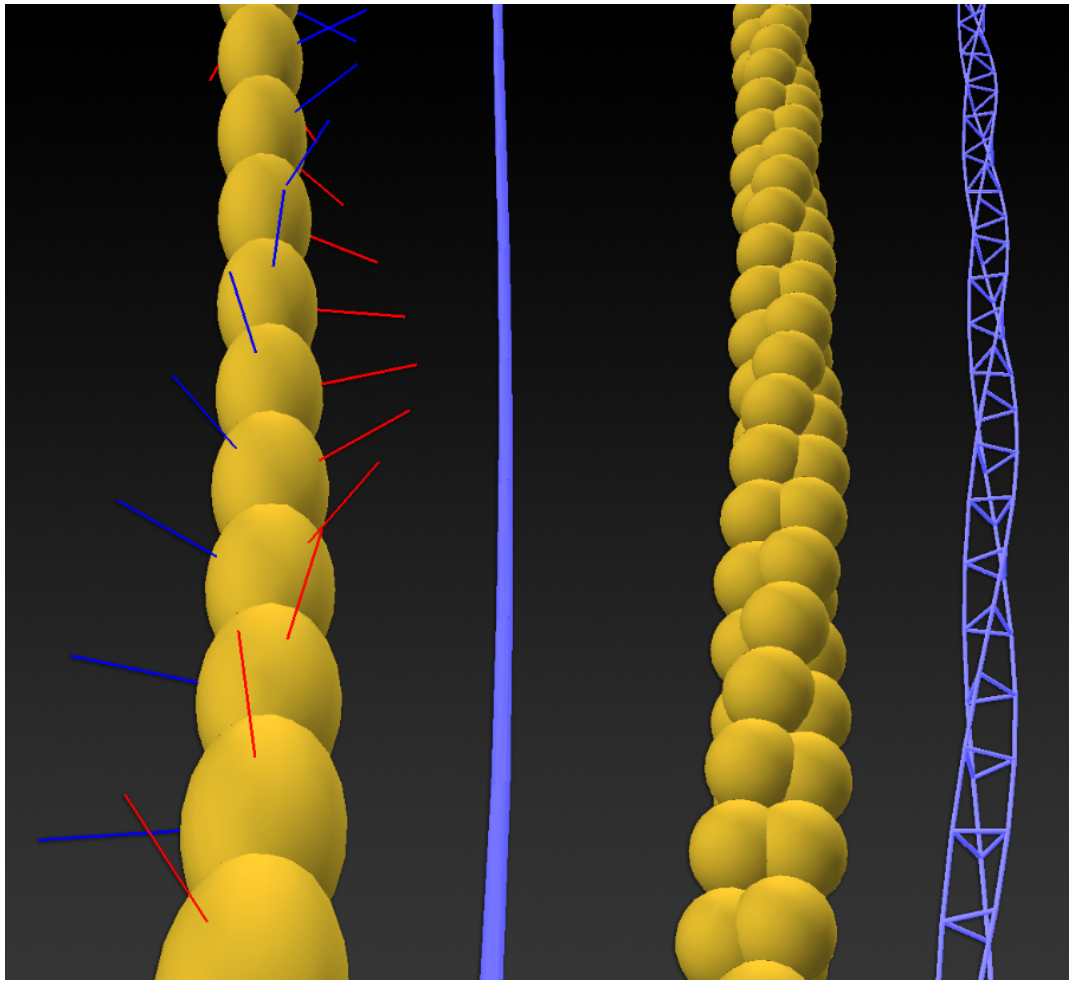


Ergebnisse - Seil

Hellblau: Kantenstruktur
Rot: Orientierung

Rechts:

- 6x mehr Partikel
- 6x mehr Zeit



Fazit

Pro:

- Neue Methode zur Festköpersimulation mit nur wenigen Partikeln
- SM auch in dünnen Bereichen durch Orientierungs- und Rotationsinformationen
- Neue Info auch für Kollisionsvolumen und Skinning nützlich
- Vielseitige physikalische Modelle in Minuten

Contra:

- Variable Zeitschritte schwer zu realisieren
- Physikalisches Verhalten abhängig von Netzgeometrie
  $\rightarrow$ keine Konvergenz
  ($\rightarrow$ Problematisch bei auto. Generierung von Modellen unterschiedliche Auflösung für Laufzeitabschätzung)

Referenzen

- http://www.matthiasmueller.info/publications/orientedparticles.pdf
- http://www.youtube.com/watch?v=LRHqs4GJuCA
- http://www.youtube.com/watch?v=AUyCCRODkAA
- http://www.mit-heiler-haut.de/05/fotos/Ani-Intakte-Haut.jpg
- http://de.123rf.com/photo_13177445_illustration-von-einem-gl-cklichen-skelett-cartoon-auf-weissem-hintergrund.html
- http://graphics.ucsd.edu/courses/cse169_w05/3-Skin.htm
- http://irit.fr/~Rodolphe.Vaillant/?e=29
- http://www.seas.upenn.edu/~ladislav/papers/sdq-tog08/sdq-tog08.pdf
- http://www.cc.gatech.edu/~turk/my_papers/sph_surfaces.pdf
- http://www.beosil.com/download/MeshlessDeformations_SIG05.pdf
- http://www.euclideanspace.com/maths/algebra/realNormedAlgebra/quaternions/geometric/axisAngle/
- http://upload.wikimedia.org/wikipedia/commons/4/4f/Twisted_bar.png