Seminar Computergraphik

Georg-August-Universität Göttingen

Ausarbeitung zum Seminarvortrag

Polygonnetz-freie Deformationen mittels Formangleich

nach einer Veröffentlichung von

M. Müller, B. Heidelberger, M. Teschner und M. Gross [3]

Christian Georg Wehrberger
christian@wehrberger.de

Seminarleitung:	Prof. Dr. W. Kurth,
	Dipl.-Inf. R. Hemmerling
Datum des Seminarvortrags:	24.01.2012
Ausarbeitung abgegeben am:	28.02.2012

Inhaltsverzeichnis

1 Einleitung

Zur Simulation von Deformationen existieren verschiedenste Verfahren, die sich auf physikalische Korrektheit der Simulation, die physikalische Richtigkeit der Darstellung verwendeter Materialien und die Stabilität der Verfahren konzentrieren. Obwohl diese Verfahren immer wieder verfeinert und verbessert wurden, werden sie in Computerspielen und interaktiven Szenarien nahezu nicht eingesetzt. Das ist zum Teil dem hohen Rechenaufwand geschuldet, den diese Verfahren benötigen. Weiterhin ist es für den Einsatz z. B. in Computerspielen vonnöten, degenerierte Geometrien korrekt zu behandeln und numerische Stabilität gewährleistet zu haben. Dabei ist es auch von großem Vorteil, den Detailgrad der Simulation an die zu verfügbaren Ressourcen anzupassen. Das in der Veröffentlichung von M. Müller, B. Heidelberger, M. Teschner und M. Gross [3] vorgestellte Verfahren zur *Polygonnetz-freien Deformationen mittels Formangleich* ist ein geometrisch motiviertes Verfahren, das keinerlei Konnektivitätsinformation der Punktemenge benötigt, die deformiert werden soll. Sowohl die Stärke der Deformation als auch ihr Detailgrad können an die entsprechende Situation angepasst werden. Beliebige Geometrien können numerisch stabil deformiert werden. Obwohl nicht physikalisch korrekt ist das im weiteren vorgestellte Verfahren näherungsweise physikalisch realistisch.

1.1 Verwandte Arbeiten

Das Spektrum an Deformationsmodellen ist breit und reicht von elastisch-deformierbaren Modellen über Masse-Feder-Modelle bis hin zu verschiedenen numerischen Verfahren zur Lösung partieller Differentialgleichungen (Randelement-Methode, Finite-Elemente-Methode, Finite-Volumen-Methode), die in den folgenden Absätzen skizziert werden.

Elastisch-deformierbare Modelle Eine mit Methoden der Elastizitätstheorie aufgestellte Differentialgleichung beschreibt eine Wechselwirkung von Trägheitskräften, Dämpfungskräften, inneren Kräften (Verspannungen) und externen Kräften. Diese Differentialgleichung wird numerisch gelöst. Die Lösungen dieser Gleichung beschreiben die Verformung einer Funktion, die einen beliebigen Körper darstellt. [5]

Masse-Feder-Modelle Auch Masse-Feder-Modelle sind physikalisch motiviert. Alle Punkte werden als Massen aufgefasst, die über Federn und Dämpfungsgliedern (Kanten) miteinander gekoppelt sind. An Massen dieses Systems können nun externe Kräfte angreifen. Aus allen externen Kräften, Trägheitskräften und Dämpfungskräften entsteht ein (großes) System gekoppelter Differentialgleichungen, das zur Simulation der Deformation gelöst wird. [6]

Randelement-Methode, Finite-Elemente-Methode, Finite-Volumen-Methode Die Finite-Elemente-Methode ist ein Lösungsverfahren partieller Differentialgleichungen. Im Fall von Deformationsmodellen werden häufig strömungsmechanische Gleichungen verwendet, um

das zu deformierende Objekt zu beschreiben. Die Finite-Volumen-Methode ist zur Lösung von Differentialgleichungen geeignet, denen ein Erhaltungssatz zugrunde liegt; die Randelement-Methode operiert im Gegensatz zur Finite-Volumen-Methode nur auf Oberflächen. Während aus der Finite-Volumen-Methode ein Gleichungssystems resultiert, das sehr groß, aber schwach besetzt ist, ergibt die Randelement-Methode ein Gleichungssystem, welches zwar kleiner, aber dafür stärker besetzt ist. [1, 2]

2 Numerische Integration

Die Lösung einer Differentialgleichung durch numerische Integration kann grundsätzlich entweder implizit oder explizit durchgeführt werden. In jeden Iterationsschritt wird dabei der Funktionswert zum nächsten Zeitschritt berechnet. Bei impliziten Lösungsverfahren wird die Gleichung $G(X(t), X(t + \Delta t)) = 0$ nach dem Funktionswert $X(t + \Delta t)$ aufgelöst. Implizite Integrationsverfahren werden vor allem aufgrund ihrer Zeitschritt-unabhängigen numerischen Stabilität eingesetzt, obwohl das Lösen der Differentialgleichung in der Regel rechenintensiv ist. Explizite Verfahren berechnen den Funktionswert $X(t + \Delta t)$ des zukünftigen Zeitschritts direkt mittels $X(t + \Delta t) = F(X(t))$. Die Berechnung stellt sich rechnerisch weniger kostenintensiv dar, allerdings ist die numerische Stabilität expliziter Integrationsverfahren abhängig vom für die Integration gewählten Zeitschritt Δt. Am Beispiel des modifizierten (expliziten) Euler-Schemas soll die numerische Instabilität expliziter Verfahren diskutiert werden.

2.1 Modifiziertes Euler-Schema

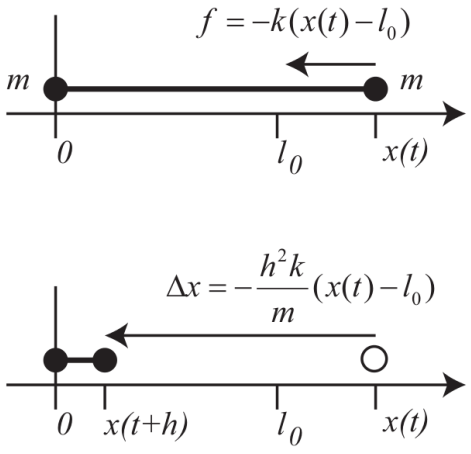

Abbildung 1: Harmonischer Oszillator [3].

Für die Diskussion der numerischen Stabilität dieses expliziten Integrationsverfahrens wird das physikalische Beispiel eines einzelnen harmonischen Oszillators betrachtet. Die betrachtete Feder mit Federkonstante k ist an einem ihrer Enden fest eingespannt, während am anderen

Ende eine punktförmige Masse m angebracht ist. Die Ruhelage der Feder wird mit l_0 bezeichnet. Abbildung 1 zeigt das betrachtete System und die verwendeten Bezeichnungen. Dabei entspricht f der rückstellenden Kraft. Die Anwendung des modifizierten Euler-Schemas ergibt folgende Differentialgleichungen:

$$v(t+h) = v(t) + h\frac{-k(x(t) - l_0)}{m} \tag{1}$$

$$x(t+h) = x(t) + hv(t+h) \tag{2}$$

Hierbei wird mit v die erste Zeitableitung des Ortes bezeichnet (Geschwindigkeit). In Matrixschreibweise erhält man also:

$$\begin{pmatrix} 1 & -\frac{hk}{m} \\ h & 1 - \frac{h^2k}{m} \end{pmatrix} \begin{pmatrix} v(t) \\ x(t) \end{pmatrix} = \begin{pmatrix} v(t+h) \\ x(t+h) \end{pmatrix} \tag{3}$$

Die Eigenwerte λ der System-Matrix ergeben sich zu:

$$\lambda_{1/2} = 1 - \frac{1}{2m}(h^2k \pm \sqrt{-4mh^2k + h^4k^2}) \tag{4}$$

Die numerische Stabilität der Integration hängt nun vom Spektralradius der System-Matrix ab. Dieser ist definiert als

$$\rho(A) = \max_{1 \leq i \leq n} |\lambda_i(A)| \tag{5}$$

und muss die Bedingung

$$\rho(A) \leq 1 \tag{6}$$

erfüllen, damit die Integration unbedingt numerisch stabil ist. Dies bedeutete in diesem Fall $h \leq 2\sqrt{\frac{m}{k}}$.

3 Algorithmus

Der Algorithmus zur Deformation ist geometrisch motiviert. Er kommt ohne Zusammenhangsinformation der Punktmenge aus und benötigt ausschließlich die initiale Konfiguration und Gewichtungen (Massen) der Punkte.

Der erste Schritt ist die Berechnung einer deformierten Punktkonfiguration durch Anwenden der Kraftgesetze, die für den betrachteten Fall gelten. Anschließend wird der Formangleich vorgenommen und damit Zielpositionen determiniert, zu denen im nächsten Schritt die lose deformierte Konfiguration "hingezogen" wird. Der Algorithmus ist in Abbildung 2 dargestellt.

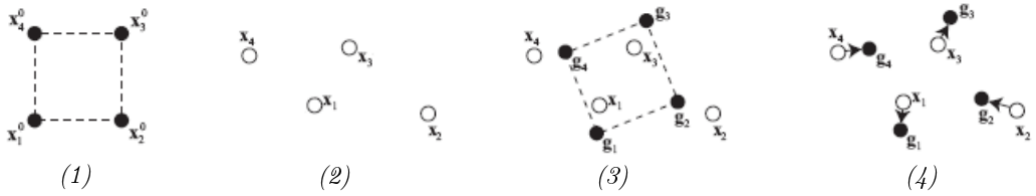

Abbildung 2: Algorithmus zum Formangleich [4].

3.1 Formangleich

Seien die Punktemengen x_i^0 (initiale Konfiguration) und x_i (deformierte Konfiguration) gegeben. Dann werden die Rotationsmatrix R und Verschiebungen t_0, t gesucht, sodass der Ausdruck

$$\sum_i w_i(R(x_i^0 - t_0) - (x_i - t))^2 \tag{7}$$

minimiert wird (dabei w_i Gewichtung). Das Ergebnis dieser Minimierung ist direkt

$$t_0 = \frac{\sum_i w_i x_i^0}{\sum_i w_i} := x_{cm}^0 \tag{8}$$

$$t = \frac{\sum_i w_i x_i}{\sum_i w_i} := x_{cm}. \tag{9}$$

Für die Bestimmung der Rotationsmatrix wird eine allgemeine lineare Transformation A angesetzt, die sich im Minimierungsproblem zu

$$A = \left(\sum_i w_i(x_i - x_{cm})(x_i^0 - x_{cm}^0)^T\right)\left(\sum_i w_i(x_i^0 - x_{cm}^0)(x_i^0 - x_{cm}^0)^T\right) := A_{pq}A_{qq} \tag{10}$$

ergibt. Mittels Polarzerlegung kann der Rotationsanteil von A extrahiert werden. Für diese Zerlegung muss nur die Matrix A_{qq} betrachtet werden, weil A_{pq} ausschließlich Streckungen darstellt (symmetrische Matrix). Die Polarzerlegung ergibt:

$$R = A_{pq}\sqrt{A_{pq}^T A_{pq}}^{-1} \tag{11}$$

Die Zielpositionen g_i ergeben sich schließlich zu:

$$g_i = R(x_i^0 - x_{cm}^0) + x_{cm} \tag{12}$$

3.2 Integration

Mit dem Wissen über die Zielpositionen kann die explizite Integration nun bedingungslos stabil durchgeführt werden. Das Integrationsschema sieht wiefolgt aus:

$$v_i(t+h) = v_i(t) + \alpha\frac{g_i(t) - x_i(t)}{h} + h\frac{f_{ext}(t)}{w_i} \tag{13}$$

$$x_i(t+h) = x_i(t) + hv_i(t+h) \tag{14}$$

Dabei ist $\alpha \in [0 \dots 1]$ ein Steifigkeits-Parameter.

Die System-Matrix hat dann folgende Form:

$$\begin{pmatrix} 1 & -\frac{\alpha}{h} \\ h & 1-\alpha \end{pmatrix} \tag{15}$$

Ihre Eigenwerte ergeben sich zu

$$\lambda_{1/2} = (1 - \frac{\alpha}{2}) \pm \frac{i}{2}\sqrt{\alpha(4-\alpha)}. \tag{16}$$

Für ihren Spektralradius ergibt sich

$$|\lambda_{1/2}| = 1. \tag{17}$$

Die numerische Stabilität ist also bedingungslos stabil, unabhängig von der Wahl des Zeitschritts.

3.3 Diskussion des Ansatzes

3.3.1 Berechnung

Für den Formangleich kann die Vorberechnung von x^0_{cm} und $x^0_i - x^0_{cm}$ durchgeführt werden. Besonders bei aufwendigen (Realzeit-)Szenarien kann dies die Geschwindigkeit (Bildwiederholrate) der Simulation erhöhen. In jedem Zeitschritt der Simulation müssen allerdings sowohl A_{pq} wie auch der Ausdruck $\sqrt{A^T_{pq}A_{pq}}^{-1}$ berechnet werden. Unabhängig von der Anzahl der Punkte kann der Term $A^T_{pq}A_{pq}$ in konstanter Laufzeit diagonalisiert werden.

Der bisherige Ansatz ist gut geeignet für kleine Deformationen. Auf die Simulation größerer Deformationen wird im Abschnitt *Erweiterungen* eingegangen.

3.3.2 Physikalischer Realismus

Der Formangleich ist nur näherungsweise physikalisch realistisch, weil er vor allem geometrisch motiviert ist. Trotzdem erfüllt er zwei wesentliche Kriterien, die ein Simulationsszenario realistisch wirken lassen. Die Verwendung des Massenschwerpunktes beim Formangleich (siehe Gleichung 9) garantiert Impulserhaltung. Wenn für die Gewichtungen der Punkte deren Massen verwendet werden, ist außerdem auch Drehimpulserhaltung gegeben.

4 Erweiterungen

4.1 Starre Körper

Indem der oben eingeführte Steifigkeitsparameter auf $\alpha = 1$ gesetzt wird, kann das Verhalten eines starren Körpers simuliert werden. Mit dieser Wahl von α werden die Zielpositionen in einem Zeitschritt erreicht und entsprechen der Ausgangskonfiguration bis auf Drehung und Verschiebung.

4.2 Allgemeine lineare Deformationen

Um stärkere Deformationen zu ermöglichen, kann anstelle der bisherigen Transformation eine allgemeine lineare Transformation verwendet werden. Statt einer Rotationsmatrix R wird eine zusammengesetzte Matrix

$$\beta A + (1 - \beta)R \tag{18}$$

eingesetzt. Der allgemeine lineare Anteil wird durch A beschrieben, das mit dem Faktor β gewichtet wird, der Rotationsanteil weiterhin durch R, gewichtet mit Faktor $1 - \beta$. Die Beibehaltung eines reinen Rotationsanteils gewährleistet, dass die initiale Punktkonfiguration nicht zu stark "zerfließt" und kann als Analogie zum einem Polygonnetz betrachtet werden, das eine Zusammenhangsinformation beinhaltet. Zur Volumenerhaltung muss die Matrix A weiterhin mit dem Faktor $1/\sqrt[3]{A}$ normiert werden. Die genaue Berechnung von A wurde bereits vorhergehend betrachtet (Gleichung 10).

4.3 Quadratische Deformationen

Über die Flexibilität der linearen Transformation hinausgehend können auch quadratische Transformationen eingesetzt werden (Twist und Biegung). Die verwendete Transformation ist analog zur vorherigen Vorgehensweise gegeben durch

$$\beta \tilde{A} + (1 - \beta)\tilde{R} \tag{19}$$

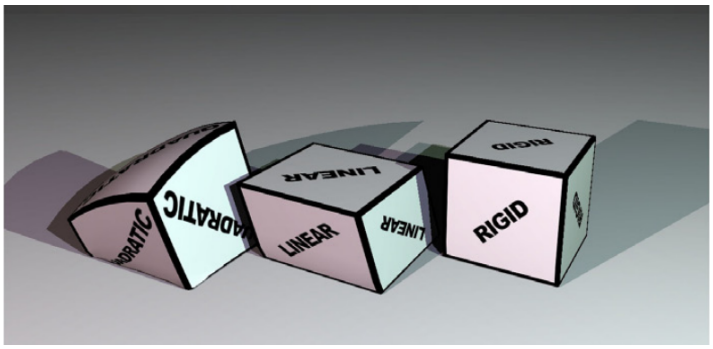

Abbildung 3: Vergleich von linearer, quadratischer und starrer Deformation [3].

mit quadratischem Anteil $\tilde{A}$. Dieser Anteil ist

$$\tilde{A} = [A\ Q\ M] \in \mathbb{R}^{3\times 9}, \tag{20}$$

wobei A den linearen Anteil darstellt, Q den rein quadratischen und M die gemischten Terme repräsentiert. $\tilde{R} = [R\ 0\ 0] \in \mathbb{R}^{3\times 9}$ stellt einen Rotationsanteil dar, der zwecks Formerhaltung benutzt wird. Die Lösung des Minimierungsproblems ergibt:

$$\tilde{A} = \left(\sum_i w_i(x_i - x_{cm})\tilde{q}_i^T\right)\left(\sum_i w_i\tilde{q}_i\tilde{q}_i^T\right) \tag{21}$$

Damit lassen sich nun die Zielpositionen berechnen:

$$\tilde{g}_i = \tilde{A}\tilde{q}_i \tag{22}$$

(mit $\tilde{q} = \begin{pmatrix} q_x & q_y & q_z & {q_x}^2 & {q_y}^2 & {q_z}^2 & q_xq_y & q_yq_z & q_zq_y \end{pmatrix}^T$ und $q_{i\alpha} = (x_i^0 - x_{cm}^0)_\alpha$ ($\alpha \in \{x, y, z\}$)

In Abbildung 3 werden lineare und quadratische Transformation miteinander verglichen.

4.4 Deformation von Punktgruppen

Auch unter Verwendung von quadratischen Transformationen sind besonders starke Verformungen nicht möglich. Unter Zuhilfenahme Punktgruppen-basierter Deformationen erhöht sich die Flexibilität des Modells drastisch (Abbildung 4). Eine einfache Methode Punktgruppen zu bilden ist eine regelmäßige Einteilung der initialen Punktwolke in Kuben. Die Ortsänderung einer Punktgruppe erhält dann den zusätzlichen Term:

$$\Delta v_i = \alpha\frac{g_i^c - x_i}{h} \tag{23}$$

Hierbei ist g_i^c der Zielpunkt der Punktgruppe.

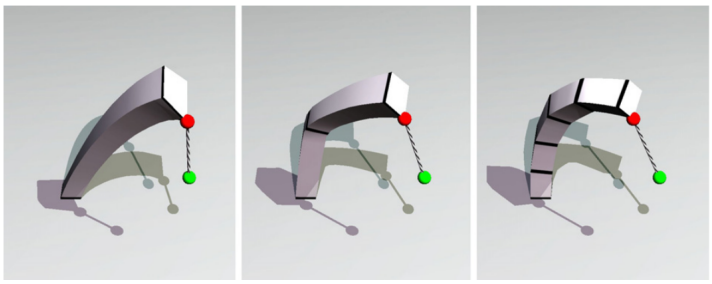

Abbildung 4: Punktgruppenbasierte Deformation [3].

4.5 Plastizität

Um Plastizität zu simulieren, kann die Polarzerlegung der linearen Transformation genutzt werden. Während im Ausdruck $A = RS$ die Matrix R den Rotationsanteil darstellt, enthält die Matrix S den Deformationsanteil der Transformation. Das Konzept der Punktgruppenbasierten Deformation kommt hier zum Einsatz. Sei S^P die Matrix, die die Zustandsinformation der Deformation enthält. Weil im Ausgangszustand keine Deformation vorliegt, wird S^P mit der Einheitsmatrix I initialisiert. In jedem Zeitschritt der Länge h wird S^P aktualisiert, sofern

$$||S - I||_2 > c_{yield}. \tag{24}$$

Die Matrix S^P verändert sich dann zu:

$$S^P \leftarrow (I + hc_{creep}(S - I))S^P \tag{25}$$

Zur Volumenerhaltung muss S^P in jedem Zeitschritt mittels Division durch $\sqrt[3]{S^P}$ normiert werden. Der Parameter c_{creep} regelt die Stärke der Deformation, während der Parameter c_{yield} eine Schwelle darstellt, ab der das Objekt überhaupt deformiert werden kann. Zur Beschränkung der Deformation kann in jedem Zeitschritt die Bedingung

$$||S - I||_2 < c_{max} \tag{26}$$

überprüft werden. Wird c_{max} überschritten, so setzt man

$$S^P \leftarrow I + c_{max}(S^P - I)/||S^P - I||_2. \tag{27}$$

Auch in diesem Fall muss S^P renormiert werden. Angewendet wird die Deformation dann durch die Ersetzung von $x_i^0 - x_{cm}^0$ durch $S^P(x_i^0 - x_{cm}^0)$ in Gleichung 10.

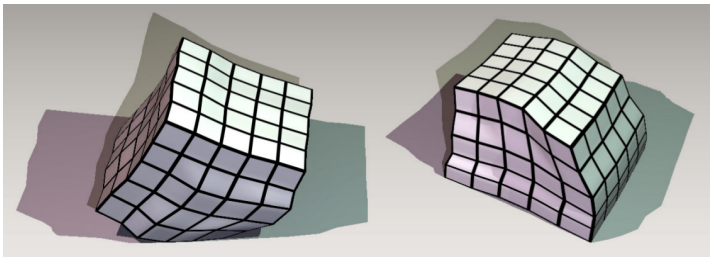

Abbildung 5: Plastische Deformation (punktgruppenbasiert) [3].

5 Ergebnisse

Testsystem Alle Simulationen wurden auf einem *Pentium 4* (3,2 GHz) durchgeführt.

5.1 Zeitverhalten

Das Zeitverhalten der Simulation ist in Abbildung 6 zu erkennen. Die Zeitmessungen wurden mit zufällig verteilten Start- und Zielpositionen der Punkte durchgeführt. Die benötigte Simulationszeit hängt ungefähr linear mit der Anzahl der Punkte sowohl für das lineare als auch für das quadratische Verfahren zusammen. Das quadratische Verfahren ist aufwändiger, benötigt aber nicht mehr als die doppelte Laufzeit des linearen Verfahrens. Auf dem verwendeten Testsystem können mit 10.000 Punkten in 8 Punktgruppen 50 Bilder pro Sekunde erreicht werden.

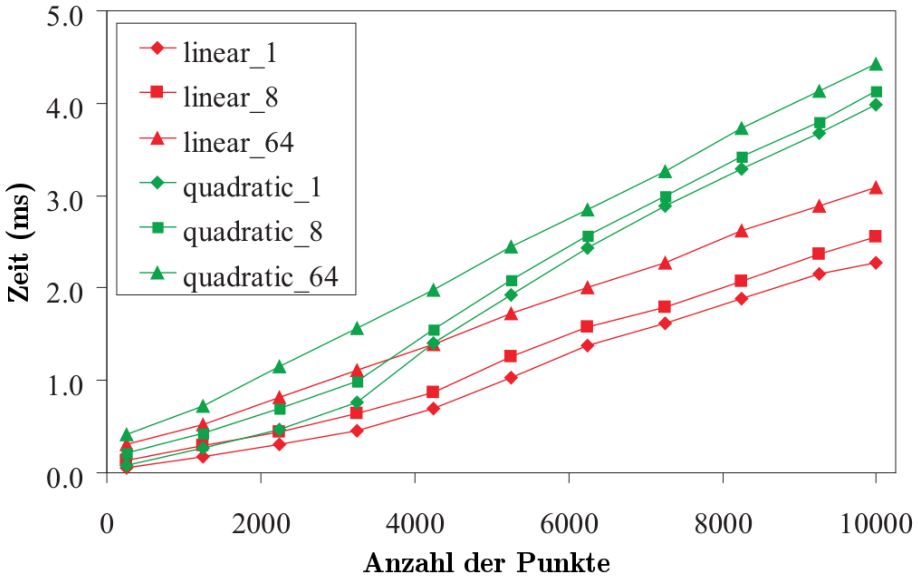

Abbildung 6: Zeitverhalten der verschiedenen Deformationen für verschieden viele Punktgruppen (1, 8 und 64) [3].

Ein aufwändigeres Testszenario ist in Abbildung 7 dargestellt. In diesem Szenario wurden 55.200 Punkte (384 Objekte) in 2.448 Punktgruppen simuliert. Je nach Objekt benötigt die Berechnung der Deformation mittels des quadratischen Verfahrens $0,008\,\text{ms}$ bis $0,096\,\text{ms}$.

Abbildung 7: Aufwändiges Simulationsszenario [3].

5.1.1 Degenerierte Geometrien

Das vorgestellte Verfahren ist auch in der Lage, degenerierte Geometrien zu simulieren. In Abbildung 8 ist eine solche zu finden. Insbesondere für interaktive Szenarien ist dies von Vorteil.

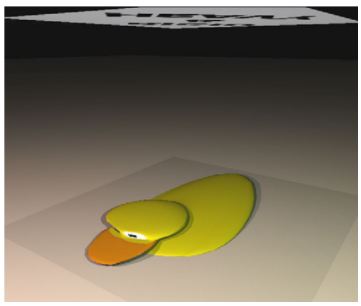

Abbildung 8: Degenerierte Geometrie [3].

5.2 Diskussion

5.2.1 Vorteile des Verfahrens

Durch die schnelle und einfache Berechnung hebt sich das beschriebene Deformationsmodell von physikalisch korrekten Deformationsmodellen ab. Die Implementation ist einfach und wenig fehleranfällig. Auch der Aufwand durch Vorberechnungen ist gering (proportional zur Anzahl der Punkte). Das Verfahren garantiert eine unbedingte numerische Stabilität durch die Kenntnis der Zielpunkte, was den Einsatz z. B. in Computerspielen erleichtert; auch die Anpassbarkeit der Deformationsstärke trägt wesentlich dazu bei. Obgleich das Verfahren nicht physikalisch korrekt ist, bietet es trotzdem physikalisch plausible Resultate durch Impuls- und

Drehimpulserhaltung.

Das Verfahren ist aufgrund dieser Vorteile geeignet für Computerspiele und interaktive Software, bei der Deformationen simuliert werden sollen. Auf modernen Rechnern ist eine Realzeit-Deformation auch großer Punktmengen bei gleichzeitig flüssiger Darstellung möglich.

5.2.2 Einschränkungen

Anders als typischerweise bei Modellen der Fall, die im Abschnitt *Verwandte Arbeiten* vorgestellt wurden, kann durch die Verwendung höherer Deformationsmoden keine höhere physikalische Genauigkeit geschlussfolgert werden, weil das Modell nur geometrisch motiviert ist.

Die Erweiterung des Verfahrens zur Darstellung plastischen Verhaltens deformiert ein Objekt nur dann, wenn eine gewisse Schwelle der einwirkenden Kraft überschritten wird. Reale Objekte verformen sich allerdings auch unter langer Einwirkung einer kleinen Kraft.

Das Verfahren benötigt lediglich eine Punktmenge ohne Zusammenhangsinformation. Dies erschwert Kollisionserkennung oder macht sie im Zweifelsfall sogar unmöglich. Die für die Simulationsszenarios eingesetzten Kollisionserkennungen stellten zudem einen Flaschenhals für die Effizienz des Verfahrens dar.

6 Anhang

Literatur

[1] An online resource for information on the Boundary Element Method. In: *http://www.boundaryelements.com/index.php (abgerufen am 22.01.2012)*

[2] Ang, Whye-Teong: A Beginner's Course in Boundary Element Methods. In: *http://www.ntu.edu.sg/home/mwtang/bem2011.html (abgerufen am 22.01.2012)*

[3] Müller, M. ; Heidelberger, B. ; Teschner, M. ; Gross, M.: Meshless deformations based on shape matching. In: *ACM Transactions on Graphics (TOG)* 24 (3) (2005), S. 471–478

[4] Müller, M. ; Heidelberger, B. ; Teschner, M. ; Gross, M.: Meshless deformations based on shape matching (Report). In: *SIGGRAPH* (2005)

[5] Terzopoulos, D. ; Platt, J. ; Barr, A. ; Fleischer, K.: Elastically deformable models. In: *ACM Siggraph Computer Graphics* 21 (4) (1987), S. 205–214

[6] Terzopoulos, D. ; Waters, K.: Physically-based facial modelling, analysis, and animation. In: *The journal of visualization and computer animation* 1 (1990), Nr. 2, S. 73–80

Abbildungsverzeichnis