Universität Karlsruhe – Fakultät für Informatik – Bibliothek – Postfach 6980 – 76128 Karlsruhe

Ein vereinfachtes numerisches Verfahren für die mechanische Simulation in Virtual-Reality-Systemen

Alfred Schmitt Sven Thüring

Interner Bericht 2000-26

Universität Karlsruhe

Fakultät für Informatik

ISSN 1432 - 7864

Ein vereinfachtes numerisches Verfahren für die mechanische Simulation in Virtual-Reality-Systemen

Kurzfassung

Derzeitige Systeme für virtuelle Realität (VR-Systeme) müssen noch in erheblichem Umfang funktional erweitert werden, wenn man mit ihrer Hilfe z.B. autonome mobile Roboter als Interaktionspartner in der VR-Welt modellieren will. Insbesondere muss zukünftig verlangt werden, dass mechanische Vorgänge in der VR-Simulation weitgehend korrekt nachbildbar sind. Mit diesem Ziel im Hintergrund wird hier ein vereinfachtes numerisches Verfahren vorgestellt, welches die dynamische und kinematische Simulation von gelenkgekoppelten Mehrkörpersystemen erlaubt. Das Verfahren ist konsequent auf der elementaren Mechanik der Massenpunktsysteme aufgebaut, umgeht die komplizierte mechanische Gleichungstheorie für Mehrkörpersysteme und führt zu extrem kurzen und durchsichtigen Simulationsprogrammen, wobei allerdings bei der Körpermodellierung zunächst nur diskrete Massenpunktverteilungen realisierbar sind, was für VR-Systeme völlig ausreichend erscheint.

A simplified numeric method for the mechanical simulation in virtual realty systems

Abstract

Present virtual reality systems (VR-systems) must still extended considerably in their functionality to model autonomous mobile robots as interaction-partners. Especially it must be possible to simulate nearly precise the mechanics in the VR-simulation. With this in mind, we present a simplified numerical method, allowing the dynamical and kinematical simulation of joint-coupled multi-body-systems. This method is consequently built on the elementary mechanics of mass-point-systems. It avoids the complicated mechanical equation-theory for multi-body-systems and leads to extremely short and transparent simulation programs. Until now the modelling of objects is only feasible with discrete mass-point-distributions, but this restriction appears sufficient for VR-systems.

Inhaltsverzeichnis

1 Einleitung und Motivation

VR-Systeme ohne speziellen Simulator-Hintergrund verzichten in der Regel auf korrekte dynamische Simulation mechanischer Vorgänge (Ausnahmen: VR-Systeme wie Flug- und Fahrsimulatoren). Speziell die für allgemeine VR-Anwendungen konzipierten Software-Systeme wie z.B. VRML legen zwar großen Wert auf ansprechende Bilderzeugung, vielfältige Beleuchtungseffekte und Navigationsmöglichkeiten in komplexen Szenarien. Die in den Script-Sprachen beschriebenen Welten sind jedoch relativ tot. Zwar sind bewegte Objekte wie z.B. ein fahrendes Auto modellierbar, jedoch können diese Fahrzeuge an Straßenkreuzungen keine Unfälle erleiden, weil keine Kollisionsdetektion durchgeführt wird und daher auch keine entsprechenden Folgeereignisse modelliert werden können.

Soll zukünftig die VR-Technik jedoch auch von Ingenieuren und Konstrukteuren genutzt werden, um z.B. autonom agierende Roboter wie z.B. Laufmaschinen in der Simulation zu erproben, bevor ein reales Gerät bereits gebaut ist, so müssen die Modellierungsmöglichkeiten der VR-Systeme noch wesentlich ausgeweitet werden. Geht man von dem erweiterten VR-Paradigma aus, dass in der VR-Simulation möglichst viele Effekte der realen Welt mit den Möglichkeit der Virtualität darstellbar sein sollen, so ist von zukünftigen VR-Systemen zu fordern:

(a) Komplexe mechanische Modellierungen: Starre Körper, Roboterarme, Greifbewegungen, gelenkgekoppelte Mehrkörpersysteme, Kollisions- und Stoß-Effekte, Servo-Antriebe, Pneumatik, Reibungseffekte usw. Vergleiche hierzu z.B. die ältere Arbeit von Isaacs und Cohen [IC87], in der viele derartige Aspekte angesprochen werden.

(b) Simulierte Sensorik und Messtechnik: Berührungs-, Ultraschall-, Infrarot-Sensoren, Kraftsensoren, simulierte Video-Kamera-Augen, Akustik-Simulation usw.

(c) Softwareintegration: Wenn z.B. ein autonom agierender Roboter in eine VR-Welt eingebettet werden soll, so muss die gesamte Software des zu simulierenden Roboters inklusive Sensorik-Input integraler Bestandteil der VR-Simulation sein. Man denke nur an die aufwendigen Lernverfahren in der Roboter-Software, die unbedingt mit ablaufen müssen, wenn die Roboter realistisch in einer VR-Welt erprobt und trainiert werden sollen.

(d) VR-Immersion für Menschen: Interagieren mit der VR-Simulation sowohl aus externer Sicht (Beobachter-, Bedienersicht) als auch in der direkten Immersion mit Datenanzug, Datenhandschuh und 3D-Sicht. Problematisch und technisch schwierig ist die technische Realisierung der Kraftkopplung im Ganzkörperbereich.

Selbstverständlich soll dabei die realitätsnahe Visualisierung (Bilderzeugung), die flexible Navigation und die flexible Interaktion mit der simulierten Welt nicht vernachlässigt werden.

In Einzelfällen und für spezielle Anwendungen wie z.B. Roboter-Simulationen wurde bereits in der Vergangenheit komplexe Simulationssoftware eingesetzt. Was jedoch den derzeitigen VR-Entwicklungen fehlt, ist die Integration der vielen obengenannten Effekte unter dem Dach eines einzigen Systems, das seiner ganzen Natur nach

umfassend und offen sein muss. Das bedeutet insbesondere, dass das VR-System als Definitionsbasis auf einer Skript-Sprache aufgebaut sein muss. Beispiele für komplexere Skriptsprachen findet man in der Computeranimation, der Bilderzeugung (Raytracer) und z.B. auch bei VRML = „Virtual Reality Modelling Language“. Ein wesentliches Charakteristikum der Skriptsprachen ist ihr aufzählender Charakter: Alle Details der zu simulierenden Welt werden in einer allgemein beschreibenden Sprache textuell definiert, und zwar nur in dem Umfang, wie sie für die jeweilige Welt-Simulation erforderlich sind. Legt das Skript z.B. für einen Quader = Starrkörper keine Farbe bzw. keine Oberflächenqualitäten wie Texturen fest, so ist das zulässig. Das VR-System muss fehlende Angaben durch einfache Default-Verfahren ergänzen, um die Bilderzeugung zu gewährleisten.

In der großen Zahl der bekannt gewordenen Formalsprachen für Computeranwendungen kann man die Skriptsprachen als diejenigen Sprachen kennzeichnen, die mindestens im semantischen Kontext den natürlichen Sprachen am nächsten stehen und daher auch am einfachsten zu erlernen sind. Sie sind allerdings in dem Sinne beschränkt, dass man nur das ausdrücken kann, was die Skript-Befehle auszudrücken gestatten. In der Bilderzeugung und der Computeranimation hat sich allerdings herausgestellt, dass man mit relativ wenigen Geometrie- Textur- und Beleuchtungselementen mit der zusammensetzenden Methode unglaublich vielfältige Bildergebnisse erzielen kann.

2 Dynamische Bewegungssimulation mit Massenpunktsystemen

Massenpunktsysteme wurden bereits in vielen numerischen Simulationen benutzt, um mechanische Eigenschaften zu studieren. Insbesondere auch für Zwecke der realitätsnahen Bewegungsmodellierung in der Computeranimation wurden sie herangezogen, um z.B. Kleider und Gewebe zu animieren, vergleiche z.B. [LPC95]. Speziell bei der Simulation elastischer Materialien sind sie neben der Finiten-Elemente-Methoden (FEM) sehr wichtig, man vergleiche Deussen [De96] und [Kuh97]. Wenn es speziell um approximative Simulationen von elastischen Materialien geht, ist der Massenpunkt-Ansatz sehr attraktiv.

Offen ist jedoch bisher, ob sich auch die Starrkörperdynamik, vor allem die gelenkgekoppelten Systeme, auf der Basis der Massenpunktsysteme realisieren lässt. Um dies näher zu ergründen, haben wir ein einfaches Simulationsverfahren für starre Massenpunktsysteme entwickelt und an kleineren Beispielen experimentell erprobt. Die überaus interessanten Resultate dieser Experimente vor allem mit der Blickrichtung auf zukünftige VR-Systeme lassen es sinnvoll erscheinen, hier sowohl das numerische Verfahren als auch erste Simulationsergebnisse vorzustellen.

Unser Ansatz basiert auf folgenden grundsätzlichen Annahmen: Ein Massenpunktsystem besteht aus n in jeweils einem Punkt $P_i = (x_i,y_i,z_i) \in \mathrm{R}^3$ ($i = 1...n$) konzentrierten Massen m_i. Der vom Zeitpunkt t abhängige Bewegungszustand des Massenpunktsystems wird hier beschrieben durch:

$P_i(t) \in \mathrm{R}^3$:= Ort des Massenpunktes i zum Zeitpunkt t,

$V_i(t) \in \mathrm{R}^3$:= Geschwindigkeitsvektor,

$F_i(t) \in \mathrm{R}^3$:= Kraftvektor, d.h. Richtung und Stärke der Kraft, die zum Zeitpunkt *t* auf den Massenpunkt *i* einwirkt,

$m_i \in \mathrm{R}$ = Masse (skalare Größe!) des *i*-ten Massenpunktes, sie wird in der Regel als über der Zeit konstant angenommen.

Im folgenden versuchen wir nicht, wie das in der klassischen Theorie der Mehrkörpersysteme üblich ist, ein System von Differentialgleichungen aufzustellen und dann numerisch zu lösen. Wir verfolgen vielmehr die Strategie, die Bewegung des einzelnen Massenpunktes, speziell die auf ihn wirkenden Kräfte und damit Beschleunigungen im einzelnen mindestens näherungsweise zu bestimmen und daraus den weiteren Bewegungsablauf des ganzen Systems zu berechnen. Der Kraftvektor $F_i(t)$ fasst die inneren Kräfte $Fin_i(t)$ und die von außen einwirkenden Kräfte $Fex_i(t)$ und die evtl. vorhandene Schwerkraft $m_i g$ zusammen:

$$F_i(t) := Fin_i(t) + Fex_i(t) + m_i g$$

Die inneren Kräfte sind in der Regel nicht bekannt. Sie wirken an Gelenken und müssen vor allem dafür sorgen, dass die einzelnen Massenpunkte, die einen Starrkörper modellieren, stets die vorgegebenen konstanten Abstände voneinander behalten, damit sich der Starrkörper nicht verformt. Mit einem relativ einfachen iterativen Annäherungsverfahren wird es uns gelingen, die inneren Kräfte $Fin_i(t)$ näherungsweise zu bestimmen.

Die oben genannten Größen sind noch keine vollständige Modellierung eines Massenpunktsystems, weil die inneren Beziehungen der Massenpunkte zueinander noch nicht spezifiziert sind. Im hier betrachteten Kontext, wir befassen uns nicht mit Feder- und Dämpfungskräften, gibt es nur dann spezielle Beziehungen zwischen zwei Punkten, wenn zwischen ihnen ein Soll-Abstand festgelegt ist und stets eingehalten werden muss. Das gilt für alle Punkte-Paare eines Starrkörpers und für Gelenkpunkte. Daneben benötigen wir noch Aufhänge-Punkte: Sie sind unveränderlich im *x*-*y*-*z*-Raum fixiert und dienen z.B. dazu, ein Pendel oder eine Gliederkette an ihnen schwingend aufzuhängen, ohne dass die auf den Punkt wirkenden Kräfte diesen wegbewegen können.

Die Abstandsbedingungen werden in einer Liste (Menge von Tripeln)

$$L = \{(i_k,j_k,d_k)\}\ (k = 1...m)$$

zusammengefasst, wobei ein Tripel (i,j,d) aussagt, dass zwischen $P_i(t)$ und $P_j(t)$ der konstante Abstand $|P_i(t) - P_j(t)| = d$ gefordert wird. Falls das Tripel (i_k,j_k,d_k) in L enthalten ist, wird im weiteren angenommen, dass auch automatisch (j_k,i_k,d_k) in L enthalten ist. Mit den initialen Punktorten $P_i(0)$, den Geschwindigkeiten $V_i(0)$, den Massen m_i und den Abstandsangaben in der Liste L ist ein Massenpunktsystem (für die Starrkörper-Simulation) vollständig definiert. Auch können die einfachen Konstant-Abstandsbedingungen durch weitere Abstandsregeln ergänzt werden, um eine größere Vielfalt kinematischer Konstruktionen zu ermöglichen.

Um das mechanische Modell aus einem Ruhezustand heraus in Bewegung zu versetzen, betrachten wir auch von außen einwirkende Kräfte *Fex,* die aber in unseren einfachen Test-Simulationen nur in einem kurzen Zeitintervall einwirken.

3 Das Einzelschritt-Integrationsverfahren

Wir beschreiben nun, wie man bei den oben eingeführten Massenpunktsystemen aus einem konsistenten Bewegungszustand zum Zeitpunkt $t = 0$ einen konsistenten Bewegungszustand zum Zeitpunkt $t = h > 0$ berechnet. Die Simulation über längere Zeitabschnitte erfolgt dann so, dass Einzelschritte mit der t-Schrittweite h nacheinander ausgeführt werden. Basis der Einzelschritte sind die elementaren Formeln:

$$V_i(h) := V_i(0) + \frac{1}{m_i}\int_0^h F_i(t)dt$$

$$P_i(h) = P_i(0) + V_i(0)h + \frac{1}{m_i}\int_0^h (\int F_i(t)dt)dt$$

Die auf den Massenpunkt i einwirkenden Kräfte $F_i(t)$ setzen sich bei uns wie folgt zusammen:

$$F_i(t) := Fin_i(t) + Fex_i(t) + m_i g$$

Die Anteile *Fex* und die Schwerkraftwirkung *mg* können als bekannt vorausgesetzt werden, während die inneren Kräfte *Fin* zunächst völlig unbekannt sind. Sie müssen für jedes neue *h*-Intervall neu bestimmt werden und haben die Aufgabe, die Abstandsbedingungen $(i,j,d) \in L$ für die neuen Punktlagen $P_i(h)$ sicherzustellen. Man vergleiche hierzu die Anmerkungen in Wittenburg [Wit77] auf Seite 39, die uns die Anregung zur Entwicklung unseres Verfahrens gegeben haben.

Bestimmung der inneren Kräfte *Fin*

Die *Fin*-Kräfte müssen so an den Massenpunkten angreifen, dass sie dem Massenpunktsystem keine Energie zuführen oder wegnehmen und daneben keine rotatorischen Einflüsse ausüben. Es muss sich also um „stille“ Kräfte handeln, die nur dem Zweck dienen, die Abstandsbedingungen nach dem *h*-Integrationsschritt wiederherzustellen.

Bei unseren ersten Rechnungen mit dem später noch genauer diskutierten Würfelmodell haben wir versucht, *Fin*-Kräfte als in Richtung und Stärke konstante, im ganzen *h*-Intervall wirkende Kräfte zu modellieren. Es gelang zwar so, die Abstandsbedingungen am Ende des *h*-Intervalls zu garantieren. Allerdings hat das Massenpunktsystem dabei zusätzliche Energie aufgenommen. Das ist darauf zurückzuführen, dass sich die Massenpunkte bewegen und dass es mit einfachen Verfahren nicht möglich ist, die Wirkungsrichtung der *Fin*-Kräfte an die Punktbewegungen zu koppeln. Das gelingt erst mit einem erheblich komplexeren Differentialgleichungsansatz, den wir vermeiden wollen.

Das Problem mit den *Fin*-Kräften konnten wir schließlich mit folgender Vorgehensweise befriedigend lösen:

- Statt kontinuierliche, im ganzen *h*-Intervall wirkende *Fin*-Kräfte zu modellieren, realisieren wir sie als starke und kurzfristig wirkende Impulse fin_i. Die

Impulsänderung findet nur in einem infinitesimal kleinen Zeitintervall statt – bei uns zum Zeitpunkt $t = 0$ –, und verleiht dem Massenpunkt sofort eine zusätzliche Geschwindigkeitskomponente der Richtung und Stärke, $^1/m_i\ fin_i$ die auf den aktuellen Geschwindigkeitsvektor aufaddiert wird.

- Die einzelnen Anteile der Impulse wirken grundsätzlich entgegengerichtet auf ein Massenpunktpaar $P_i(0)$, $P_j(0)$ mit Abstandsbedingung $(i,j,d) \in L$. Damit sind translatorische und rotatorische Einflüsse auf das Massenpunktsystem ausgeschlossen.
- Da die Impulse nur so wirken, dass sie die konstanten Abstandsbedingungen aufrechterhalten, leisten die inneren Kräfte keine Arbeit und führen daher auch nicht zu Störungen der Energiebilanz.

Damit ergeben sich die folgenden leicht geänderten Integrationsformeln:

$$V_i(h) := V_i(0) + \frac{1}{m_i}\, fin_i + \frac{1}{m_i}\int_0^h F_i^0(t)dt$$

$$P_i(h) = P_i(0) + (V_i(0) + \frac{1}{m_i}\, fin_i)h + \frac{1}{m_i}\int_0^h (\int F_i^0(t)dt)dt$$

Dabei besteht $F_i^0(t) := Fex_i(t) + m_i g$ nur noch aus bekannten Kräften, während die Terme $^1/m_i\ fin_i$ als noch unbekannte Geschwindigkeiten aufzufassen sind, die zum Zeitpunkt $t = 0$ eingeführt werden.

Der Gesamt-Impuls fin_i für einen Massenpunkt setzt sich aus Einzel-Impulsen zusammen. Besteht zwischen P_i und P_j eine Abstandsbedingung $(i,j,d) \in L$, so enthält fin_i einen Anteil $c_{ij}\ (P_j(0) - P_i(0))\ /\ |P_j(0) - P_i(0)|$, der von P_i nach P_j wirkt, und fin_j einen umgekehrt wirkenden Anteil $-c_{ij}\ (P_j(0) - P_i(0))\ /\ |P_j(0) - P_i(0)|$, mit zunächst noch unbekannter Stärke c_{ij}. So erhält man den Ansatz:

$$fin_i = \sum_{(i,j,d)\in L} c_{ij}\, \frac{P_j(0) - P_i(0)}{|\,P_j(0) - P_i(0)\,|}$$

Man beachte, dass $|P_j(0) - P_i(0)| = d$ Sollabstand gilt, denn zum Zeitpunkt $t = 0$ liegt ein konsistenter Systemzustand vor.

Wie wird nun vorgegangen, um die noch unbekannten Koeffizienten c_{ij} näherungsweise zu bestimmen? Wir verwenden ein iteratives Korrekturverfahren, das nach folgender Strategie abläuft:

Iterative *fin*-Korrektur

(1) Setze $fin_i := 0$ (für alle i), setze $eps := 0{,}00001 > 0$, setze $Gewicht := 1.0$

(2) Bestimme eine Näherung für $P_i(h)$ und $V_i(h)$:

$$\overline{P}_i(h) = P_i(0) + (V_i(0) + \frac{fin_i}{m_i}) \cdot h + (\frac{Fex_i}{2m_i} + \frac{g}{2}) \cdot h^2$$

$$\overline{V}_i(h) = V_i(0) + \frac{fin_i}{m_i} + \frac{Fex_i}{m_i} h + gh \qquad \text{(Beachte: } fin_i := 0!)$$

(3) Korrekturschleife:

Solange für ein $(i,j,d_{ij}) \in L$ gilt:

$$\| \overline{P}_j(h) - \overline{P}_i(h) | - d_{ij} | \geq eps$$

führe Einzelkorrektur aus:

Bestimme c_{ij} so, dass

$$| \overline{P}_j(h) - \frac{h}{m_j} \mathrm{c_{ij}} \frac{P_j(0) - P_i(0)}{| P_j(0) - P_i(0) |} - (\overline{P}_i(h) + \frac{h}{m_i} \mathrm{c_{ij}} \frac{P_j(0) - P_i(0)}{| P_j(0) - P_i(0) |}) | = d_{ij}$$

eine einfache Näherung hierfür kann wie folgt berechnet werden:

$$c_{ij} = \frac{\left| \overline{P}_j(h) - \overline{P}_i(h) \right| - d_{ij}}{h \cdot (\frac{1}{m_i} + \frac{1}{m_j})}$$

Korrigiere:

$$fin_i := fin_i + \mathrm{c_{ij}} \frac{P_j(0) - P_i(0)}{| P_j(0) - P_i(0) |} \times Gewicht$$

$$fin_j := fin_j - \mathrm{c_{ij}} \frac{P_j(0) - P_i(0)}{| P_j(0) - P_i(0) |} \times Gewicht$$

$$\overline{P}_i(h) := \overline{P}_i(h) + \frac{h}{m_i} \mathrm{c_{ij}} \frac{P_j(0) - P_i(0)}{| P_j(0) - P_i(0) |} \times Gewicht$$

$$\overline{P}_j(h) := \overline{P}_j(h) - \frac{h}{m_j} \mathrm{c_{ij}} \frac{P_j(0) - P_i(0)}{| P_j(0) - P_i(0) |} \times Gewicht$$

$$\overline{V}_i(h) := \overline{V}_i(h) + \frac{1}{m_i} \mathrm{c_{ij}} \frac{P_j(0) - P_i(0)}{| P_j(0) - P_i(0) |} \times Gewicht$$

$$\overline{V}_j(h) := \overline{V}_j(h) - \frac{1}{m_j} \mathrm{c_{ij}} \frac{P_j(0) - P_i(0)}{| P_j(0) - P_i(0) |} \times Gewicht$$

Ende der Korrekturschleife.

Mit diesem Korrekturalgorithmus werden schließlich solche *fin*-Werte bestimmt, die dafür sorgen, dass alle Abstandsbedingungen bis auf einen Restabstandsfehler *eps* erfüllt sind. Mit den aufsummierten Werten fin_i können schließlich auch die Integrationen für die Geschwindigkeiten $V_i(h)$ durchgeführt werden.

Das beschriebene Verfahren ist außerordentlich durchsichtig und einfach zu implementieren. Es hat bei den von uns bisher durchgerechneten Mechanik-Modellen stets problemlos funktioniert, wobei sich aber die mechanische Schwierigkeit eines Modells in unterschiedlichem Korrekturaufwand niederschlägt. In einfachen Fällen wurden 2 bis 5 Korrekturdurchgänge durch die Liste L beobachtet, bei komplizierten Modellen wurden 10 bis 20 und gelegentlich sogar noch mehr Durchläufe registriert. In einigen Fällen reduziert eine Über-Korrektur mit Werten vom *Gewicht* := 1.4 die Zahl der Durchläufe erheblich.

Das vorgestellte Korrekturverfahren löst im Prinzip ein nichtlineares Gleichungssystem für die Koeffizienten c_{ij}. Dabei ergibt sich für jede Einzelbedingung $(i,j,d) \in L$ eine quadratische Form in den c_{ij}. Wir haben bisher nicht untersucht, ob sich solche Gleichungssysteme effizienter als mit unserer iterativen Methode lösen lassen. Möglicherweise kann auch der Werteverlauf der c_{ij} über mehrer h-Intervalle genutzt werden, um gute a-priori Annäherungen vorherzusagen. Hier bleibt also noch Raum für mancherlei Verbesserungen.

Geschwindigkeitskorrektur

Bezüglich der Geschwindigkeitsentwicklung $V_i(0) \rightarrow V_i(h)$ ist das vorgestellte *Fin*-Korrekturverfahren nahezu blind, denn die Zielwerte $V_i(h)$ gehen an keiner Stelle in die *Fin*-Korrektur ein. Die nur am Anfang des Integrationsintervalls einwirkenden *Fin*-Kraftimpulse führen bei $V_i(h)$ zu Integrationswerten, die nicht korrekt sind, weil die Relativgeschwindigkeiten von starr verbundenen Massenpunkten nicht verschwinden, wie es sein sollte. Daher müssen zum Zeitpunkt $t = h$ die Geschwindigkeiten so korrigiert werden, dass die Relativgeschwindigkeiten verschwinden. Die Bedingung hierfür lautet für jedes $(i,j,d) \in L$:

$$(P_j(h) - P_i(h))\,(V_j(h) - V_i(h)) = 0 \qquad (1)$$

Wie erreichen wir das mit kinematisch-dynamisch neutralen Eingriffen in das Massenpunktsystem?

Wir bestimmen am Ende des h-Intervalls analog wie bei *Fin* Impulskräfte, die die Geschwindigkeitsvektoren sofort verändern, so dass alle Gleichungen (1) erfüllt sind. Gesucht werden also auch hier entgegengesetzte Impulskräftepaare, die die $V(h)$-Werte im gewünschten Sinn verändern. Wie bei der *Fin*-Korrektur wird auch diese *V*-Korrektur mit einem ganz ähnlichen Näherungsverfahren vorgenommen. Mathematisch gesehen lösen wir hier jedoch nur ein lineares Gleichungssystem, wie man der Bedingung (1) bei näherem hinschauen sofort ansieht.

Welche Änderungen ergeben sich bei längeren Simulationen, wenn mit oder ohne Geschwindigkeitskorrektur gerechnet wird? Das Ergebnis ist frappierend: Die Geschwindigkeitskorrektur hat keinerlei Einfluss auf die Punkt-Ort-Ergebnisse und das erklärt sich wie folgt: Die *Fin*-Korrektur im nächsten Iterationsschritt findet unmittelbar nach der *V*-Korrektur des vorherigen Iterationsintervalls statt und verhält sich offensichtlich so, dass sie die *V*-Korrektur wieder völlig zu Disposition stellt. Wer sich also nur für die Orte der Massenpunkte interessiert, kann getrost auf den zusätzlichen Aufwand der Korrektur der Relativgeschwindigkeiten verzichten. Nur wenn genauere Werte für die Punktgeschwindigkeiten benötigt werden, ist es also erforderlich, die *V*-Korrektur durchzuführen.

4 Erweiterungen der Punkt-Abstands-Bedingungen

Bereits mit den oben eingeführten Konstant-Abstandsbedingungen der Gestalt (i,j,d), die in der Liste L zusammengefasst sind, lassen sich Starrkörper, Gelenke zwischen ihnen, Kugelgelenke, Kreisel usw. modellieren. Das vorgestellte numerische Verfahren kann jedoch auch auf weitere Typen von Abstandsbedingungen erweitert werden, z.B.:

- Der frei bewegliche Massenpunkt P_3 muss stets auf der Verbindungsstrecke zwischen den Punkten P_1 und P_2 liegen. Das entspricht einer Schienenführung für P_3.
- Der frei bewegliche Massenpunkt P_3 muss sich stets innerhalb des ebenen räumlichen Polygons bewegen, das von drei oder mehr Massenpunkten aufgespannt wir, die einen Starrkörper bilden. Dies entspricht z.B. einer rollenden Kugel auf einer Ebene, die stets den Abstand null zur Ebene haben muss.

Auch frei bewegliche Massenpunktsysteme, die bestimmte Hindernisse wie Wände, Böden usw. nicht durchdringen können, sind mit oder ohne Stoßimpuls-Weitergabe simulierbar. Bemerkenswert ist vor allem, dass das vorgestellte, rein an Kraftsimulation orientierte Verfahren auch Kollisionssituationen mechanisch korrekt handhaben sollte.

5 Ergebnisse der experimentellen Simulationsrechnungen

Das oben vorgestellte Simulationsverfahren wurde an einigen einfachen mechanischen Modellen erprobt. Die Vorgabe bzw. Spezifikation der Modelle ist sehr einfach, da nur die Punkte $P_i(0)$, die aktuellen Startgeschwindigkeiten $V_i(0)$ (in der Regel 0), die Massen m_i und die Abstandsbedingungen $(i,j,d) \in L$ vorzugeben sind, um die Simulation ohne sonstige Vorverarbeitung starten zu können. In einigen Fällen wurden noch Anschubkräfte Fex_i benötigt, um die ruhenden Modelle in Bewegung zu versetzen.

5.1 Mathematisches Pendel von 1 m Länge

Das 1-m-Pendel wurde mit einem maximalen Ausschlagwinkel von 5° simuliert, um die durch die Simulation gewonnenen Schwingungszeiten mit der exakten mechanischen Theorie zu vergleichen.

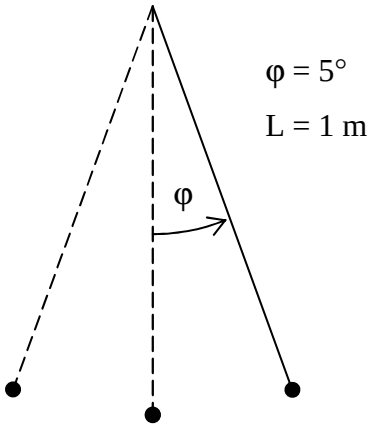

Abbildung 5.1-1: Mathematisches Pendel

Bei der Erdbeschleunigung von g = 9.81 m/s² ergibt sich ohne Korrektur, also bei sehr kleinem Ausschlagwinkel, die Schwingungszeit von T_0 = 2.0060666807 s, während sich beim 5°-Ausschlag aus der Theorie die Schwingungszeit von T_5 = 2.0070219145 s ergibt.

Die Rechnung über 1000 Iterationen (Gesamtzeit 20.070219 s, also $10 \cdot T_5$) ergab sich in der Simulation eine Schwingungszeit von T_{sim} = 2.006687, d.h. die Simulation läuft insgesamt etwa um 0.0166 % zu schnell ab.

Diese kleine Differenz zu T_5 ist auf den Fehler durch die Schrittweite zurückzuführen, denn bei einer 10 mal kleineren Schrittweite ergab sich T_{sim} = 2.00701857, also eine um nur noch rund $3.3 \cdot 10^{-6}$ % schnellere Simulation. Bei entsprechender Verringerung der Schrittweite h konvergiert unser Verfahren also gegen die exakte Lösung. Es wurde beobachtet, dass eine Halbierung der Schrittweite den Zeitfehler ziemlich genau auf ¼ absenkt. Wir vermuten daher, dass der Zeitfehler von der Ordnung $O(h^2)$ ist und dass das eine allgemeine Eigenschaft des Verfahren ist.

Fehleranalyse

Die Pendelsimulation ist das einfachste mechanische Modell, das man sich vorstellen kann, da nur eine Abstandsbedingung zu einem Fixpunkt einzuhalten ist. Daher eignet sie sich auch gut für Fehleranalysen. Genauere Untersuchungen ergaben folgendes: Der Punktort $P_2(h)$ der Pendelmasse wird bestimmt durch vektorielle Streckenaddition der Einzelanteile $V_i(0) \cdot h$, $^1/_m \cdot Fin_2 \cdot h$ und $½ \cdot g \cdot h^2$ auf die Ausgangspunktlagen $P_2(0)$. Der Punkt wird dabei nicht auf der Kreisbahn bewegt, wie es sein müsste, sondern abgesehen von dem kleinen Wurfparabeleffekt durch g auf der direkten Verbindungsstrecke von $P_2(0)$ nach $P_2(h)$. $P_2(h)$ erfüllt wieder die Abstandsbedingungen zum Aufhängepunkt P_1. Betrachtet man nun die Energiebilanz $E_{kin} + E_{pot} = 0$ im Punkt $P_2(0)$ und $P_2(h)$, so stellt man ein kleines Defizit fest. Das bedeutet, dass die Geschwindigkeit (Tangentialgeschwindigkeit ermittelt mit V-Korrektur) in $P_2(h)$ etwas zu niedrig ist. Um den Fehler auszugleichen, müsste $P_2(h)$ auf der Soll-Kreisbahn eine winzige Strecke zurückversetzt werden, so dass die dann etwas größere potentielle Energie E_{pot} das Energiedefizit ausgleicht. Den beschriebenen Fehler

bezeichnen wir als Kurvaturfehler. Er ist umso größer, je größer die Differenz zwischen der Sehnenlänge $P_2(h)$ - $P_2(0)$ von der Länge des Kreisbogens zwischen $P_2(0)$ und $P_2(h)$ ausfällt.

Bei der Pendelsimulation beobachten wir bei der Schrittweite h = 0.02 s ein Gesamt-Energiedefizit von rd. 0.1 % bezogen auf den jeweiligen Wert von E_{kin}, bei h = 0.0025 s, also bei $^1/_8$ der Ausgangsschrittweite, waren es nur noch 0.0015 %, eine Halbierung der Schrittweite führt also ebenfalls zu ¼ Energiedefizit. Dieses Energiedefizit gleicht sich stets wieder aus, wenn das Pendel in oberen Umkehrpunkten (E_{kin} = 0) ankommt und beeinträchtigt auch bei langen Simulationen in keiner Weise die konstante Schwingungszeit der Simulation.

Auch bei den anderen durchsimulierten Modellen wurden analoge Energiedefizite festgestellt, die aber stets transient sind und sich längerfristig keineswegs aufsummieren. Diese winzigen Energiedefizite sind also eine inhärente Eigenschaft unseres Rechenverfahrens und werden nicht durch Rundungsfehler oder zu klein bemessene Werte für *eps* verursacht. Ob es möglich ist, eine Restkorrektur für den Kurvaturfehler in das Verfahren einzubeziehen, konnte noch nicht näher untersucht werden.

5.2 Würfel frei im schwerelosen Raum fliegend

Zum Zeitpunkt $t = 0$ befindet sich der Würfel (siehe Abb. 5.2-1) mit den 8 Eckpunkt-Koordinaten ($\pm 1,\pm 1,\pm 1$) (+- in Ruhe. Die 8 Massenpunkte haben jeweils die Masse 1 kg, wir verwenden das MKS-Maßsystem. Der Würfel hat also eine Kantenlänge von 2 m. In diesem Zustand ist der Würfel wegen fehlender Schwerkraft frei von inneren Kräften.

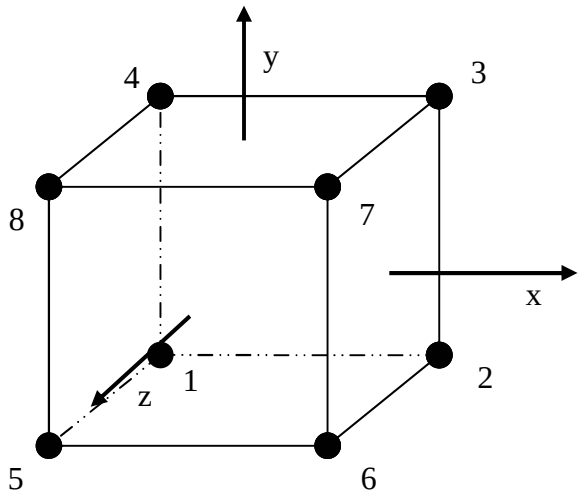

Abbildung 5.2-1: Würfel

Der Würfel wird nun durch eine kurze, von außen auf die Eckpunkte 6 und 7 wirkende Anschubkraft in Richtung (-1, 0, 0) in Bewegung gesetzt und fliegt dann rotierend in die Richtung der Kraftvektoren, also in Richtung der negativen *x*-Achse. Der Schwerpunkt bewegt sich exakt auf der *x*-Achse. Mit dem *h*-Zeitintervall von 0.16 s pro Iteration wurde nun für einen Gesamtzeitraum von 160 s die Bewegung des Würfels simuliert. Der Würfel verhielt sich genau so, wie es die Gesetze der Mechanik verlangen. Er legte 640 m zurück und drehte sich in jedem *h*-Intervall von 0.16 s mit dem Winkel 20.092 Grad um die senkrechte Drehachse. Die Eckpunkte des Würfels werden durch die iterative *Fin*-Korrektur auf den bei diesem Starrkörper vorgeschriebenen Soll-Abständen gehalten, dabei wurden im numerischen Verfahren Toleranzen von maximal 0.00001 m erlaubt. Bemerkenswert ist nun, dass selbst bei den durch die Rotation recht starken Fliehkräften die *Fin*-Korrektur perfekt funktioniert: Ein Massenpunkt bewegt sich in einem *h*-Intervall ohne *Fin*-Kräfte tangential zur Kreisbahn, und zwar eine Strecke von ca. 0.5173 m weit. Dann beträgt der Abstand zur Soll-Kreisbahn bereits 0.0916 m. Dieser Abstand muss nun durch die *Fin*-Korrektur wieder rückgängig gemacht werden, ohne die innermechanischen Eigenschaften des Massenpunktsystems zu beeinflussen. Dies gelingt mit bemerkenswerter Präzision, wie durch folgende Zahlen belegt wird: Während der 1000 Iterationen betrug der Drehwinkel jeweils genau 20.09213 Grad und die kinetische Energie 100.88887682.

Die Werte stimmten in Rahmen der ausgedruckten Stellen perfekt überein. Es ergibt sich also keinerlei Veränderung der Energiebilanz. Van Overveld und Barenbruck [vOB95] haben einen ähnlichen Ansatz wie wir verfolgt und bestimmen die inneren

Kräfte mit einem Newton-Verfahren und die Vorwärtsintegration mit einem einfachen Euler-Verfahren. Sie simulieren den gleichen Massenpunktwürfel wie wir und stellen dann fest, dass bei ihnen die Energie auf vier Stellen genau konserviert wird, was also verglichen mit unseren Ergebnissen markant schlechter ist. Vermutlich ist dies darauf zurückzuführen, dass sie die *Fin*-Kräfte im ganzen *h*-Intervall wirken lassen und dass sich dadurch kleine Fehler einschleichen, die sich aber bei der relativ gleichförmigen Würfelbewegung immer wieder kompensieren.

Wenn wir unsere Simulation der Würfelbewegung ohne Geschwindigkeitskorrektur durchführen, werden über die 1000 Iterationsschritte hinweg die gleichen Punktorte $P_i(t)$ berechnet wie mit *V*-Korrektur, allerdings mit kleinen Koordinaten-Differenzen in der 5. Dezimalstelle nach dem Komma, was vermutlich durch den *eps*-Wert von 0.00001 verursacht wird. Erstaunlich ist dabei die deutlich höhere kinetische Energie von 102.03840430, die zwar konstant bleibt, aber durch die nicht korrigierten Relativgeschwindigkeiten bedingt ist. Die *V*-Korrektur eliminiert diese zusätzliche Energie.

Während die translatorische Drehbewegung des Würfels im Raum scheinbar völlig korrekt verläuft, kann man sich fragen, was aus dem Kurvaturfehler geworden ist, den wir beim mathematischen Pendel beschrieben haben. Er wirkt sich beim Würfel so aus, dass die Geschwindigkeit der Eckpunkte ebenfalls etwas zu gering ist, wenn die Punkte auf der korrekten Kreisbahn laufen müssten. Auch hier müssten also, um den Kurvaturfehler zu kompensieren, die Eckpunkte auf ihrer Kreisbahn nach jeder *h*-Iteration ein Stück zurück versetzt werden. Bei $h = 0.04$ um 2.08 %, bei $h = 0.02$ um 0.5% und bei $h = 0.000625$ noch um $5.4 \cdot 10^{-4}$ %, also mit der gleichen Gesetzmäßigkeit wie beim Pendel. Eine Veränderung der Punktgeschwindigkeiten verbietet sich, um die Energiebilanz nicht zu stören.

Im Falle der Würfelsimulation wurde noch ein weiterer überraschender Effekt beobachtet: Wurde die Rechnung mit dem Gleitpunkt-Zahlenformat „real“, also mit ca. 7 Dezimalstellen Genauigkeit durchgeführt, so verlief die Rechnung zunächst korrekt, um dann ab Iterationsschritt 202 (also bei $t = 32.32$ s) in ein Zustand zu geraten, wo die iterativen *Fin*-Korrekturen plötzlich stets die Höchstzahl von 500 Iterationen erreichte und ohne Befriedigung der vorgeschriebenen Toleranz von *eps* = 0.00001 m abgebrochen werden musste, was die Numerik selbständig erledigt. Die Ursache dieses zunächst mysteriösen Verhaltens liegt in der beschränkten Genauigkeit von ca. 7 Stellen. Denn der Würfel bewegt sich vom Nullpunkt des Koordinatensystems weg, die Zahlenwerte der Koordinaten werden immer größer was schließlich dazu führt, dass die Abfrage Punktabstandsdifferenz < *eps* vollständig den Rundungszufälligkeiten zum Opfer fällt: Bei Iteration Nr. 202 liegt der Würfelschwerpunkt bei der *x*-Koordinate – 128.96 m. Durch Rechnen mit doppelter Genauigkeit (ca. 13 Dezimalstellen) wird das Problem selbstverständlich behoben, aber auch dadurch, dass man das Toleranzmaß *eps* = 0.00001 vergrößert. Nach diesen Erfahrungen wurden die Testrechnungen mit doppelter Genauigkeit durchgeführt (64-Bit-Gleitpunktzahlen).

5.3 Starrarmiges 3-Massen-Pendel

Das Modell besteht aus 4 Massenpunkten. Punkt 1 ist der unbeweglich im Raum fixierte Aufhängepunkt, die Punkte 2, 3, 4 werden jeweils starr auf 1 m Abstand vom vorhergehenden Punkt gehalten. Die Masse der Punkte 2, 3, 4 ist jeweils 2 kg, Erdbeschleunigung 9.81 m/s². Zum Zeitpunkt $t = 0$ befinden sich die Punkte waagerecht angeordnet auf den Positionen $P_1 = (0,0)$, $P_2 = (1,0)$, $P_3 = (2,0)$ und $P_4 = (3,0)$ der x-Achse in Ruhe und werden dann freigegeben, so dass sie nach unten fallen können.

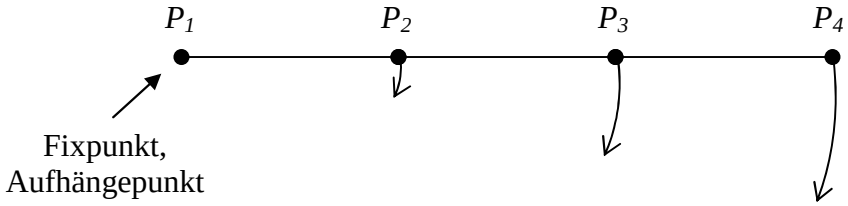

Abbildung 5.3-1: Starrarmiges 3-fach-Pendel

Die numerische Rechnung wurde mit der Zeit-Schrittweite von $h = 0.01$ s durchgeführt, und zwar mit insgesamt 2000 Iterationen. Das sich ergebende Schwingungsverhalten ist in Abbildung 5.3-2 graphisch veranschaulicht. Bei diesem 2-dimensionalen Modell, welches ohne Kollisionsrechnung durchgeführt wurde, konnte folgendes beobachtet werden:

(1) Das Modell gerät mit zunehmender Schwingungsdauer in einen mehr oder weniger chaotischen Bewegungszustand, vgl. z.B. Abbildung 5.3-2.

(2) Die Energiebilanz $E_{kin} + E_{pot} = 0$, die im Anfangszustand bei $t = 0$ herrscht, wird im Verlaufe der Iterationen leicht gestört, allerdings bedingt durch den unregelmäßigen Bewegungsverlauf auch entsprechend unregelmäßig. Schön zu sehen ist dies im Umfeld von Iteration 778, wo durch ruckartige Pendelstreckung der Extremwert $E_{kin} + E_p = -1.345$ erreicht wird, siehe Abbildung 5.3-2.

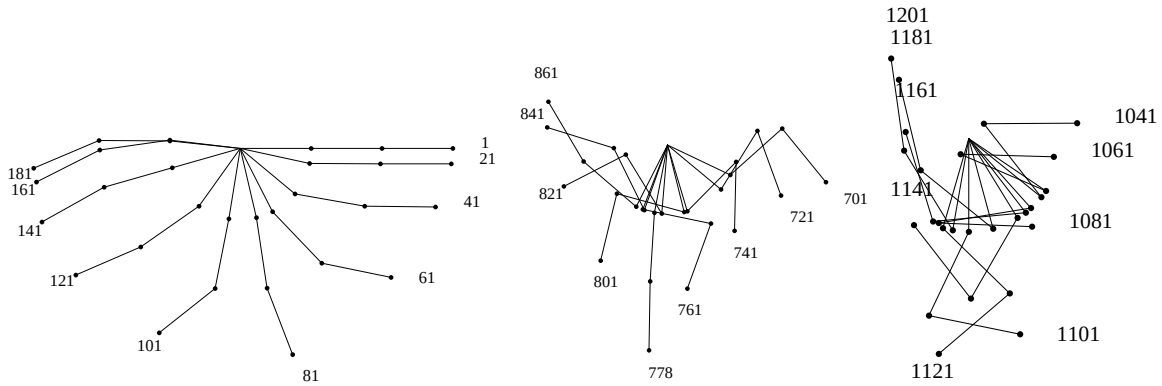

Abbildung 5.3-2: Starrarmiges 3-fach-Pendel im Zustand der Iterationen 1-181, 701–861, und 1041–1201

Erstaunlicherweise sind auch solche Energieverluste transient und verschwinden nach wenigen Iterationen wieder vollständig aus dem System. Der Wert von $E_{kin} + E_{pot}$ ist durch die geschilderten Effekte in geringer Größenordnung negativ, über 2000 Iterationen wurde im Mittel ein Wert von 0.038% von E_{kin} ermittelt. Stärkere negative Energieverluste gleichen sich später wieder aus.

5.4 Fadenpendel mit 3 Massen

Das Modell stimmt fast vollständig mit dem in 5.3 beschriebenen starrarmigen 3-fach-Massen-Pendel überein mit einer einzigen markanten Differenz: Die Massenpunkte P_1, P_2, P_3, P_4 sind durch masselose Fäden von jeweils 1 m Länge verbunden. Die *Fin*-Korrektur erlaubt nun, dass verbundene Massen sich zwar näher als 1 m kommen dürfen, der Maximalabstand von 1 m darf aber nicht überschritten werden. Die Abstandsbedingungen lauten also

$$|P_i - P_{i+1}| \leq 1 \quad \text{für } i = 1, 2, 3$$

Selbstverständlich treten bei der Simulation nach einiger Zeit immer wieder Situationen auf, wo die realen Abstände von Massenpunkten erheblich kleiner als 1 m werden. Zwei dieser interessanten Konfigurationen sind in der Abbildung 5.4-1 dokumentiert. Bei einer ganzen Reihe von Iterationsschritten, insbesondere bei den hier näher betrachteten Schritten 1269 und 1488 tritt das Ereignis des Fadenspannens ein, wo also die Auseinanderbewegung zweier Massenpunkte abrupt bei Erreichen des Abstands 1 m durch *Fin*-Kräfte gestoppt wird. Dabei werden erhebliche Anteile der kinetischen Energie des Gesamtsystems absorbiert und zwar bei Iterationsschritt 1269 ca. 2.9 und bei 1488 ca. 6.5 kgm²/s². Insgesamt sinkt die maximale beobachtete kinetische Energie von 116.88 am Anfang der Simulation (beim ersten Durchschwingen) bis auf den Maximalwert von ca. 23.91 nach 5000 Iterationen. Das System büßt also durch die Ereignisse des Fadenspannens erheblich an kinetischer Energie ein und nähert sich also auch ohne Reibungseffekte einem Bewegungszustand, bei dem die Abstände der Massenpunkte permanent auf 1 m verbleiben. Dies wird von Iterationen Nr. 3063 bis 5000 beobachtet. Dass Fadenspannereignisse kinetische Energie schlucken und in der physikalischen Realität in Reiß- oder Verformungsarbeit umsetzen, macht man sich ganz einfach klar, indem man eine an einem längeren Faden befestigte Kugel fallen lässt. Sie sollte beim Spannen des (unelastischen) Fadens sofort zur Ruhe kommen, wenn wir keine Federkräfte oder Fadenreißen einrechnen.

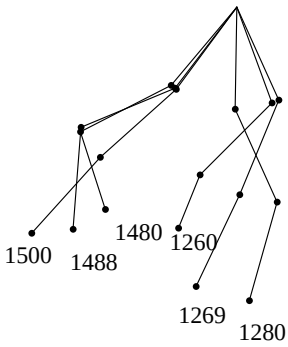

Abbildung 5.4-1: 2 Fälle (Iterationen 1488 und 1269) von Fadenspannen.

In Abb. 5.4.-1 erkennt man deutlich, dass der Abstand zwischen P_3 und P_4 zunächst deutlich kleiner als 1 m ist (1480 und 1260) und dann genau bei der Iteration 1488 und 1269 die Länge 1 m erreicht und damit das energieschluckende Ereignis der Fadenspannung auslöst.

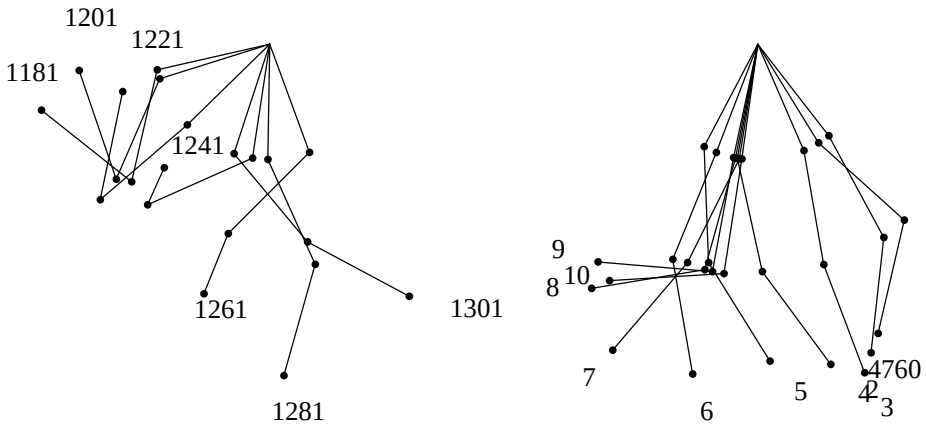

Abb. 5.4-2: Bewegung des 3-teiligen Fadenpendels ab Iteration 1181 bzw. 4760. Die geringe im System verbliebene kinetische Energie hat unter anderem zur Folge, dass ab Iteration 3063 die Massenpunkte stets den Abstand 1 m behalten.

5.5 3-fach Starrpendel in Gleitschiene beweglich aufgehängt

Durch diese Simulation soll demonstriert werden, dass man einen Massenpunkt auch zwanghaft nur auf einer Geraden beweglich leicht in das *Fin*-Korrektursystem integrieren kann. Wir hängen das starre 3-fach-Pendel frei beweglich auf der *x*-Achse auf, der jetzt nicht mehr fixe Punkt P_1 erhält wie die anderen Punkte die Masse 2 kg. Das Pendel wird auf der *x*-Achse nach rechts ausgerichtet (P_1 im Nullpunkt des Koordinatensystems), unter Schwerkraft losgelassen. Wie in Abbildung 5.5-1 zu sehen ist, bewegt sich P_1 auf der *x*-Achse nach rechts, was gleich zu Pendel-Überschlägen führt. Die Bewegung wird insgesamt schnell chaotisch. Da das auf der *x*-Achse gleitend aufgehängte System zum Zeitpunkt $t = 0$ den Schwerpunkt auf der Position $x = 1.5$ hat, bleibt dieser Wert wegen fehlender Verschiebungskräfte bei der Simulation konstant.

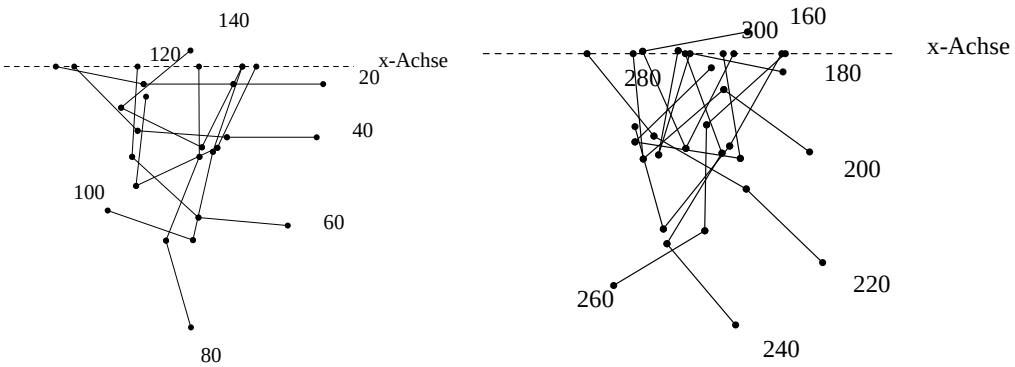

Abbildung 5.5-1: 3-fach Starrpendel

5.6 Kinematisch geschlossene Kette

Hier simulieren wir eine fallende Gliederkette, die an beiden Enden an unbeweglichen Fixpunkten befestigt ist. Das Modell ist wie folgt aufgebaut:

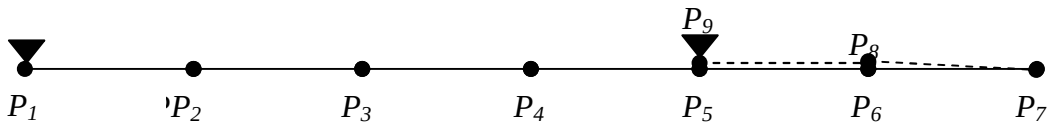

Abbildung 5.6-1: Gliederkette im Ruhezustand (t = 0)

Die Kette besteht aus den Aufhängepunkten P_1 und P_9, deren Masse irrelevant ist, da sie nicht bewegt werden, und den restlichen beweglichen Massenpunkten P_2,..., P_8 mit Masse von jeweils 1 kg und Abstandsbedingung $|P_{i+1} - P_i| = 1$ für i = 1,...,8. Zum Zeitpunkt t = 0 befindet sich die Kette in Ruhe und wird dann unter dem Einfluss der Schwerkraft losgelassen. Iteriert wird mit h = 0.01 s. Der Bewegungsverlauf am Anfang und aus einer Situation mit hoher kinetischer Energie (Iteration 980 ff) ist aus Abbildung 5.6-2 ersichtlich.

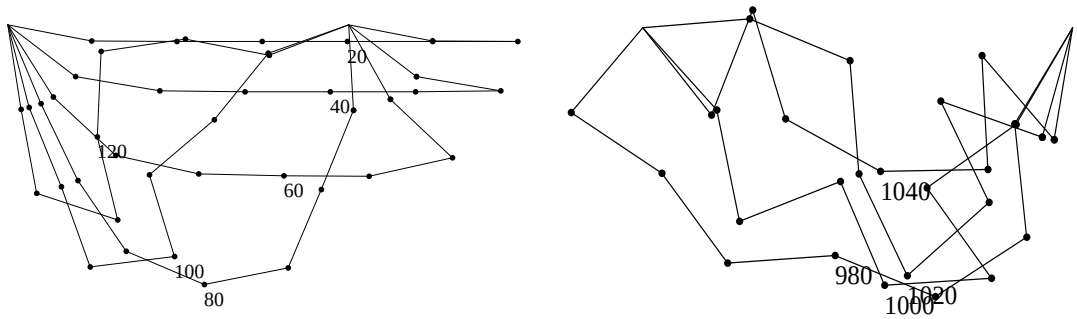

Abbildung 5.6-2: Gliederkette im Zustand 20 ff und 980 ff

Wie bei diesem Modell nicht anders zu erwarten ist, ergeben sich auch hier kleine Energieverluste bezüglich der Anfangssituation $E_{kin} + E_{pot} = 0$, die sich aber wie bei den anderen Modellen nicht aufsummieren, sondern nur etwas pulsieren. Über die durchsimulierten 1200 Iterationen mit einer Realzeit von 12 s ergibt sich das mittlere prozentuale Energiedefizit zu 0.089% bezogen auf die jeweilige kinetische Energie, die am Anfang einen Spitzenwert von 140 erreichen kann, später aber bedingt durch chaotische Bewegungselemente einen Wert von 115 nicht mehr übersteigt.

Das kinematisch geschlossene Modell erfordert von den Korrekturiterationen für *Fin* und der Relativgeschwindigkeitsangleichung in mechanisch kritischen Situationen deutlich mehr Aufwand als bei den baum-strukturierten 3-Pendel-Elementen. (In dieser Hinsicht sind die Starrkörper Würfel und Kreisel völlig problemlos). Ganz am Anfang der Iterationen ist die *V*-Korrektur durch Verklemmungseffekte im Modell während dreier Iterationsschritte gar nicht in der Lage, die Relativgeschwindigkeit innerhalb der Maximalanzahl von 200 iterativen Durchgängen zu korrigieren. Später ist dies dann wieder mit durchschnittlich 10 Iterationen möglich. Für unsere Rechenmethode ist es

aber völlig unerheblich, ob ein baumstrukturiertes Mehrkörpersystem oder ein stark zyklisch geschlossenes System vorliegt. Allerdings muss man bei geschlossenen Systemen oft mit den oben schon angesprochenen Verklemmungssituationen und Kraftasymptoten, also mechanischen Sonderbedingungen, rechnen.

5.7 Kreisel auf einem Punkt stehend

Der Kreisel wurde wie aus Zeichnung 5.7-1 ersichtlich als Massenpunktsystem modelliert. Es handelt sich um eine auf der Spitze stehende Pyramide.

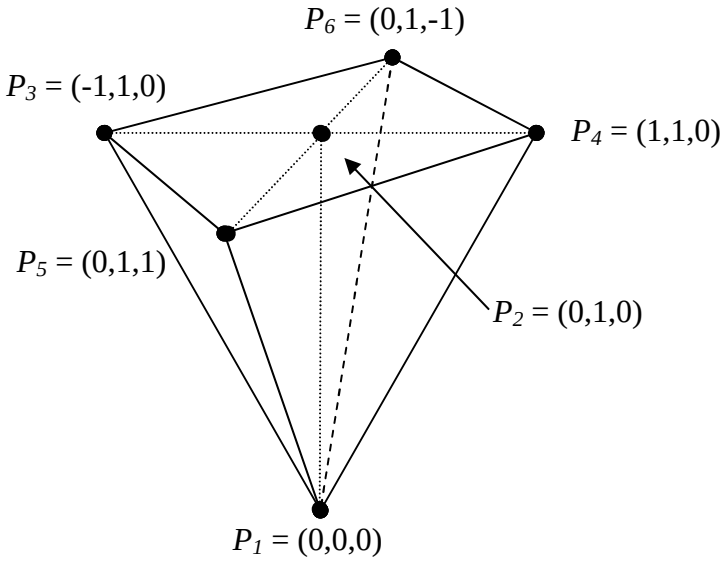

Abbildung 5.7-1: Kreiselmodell, P_1 fixiert, der Pyramidenkörper frei beweglich.

P_2 ist mechanisch überflüssig und dient als Datenpunkt, da an ihm die Kreiselachse bequem abgelesen werden kann. Der Punkt P_1 im Nullpunkt des Koordinatensystems ist ein feststehender unbeweglicher Aufhängepunkt. Unser Kreisel ist ziemlich groß, denn der Abstand gegenüberliegender Außen-Massenpunkte ist 2 m. Die Masse der Massenpunkte ist jeweils 1 kg. Um den Kreisel aufrechtstehend in Rotationsbewegung zu versetzen, wurden an den Außenpunkten für den Zeitraum h = 0.005 s extern einwirkende Kräftevektoren (*Fex*-Kräfte) eingesetzt:

P_3: (0,0,-6000), P_4: (0,0,4000), P_5: (-4000,0,0), P_6: (+4000,0,0)

Durch die bei P_3 unsymmetrisch eingesetzte Zusatzkraft von -2000 wird der Kreisel aus der exakten Vertikal-Lage herausgedrückt, durch die restlichen rotatorisch ansetzenden Kräfte in Umdrehung versetzt. Er führt dann die aus der Mechanik bekannten schlingernden Bewegungen der Zentralachse aus. Er rotiert bei maximaler Schräglage der Achse, d.h. bei etwas geringerer potentiellen Energie, mit 3.5944 Umdrehungen/Sekunde, wenn die Achse senkrecht aufgestellt ist, ergeben sich nur 3.5939 Umdrehungen/Sekunde. Die maximale Neigung des Achswinkel gegenüber der Senkrechten beträgt 12.83°. Wenn man den Zentralpunkt P_2 auf die x-z-Achse projiziert und seine Bewegung über 2233 Iterationen bzw. die Zeit von 11.6 s verfolgt, so ergibt

sich das in Abb. 5.7-2 dargestellte Projektionsmuster des Zentral-Punktes P_2. Bedingt durch das Anstoßen in senkrechter Stellung kommt die Zentralachse auch immer wieder in die senkrechte Lage.

Wie bei allen in Schwerkraft aufgehängten Modellen ergeben sich auch in diesem Fall leichte Verringerungen der ursprünglichen Systemenergie. Diese sind aber im Vergleich zu den 3-armigen Pendeln sehr gering und sind bei schrägen Achslagen fast 100 x größer als bei senkrechter Achsstellung, gleichen sich aber stets wieder aus.

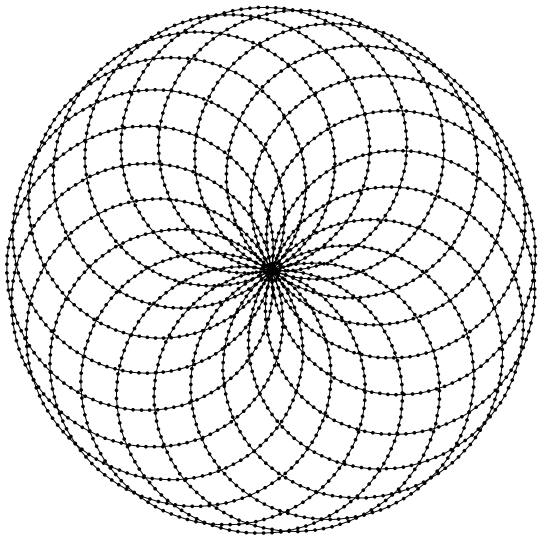

Abbildung 5.7-2: Bewegung der Kreisel-Zentralachse dargestellt durch Parallelprojektion des Punktes P_2 auf die x-z-Ebene

6 Schlussanmerkungen

Die heute weit verbreiteten Softwaresysteme für die mechanische Simulation wie z.B. Adams, Mesa Verde, SD/FAST und viele weitere basieren unseres Wissens alle auf einem Differentialgleichungsansatz. Zwar hat man sich bemüht, die Benutzerschnittstelle so zu gestalten, dass auch wenig mit der Theorie Vertraute die Systeme nutzen können. Die Komplexität der Algorithmen bleibt aber dennoch bestehen und macht sie z.B. als Submodul für VR-System ungeeignet. Das erkennt man z.B. deutlich, wenn man sich die Benutzerhandbücher anschaut, z.B. jenes für SD/FAST der Symbolic Dynamic Inc.

Wir haben uns daher das Ziel gesetzt, die dynamische Simulation für Massenpunktsysteme auf ganz direktem Weg und ohne Rückgriff auf Systeme von Differentialgleichungen durchzuführen. Wir verfolgen dabei die Strategie, den einzelnen Massenpunkt als ein mit seiner mechanischen Umgebung interagierendes Subsystem aufzufassen. Das hat insbesondere den Vorteil, dass Sonderfälle wie bei Kollisionen problemlos in die Simulation einbezogen werden können.

Das Ziel der Einfachheit und Durchsichtigkeit des numerischen Verfahrens wurde zweifellos erreicht. Ein spezieller Trick, die inneren Kräfte *Fin* als Impulskräfte zu behandeln, brachte die gewünschten Erfolge. Wir konnten auch feststellen, dass durch Verringerung der *t*-Schrittweite *h* eine beliebig hohe Genauigkeit der Simulation ohne Gesamtenergie-Fading erreichbar ist. Auch über die Natur des Restfehlers, den wir als Kurvaturfehler bezeichnet haben, konnten wir vorläufige Aussagen machen.

Allerdings steht eine genauere Analyse noch aus und es ist noch völlig offen, ob die bei VR-Anwendungen und bei graphisch dargestellten Simulationen sicherlich unerheblichen Restfehler durch ein einfaches Verfahren wie z.B. die angesprochene Punkt-Rücksetzung nachkorrigiert werden kann. Ungeklärt ist derzeit auch noch, wie das iterative Verfahren zur Bestimmung der inneren Kräfte *Fin* beschleunigt werden kann. Das *V*-Korrekturverfahren ist demgegenüber weniger bedeutsam, weil es nur dann angewandt werden muss, wenn man für die Massenpunkte korrekte Geschwindigkeitsvektoren benötigt.

Neben den angesprochenen Anwendungen in der Computergraphik betrachten wir unser Simulationsverfahren auch als didaktisch gut geeignet für die akademische Lehre, weil es mit ganz geringem Programmieraufwand verbunden ist und dadurch zum Experimentieren einlädt.

7 Literatur

[De96] O. Deussen: *Untersuchung effizienter Verfahren zur Bewegungssimulation deformierbarer Körper.* Dissertation, VDI Verlag Fortschrittberichte Reihe 20, Nr. 215, Düsseldorf 1996.

[IC87] P. M. Isaacs, M. F. Cohen: *Controlling dynamic simulation with kinematic constraints, behavior functions and inverse dynamics.* ACM Computer Graphics, Vol. 21, 215-224.

[Kuh97] C. Kuhn: *Modellbildung und Simulation deformierbarer Objekte zur Entwiklung einer interaktiven Trainingsumgebung für die minimal-invasive Chirurgie.* Dissertation, Universität Karlsruhe 1997.

[LPC95] J. Louchet, X. Provost, D. Crochemore: *Evolutionary identification of cloth animation models.* In: D. Terzopoulos und D. Thalmann, Hrsg.,Computer and Simulation '95. Springer-Verlag, 1995.

[vOB95] K. van Overveld und B. Barenburg: *All you need is force: A Constrained approach for rigid body dynamics in computer animation.* In: D. Terzopoulos und D. Thalmann, Hrsg., Computer and Simulation '95. Springer-Verlag, 1995.

[RSch] R. E. Roberson, R Schwertassek: *Dynamics of multibody systems.* Springer-Verlag 1988.

[W87] J. Wilhelms: *Using dynamic analysis for realistic animation of articulated bodies.* IEEE Computer Graphics and Applications, June 1987, 12-27.

[Wit77] J. Wittenburg: *Dynamics of systems of rigid bodies.* Leitfäden der angewandten Mathematik und Mechanik, Band 33, B. G. Teubner, Stuttgart 1977.