

Praktikum Effiziente Algorithmen

Fallende Würfel

Miao Wang, Florian Thiel, Robert Richter, Manuel Schulze

Institut für Informatik, Freie Universität Berlin

Zusammenfassung Im Praktikum „Effiziente Algorithmen“ werden Probleme aus der theoretischen Informatik behandelt. Dieses Paper beschreibt die Entwicklung einer Simulation für das Problem der fallenden Würfel.

1 Einleitung

Das Projekt, auf dem diese Arbeit beruht, entstand im „Praktikum Effiziente Algorithmen“ [5] im Sommersemester 2007 am Institut für Informatik an der Freien Universität Berlin [1]. Das Praktikum stand unter der Leitung von Prof. Dr. Günter Rote. Die Aufgabe war es, anhand ausgewählter Beispiele, effiziente Algorithmen zu entwickeln, zu analysieren und zu implementieren.

Bei dem Projekt „Fallende Würfel“ werden, wie in Abbildung 1 angedeutet, Würfel¹ gestapelt um sie dann gemäß den physikalischen Eigenschaften fallen zu lassen.

Für die von uns verwendeten Würfel gelten einige Eigenschaften:

1. die Würfel müssen nicht gleich groß sein,
2. die Würfel können unterschiedliche Gewichte haben und
3. um das Problem einfach zu halten werden die Würfel im zweidimensionalen Raum betrachtet.

Die Aufgabe und das Ziel des Projekts ist es, eine Simulation zu schreiben, die ein Szenario, wie in Abbildung 1 dargestellt, als Eingabe bekommt und den Fall unter Berücksichtigung der physikalischen Bedingungen, simuliert. Dabei sollen möglichst effiziente Algorithmen und Verfahren zum Einsatz kommen. Zusätzlich soll ein Tool bereitgestellt werden um das Ergebnis einer Simulation am Bildschirm darzustellen.

Der Inhalt dieser Arbeit gliedert sich in fünf Teile. In dieser Einleitung wird die grobe Architektur des Systems erläutert. Anschließend wird das zu Grunde liegende mathematische Modell vorgestellt und erklärt. Ein kleines Kapitel ist dem Visualisierungstool gewidmet. Abschließend diskutieren wir die von uns gewonnenen Erkenntnisse.

¹ In Wirklichkeit handelt es sich um Quader. Aber da der Titel des Projekts so lautet, verwenden wir den Begriff „Würfel“.

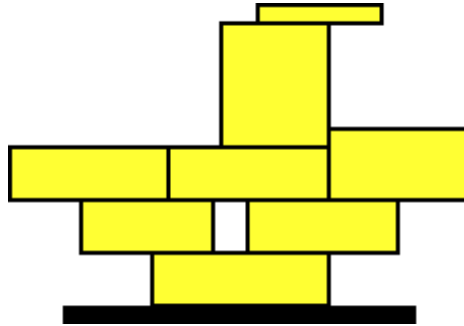


Abbildung 1: Ein Stapel Würfel in der Ausgangsposition

Als Zusammenfassung des Projekts haben wir eine Homepage [6] ins Netz gestellt. Neben allgemeinen Informationen sind dort Beispiele als Java Applets und alle Projektergebnisse als Download verfügbar. Unter den Downloads finden sich auch alle Beispiele, die während der Entwicklung entstanden sind. Die Beispiele liegen in ihrer Ausgangskonfiguration und als Simulationsergebnis vor.

Bei den Beispiellapplets handelt es sich um eine stark vereinfachte Form der Visualisierungskomponente. Jedes der dargestellten Beispiele verwendet das Ergebnis einer Simulation als Eingabe und stellt es entsprechend dar. Die Simulation kann somit für jede Iteration nachvollzogen werden.

2 Systemarchitektur

In diesem Kapitel werden die Hauptkomponenten des Systems vorgestellt.

Die von uns entwickelte Software lässt sich in zwei Komponenten unterteilen:

1. Simulation und
2. Visualisierung.

Wir haben uns für diese Trennung entschieden, weil eine Simulation in Echtzeit nicht möglich ist. In Abbildung 2 ist der schematische Aufbau und der Datenfluss dargestellt. Die Simulation bekommt ein Szenario in Textform als Eingabe und berechnet mit Hilfe von **CGAL** (Computational Geometry Algorithms Library [2]) die einzelnen Simulationsschritte. Die Ausgabe sind die simulierten Schritte (Trace), ebenfalls in Textform. Der Trace kann anschließend in die Visualisierung eingelesen und präsentiert werden.

Das Format für die Simulationseingabe und -ausgabe ist ähnlich. Der einzige Unterschied ist, dass in der Ausgabe alle Simulationsschritte in einem zusätzlichen Datenblock aufgelistet sind. Das allgemeine Format ist wie folgt definiert:

```
[meta]
Anzahl der Würfel in der Simulation (int)
```

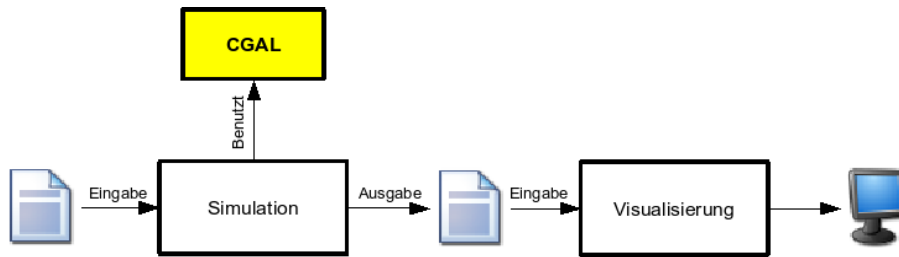


Abbildung 2: Schematischer Aufbau und Datenfluss

```

Verwendeter Zeitabstand in einem Simulationsschritt (double)
Verwendetes Impulsmodell (bool)
Fallbeschleunigung (double)
[table]
# Koordinaten des Tisches im Raum. Jede
# Zeile steht für einen Punkt. Die Reihenfolge
# ist lo, ro, lu, ru. Der Datentyp ist double.
x1 y1
x2 y2
x3 y3
x3 y4
[init]
Simulationsschrittnummer (int)
# Koordinaten, Geschwindigkeiten, Beschleunigungen 1. Würfel.
# Die Reihenfolge ist lo, ro, lu, ru. Der Datentyp ist double.
x1 y1 vx1 vy1 ax1 ay1
x2 y2 vx2 vy2 ax2 ay2
x3 y3 vx3 vy3 ax3 ay3
x4 y4 vx4 vy4 ax4 ay4
Masse des 1. Würfels (double)
# Koordinaten, Geschwindigkeiten, Beschleunigungen 2. Würfel.
...
Masse des 2. Würfels (double)
# usw.
  
```

In der Ausgabe schließt sich an den *[init]* Bereich ein *[data]* Bereich an, der folgendermaßen definiert ist:

```

[meta]
[table]
[init]
[data]
Simulationsschrittnummer (int)
# Koordinaten, Geschwindigkeiten, Beschleunigungen 1. Würfel.
  
```

```

# Format wie im init Block
# Koordinaten, Geschwindigkeiten, Beschleunigungen 2. Würfel.
# Koordinaten, Geschwindigkeiten, Beschleunigungen 3. Würfel.
Simulationsschrittnummer (int)
# usw.

```

3 Das Modell

3.1 Die Zielfunktion

CGAL bietet ein Paket zur Berechnung von konvexen quadratischen Optimierungen, welches im Folgenden als Solver [3] bezeichnet wird. Der Solver erwartet eine Eingabe einer zu minimierenden Zielfunktion in folgender Form:

$$x^T D x + c^T x + c_0 \quad (1)$$

Die einzelnen Komponenten sind:

- x als Vektor der gesuchten n Variablen $(x_0, \dots, x_n - 1)$,
- D als $n \times n$ Matrix der Koeffizienten der quadratischen Variablen,
- c als n -dimensionaler Vektor der Koeffizienten der linearen Variablen und
- c_0 als Konstante.

Zum Beispiel wäre für die Funktion $x^2 + 4y^2 - 32y + 64$ [3] die Formel:

$$\begin{pmatrix} x & y \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 4 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} 0 & -32 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + 64$$

Achtung: In **CGAL** muss $2 \cdot D$ (also $\begin{pmatrix} 2 & 0 \\ 0 & 8 \end{pmatrix}$) anstatt D in die Berechnung eingegeben werden, weil **CGAL** versucht, die allgemeine Form $\frac{1}{2}x^T D x + c^T x + c_0$ zu lösen.

Für unsere Zielfunktion

$$\begin{aligned}
& \sum_{i=1}^n m_i \|a_i - F_i\|^2 \\
&= \sum_{i=1}^n m_i \left\| \begin{pmatrix} a_{xi} \\ a_{yi} \end{pmatrix} - \begin{pmatrix} 0 \\ -g \end{pmatrix} \right\|^2 \\
&= \sum_{i=1}^n m_i \left\| \begin{pmatrix} a_{xi} \\ a_{yi} + g \end{pmatrix} \right\|^2 \\
&= \sum_{i=1}^n m_i (a_{xi}^2 + (a_{yi} + g)^2) \\
&= \sum_{i=1}^n m_i a_{xi}^2 + m_i a_{yi}^2 + 2m_i g a_{yi} + m_i g^2
\end{aligned} \quad (2)$$

lautet dementsprechend die Formel:

$$\begin{aligned}
& (a_{x1} \ a_{y1} \ \cdots \ a_{xn} \ a_{yn}) \begin{pmatrix} m_1 & 0 & \cdots & 0 \\ 0 & m_2 & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & m_n \end{pmatrix} \begin{pmatrix} a_{x1} \\ a_{y1} \\ \vdots \\ a_{xn} \\ a_{yn} \end{pmatrix} \\
& + (0 + 2m_1g \ \cdots \ 0 + 2m_ng) \begin{pmatrix} a_{x1} \\ a_{y1} \\ \vdots \\ a_{xn} \\ a_{yn} \end{pmatrix} + m_1g^2 + \cdots + m_ng^2
\end{aligned} \tag{3}$$

3.2 Die Randbedingungen

Bei den Randbedingungen muss man beachten, dass sie nicht im direkten Verhältnis zur gesuchten Variable a_{xi} oder a_{yi} stehen, sondern Längenrestriktion und Seitenrestriktion von p_{xi} und p_{yi} abhängig sind.

Im Folgenden wird davon ausgegangen, dass p' der Zustand zum Zeitpunkt t darstellt und p den Zustand zum Zeitpunkt $t + 1$.

$$p = p' + v' + \ddot{a} = p' + v' \cdot \Delta t + \frac{1}{2} a \cdot \Delta t^2 \tag{4}$$

Die Längenrestriktion: Demzufolge ist die Längenrestriktion für einen Vektor zwischen p_i und p_j :

$$\langle p_i - p_j, p_i - p_j \rangle = l_{i,j}^2 \tag{5}$$

davon die 2. Ableitung:

$$\begin{aligned}
& \langle a_i - a_j, p_i - p_j \rangle + \|v_i - v_j\|^2 = 0 \\
& \left\langle \begin{pmatrix} a_{xi} - a_{xj} \\ a_{yi} - a_{yj} \end{pmatrix}, \begin{pmatrix} p_{xi} - p_{xj} \\ p_{yi} - p_{yj} \end{pmatrix} \right\rangle + \left\| \begin{pmatrix} v_{xi} - v_{xj} \\ v_{yi} - v_{yj} \end{pmatrix} \right\|^2 = 0
\end{aligned} \tag{6}$$

$$\begin{aligned}
& = a_{xi}(p'_{xi} - p'_{xj}) - a_{xj}(p'_{xi} - p'_{xj}) \\
& + a_{yi}(p'_{yi} - p'_{yj}) - a_{yj}(p'_{yi} - p'_{yj}) \\
& + (v'_{xi} - v'_{xj})^2 + (v'_{yi} - v'_{yj})^2
\end{aligned} \tag{7}$$

Diese Bedingung gilt 5-mal:

- je einmal für jede Quaderseite und
- einmal für eine Diagonale.

Die Seitenrestriktion: Die Anwendung der Seitenrestriktion ist bedingt. Sie tritt dann in Kraft, wenn ein Würfel mit einem anderen kollidiert. Modelliert wird dieses Ereignis durch drei Punkte p_i , p_j und p_k , wobei p_i , p_j dem Würfel W_{ij} und p_k dem Würfel W_k zugeordnet sind, welche ein Dreieck $D \in \{p_i, p_j, p_k\}$ in der Ebene mit dem Flächeninhalt A_D aufspannen. Der Flächeninhalt dieses Dreiecks inklusive seiner Änderung über die Zeit \dot{A}_D zeigt an, ob eine Seitenrestriktion für die drei Punkte notwendig ist.

Wenn $A_D \leq 0$ ist und $\dot{A}_D \leq 0$, dann findet die Seitenrestriktion Anwendung. Anschaulich bedeutet dies, wenn p_k genau auf oder schon hinter der Kante zwischen p_i und p_j liegt² und das auch mit fortschreitender Zeit so bleibt (relative Geschwindigkeitsdifferenz gleich 0) oder sich verstärkt (p_k dringt noch weiter in W_{ij} ein), dann muss dafür gesorgt werden, dass p_k nicht oder nicht weiter in W_{ij} eindringt, indem die 2. Ableitung der Seitenrestriktion, die die Änderung der Punktpositionen bei der Optimierung der Zielfunktion entsprechend beeinflusst, angewendet wird.

Laut [4] ist (auf unsere Syntax angepasst)

$$A_D = \begin{vmatrix} 1 & 1 & 1 \\ p_{xi} & p_{xj} & p_{xk} \\ p_{yi} & p_{yj} & p_{yk} \end{vmatrix} \quad (8)$$

Für $\dot{A}_D \geq 0$ liefert [4] eine alternative Form:

$$\dot{A}_D = \begin{vmatrix} 1 & 1 & 1 \\ p_{xi} & p_{xj} & p_{xk} \\ p_{yi} & p_{yj} & p_{yk} \end{vmatrix}' \geq 0 \iff \langle v_k, n \rangle \geq \alpha \langle v_j, n \rangle \quad (9)$$

Aus dieser Äquivalenz lässt sich schließen

$$\dot{A}_D \leq 0 \iff \langle v_k, n \rangle \leq \alpha \langle v_j, n \rangle \quad (10)$$

Es lässt sich zeigen, dass wenn p_i, p_j und p_k auf einer Geraden liegen (also $A_D = 0$) für den skalaren Wert α mit $0 \leq \alpha \leq 1$ gilt

$$\alpha = \frac{p_{xk} - p_{xi}}{p_{xj} - p_{xi}} = \frac{p_{yk} - p_{yi}}{p_{yj} - p_{yi}} \quad (11)$$

Der Vektor n ist laut [4] gegeben als $(p_j - p_i)^\perp$. Da $(p_j - p_i)$ ein Ortsvektor ist, kann er mittels einer Drehmatrix um den Ursprung gedreht werden. Dabei ergibt sich der Drehwinkel zu $\gamma = \frac{\pi}{2}$. Mit der Drehmatrix

$$R = \begin{pmatrix} \cos\gamma & -\sin\gamma \\ \sin\gamma & \cos\gamma \end{pmatrix} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \quad (12)$$

² Bei kontinuierlicher Zeit, also unendlich kleinen Berechnungsintervallen würde dieser Fall gar nicht auftreten. Da wir das Modell jedoch mit rational quantisierter Zeit berechnen, kann es passieren, dass Würfel von einem Zustand zum nächsten ineinander fallen. Je kleiner die Zeitintervalle, desto kleiner wird dieser Fehlereffekt.

ergibt sich n zu

$$n = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} p_{xj} - p_{xi} \\ p_{yj} - p_{yi} \end{pmatrix} = \begin{pmatrix} -p_{yj} + p_{yi} \\ p_{xj} - p_{xi} \end{pmatrix} \quad (13)$$

Auch für die 2. Ableitung der Seitenrestriktion liefert [4] eine alternative Form:

$$\langle a_k, n \rangle - \alpha \langle a_j, n \rangle - (1 - \alpha) \langle a_i, n \rangle + |v_i, v_j| + |v_j, v_k| + |v_k, v_i| \geq 0 \quad (14)$$

wobei man für $|v_i, v_j|$ auch schreiben kann

$$\begin{vmatrix} \dot{x}_i & \dot{x}_j \\ \dot{y}_i & \dot{y}_j \end{vmatrix} = (\dot{x}_i \dot{y}_j) - (\dot{x}_j \dot{y}_i) = (v_{xi} v_{yj}) - (v_{xj} v_{yi}) \quad (15)$$

Die Koeffizientendarstellung von (8) ergibt sich respektive (11), (13) und (15) entsprechend zu

$$\begin{aligned} & a_{xk} n_x && + \\ & a_{yk} n_y && + \\ & a_{xj} (-\alpha n_x) && + \\ & a_{yj} (-\alpha n_y) && + \\ & a_{xi} (\alpha - 1) n_x && + \\ & a_{yi} (\alpha - 1) n_y && + \\ & v_{xi} v_{yj} - v_{xj} v_{yi} && + \\ & v_{xj} v_{yk} - v_{xk} v_{yj} && + \\ & v_{xk} v_{yi} - v_{xi} v_{yk} && \geq 0 \end{aligned} \quad (16)$$

Ist also nun die Bedingung für die Seitenrestriktion der drei Punkte p_i, p_j und p_k gegeben, so muss diese Nebenbedingung bei der Optimierung der Zielfunktion gelten.

Reduzierung des Untersuchungsraum der Seitenrestriktion: Die Seitenrestriktion liefert die Möglichkeit, eine Seiten-Punkt-Konstellation zu restriktieren. Der Anwendungsbedarf wird über die Dreiecksfläche ermittelt. Ein Würfel hat jedoch vier Seiten. Es stellt sich die Frage, wie man bestimmt mit welcher der vier Seiten ein beliebiger Punkt kollidiert. Denn, da ein Punkt nur mit einer Seite eines Würfels gleichzeitig kollidieren kann, macht es wenig Sinn, ihn auch mit den restlichen Seiten zu restriktieren.

Aus diesem Grund werden dem Würfel w_i , für den bestimmt werden soll, ob es Punkte anderer Würfel w_k gibt, die mit ihm kollidieren vier Sektoren S_0, S_1, S_2, S_3 zugeordnet. Jeder dieser Sektoren besitzt eine eindeutige Seite des Würfels w_i . Befindet sich ein Punkt p_k des Würfels w_k in einem Sektor, so kann er eindeutig einer Seite des Würfels w_i zugeordnet werden. Die Sektorzuordnung erfolgt über Vollkreiswinkelbestimmung. Dabei dient der Würfelmittelpunkt p_m des Würfels w_i als Koordinatenursprung.

Um den Sektorbereich einzugrenzen, wird die Sektorbestimmung nur für Punkte p_k durchgeführt, die einen bestimmten Abstand zu p_m nicht überschreiten.

Konkret müssen also erst einmal alle Würfel mit allen Punkten der anderen Würfel untersucht werden. Dieser Untersuchungsraum wird pro untersuchten Würfel über den Mittelpunktabstand eingeschränkt. Da die Mittelpunktabstandsberechnung relativ kostengünstig ist, sollte die Anzahl ($O(n^2)$) von Untersuchungen der Komplexität $O(1)$ akzeptabel sein.

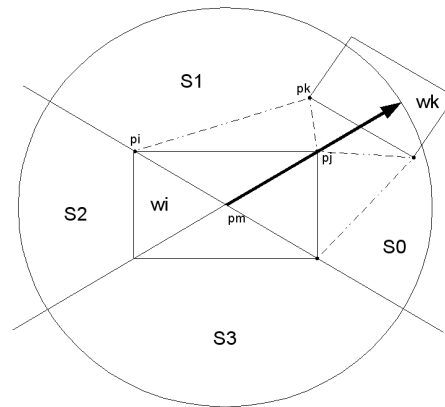


Abbildung 3: Sektorenaufteilung eines Würfels

Tisch und Tischrestriktion: Der Tisch wird zunächst behandelt, wie ein normaler Würfel. Allerdings gilt für ihn noch die Tischrestriktion. Die Tischrestriktion restriktiert die Beschleunigung der Tischpunkte auf Null. Sollte der Tisch zu Beginn der Simulation still im Raum stehen (was der Regelfall sein sollte), so wird er es dadurch auch am Ende noch tun. Wenn sich seine Punkte nie bewegen, braucht man auch keine Längenrestriktion auf sie anwenden. In die Seitenrestriktionsuntersuchung und Anwendung ist der Tisch allerdings inkludiert, damit Würfel nicht in ihn hinein beschleunigt werden können.

Die Masse des Tisches wird auf den maximal möglichen Wertes des genutzten Zahlenformats gesetzt.

Impulsübertragung: Die Seitenrestriktionen verhindern das Eindringen von Würfeln ineinander nur, wenn diese in Ruhe sind. Da sie keinen direkten Einfluss auf die Geschwindigkeit der Punkte haben, fallen unsere Würfel ineinander, wenn keine externen Maßnahmen getroffen werden. Um ein realistisches Verhalten der Würfel zu erreichen, müsste zusätzlich elastischer Stoss simuliert werden.

Hierbei sollte die Impulsübertragung zum gleichen Zeitpunkt wie die Seitenrestriktion angewandt werden, da hier ohnehin Dreiecksflächen zur Einhaltung der Seitenrestriktion berechnet werden. Wenn der Betrag dieser Dreiecksflächen sehr klein ist, kann man davon ausgehen, dass zwei Würfel zusammenstossen sind und eine Impulsübertragung vorgenommen werden muss.

Im Moment des Aufpralls muss der Impuls nun abhängig von der Masse auf alle beteiligten Körper verteilt werden. Man kann vereinfachend davon ausgehen, dass immer nur zwei Körper gleichzeitig kollidieren, niemals aber drei. Im Falle einer tatsächlichen Kollision von drei Würfeln im exakt selben Schritt würden sie Impulsübertragungen separat betrachtet werden (also als zwei unabhängige Kollisionen), was nicht den tatsächlichen physikalischen Gegebenheiten entspricht.

Die genaue Berechnung der Impulsübertragung stellt sich als nicht-trivial heraus. Gleichungen enthielten zu viele Unbekannte. Die Idee, die Bewegung in einen Linear- und einen Drehanteil zu zerlegen und Impulsübertragung über einen gemeinsamen Faktor zu modellieren führte zunächst zu folgenden Gleichungen:

$$M^I \tilde{\mathbf{v}}^I = M^I \mathbf{v}^I - \lambda \mathbf{n} \quad (17)$$

$$M^{II} \tilde{\mathbf{v}}^{II} = M^{II} \mathbf{v}^{II} + \lambda \mathbf{n} \quad (18)$$

für den Linearimpuls. M ist hier jeweils die Masse der vollständigen Körper, \mathbf{v} die Lineargeschwindigkeit des Körpers (diese ist für alle Punkte gleich). λ bezeichnet den Faktor, um den sich der Impuls jeweils ändert. \mathbf{n} ist in diesem Fall der Normalenvektor, man kann aber zeigen, dass hier beliebige Vektoren gewählt werden können.

$$I^I \tilde{\omega}^I = I^I \omega^I - \mathbf{r}^I x(\lambda \mathbf{n}) \quad (19)$$

$$I^{II} \tilde{\omega}^{II} = I^{II} \omega^{II} - \mathbf{r}^{II} x(\lambda \mathbf{n}) \quad (20)$$

$$\text{mit } I^I = \sum m_i r_i \quad (21)$$

für den Drehimpuls. m_i und r_i bezeichnen jeweils die Masse und die Entfernung zum Schwerpunkt der einzelnen Punkte innerhalb eines Körpers. \mathbf{r} ist ein beliebiger Vektor vom Schwerpunkt zu einem der äußeren Punkte.

Um nun die Impulse für die einzelnen Punkte auszurechnen, muss λ bestimmt werden. Da λ in beiden Formelsätzen vertreten ist, sind auch Übergänge von Dreh- in Linearimpuls und umgekehrt möglich. Allerdings mussten wir später feststellen, dass diese Formeln nicht allgemein gültig sind. Impulserhaltung gilt nur in der Summe der Linear- und Drehimpulse und damit ist die Isolation nach λ unzulässig. Damit war die Idee, die Variablenanzahl auf diese Weise zu reduzieren, gescheitert.

Die Suche nach einem Modell, welches die Impulserhaltung in der Summe berücksichtigt stellte sich als sehr zeitintensiv und schlussendlich ergebnislos heraus. Die verbliebene Zeit investierten wir in eine saubere Implementierung des vorhandenen Modells ohne Impulsübertragung, da dies für die Ausgangsfrage ("Wann sind Konstellationen, die anfangs in Ruhe sind, stabil?") korrekte Ergebnisse liefert.

4 Zeitenauswertung

Die folgenden Beispielaufbauten wurden berechnet und die dafür benötigte Zeit gemessen. Bei den Zeiten handelt es sich um reine Prozesszeiten. Pro Iteration ergibt sich eine konstante Zeit. Für die Zeitmessung werden pro Beispiel 20 Iterationen auf einem durchschnittlichen PC berechnet. Eine Abhängigkeit besteht offensichtlich bezüglich der Würfelanzahl. Der Werteverlauf läßt dabei ein exponentielles Wachstum vermuten.

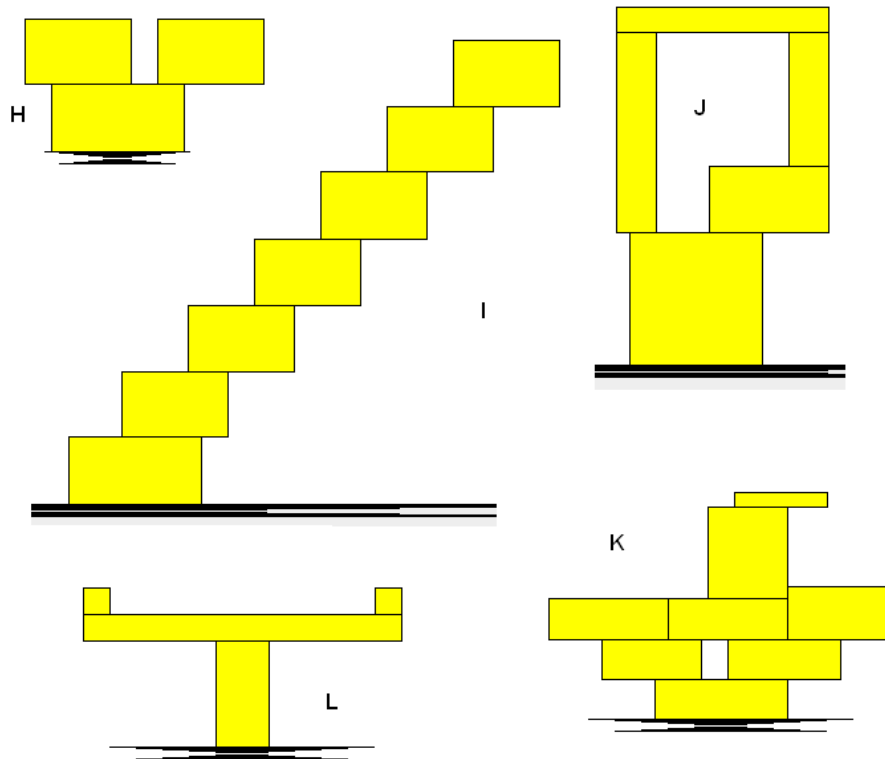


Abbildung 4: Die, für die Auswertung benutzten, Szenarien

Beispiel	H	I	J	K	L
Würfel	3	7	5	8	4
Zeit[s]	25.40	1092.53	231.34	3199.89	40.27

5 Visualisierung

Die Visualisierung ist die zweite große Komponente des Systems. Sie wurde vollständig in Java geschrieben um größtmögliche Plattformunabhängigkeit zu

gewährleisten. Außerdem erlaubt Java, mit einfachen Mitteln die Präsentation der Inhalte im Internet. Abbildung 5 zeigt einen Screenshot der Desktopanwendung. Zusätzlich zur Desktopanwendung steht ein einfaches Java-Applet bereit, um einen Trace im Internet zu präsentieren.

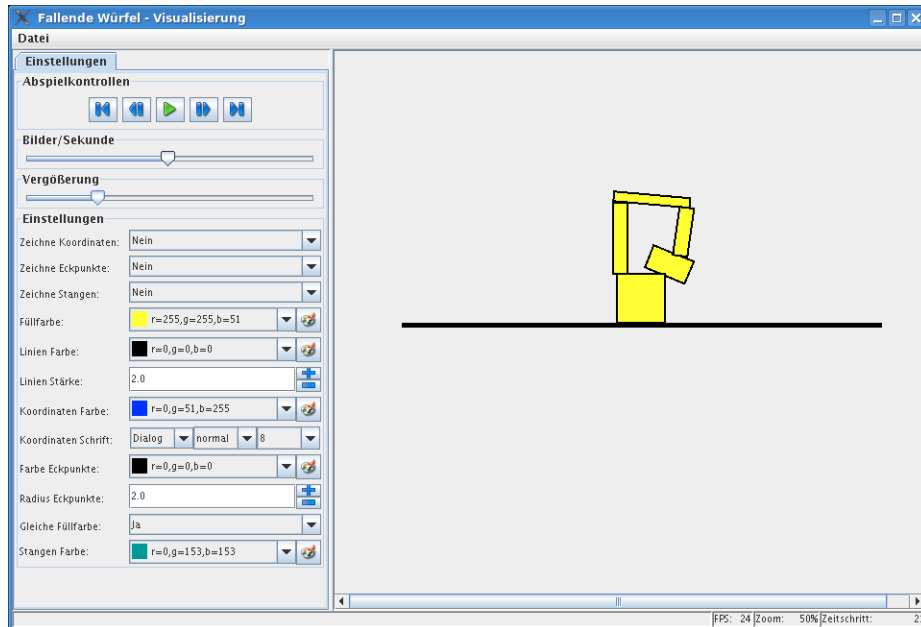


Abbildung 5: Screenshot der Desktopanwendung

Die Aufgabe der Visualisierung ist es, ein Trace einzulesen und die Informationen grafisch darzustellen. Die einzelnen Simulationsschritte werden als Folge von Einzelbildern animiert. Die Animation kann dabei interaktiv gesteuert werden.

Die Softwarearchitektur besteht aus drei Modulen.

1. Eingabemodul
2. Animationsmodul
3. Darstellungsmodul

Das Eingabemodul hat die Aufgabe, ein Trace einzulesen, die Validierung durchzuführen und die Simulationsschritte in entsprechende Objekte zu verpacken.

Das Animationsmodul benutzt die Daten der Eingabe und stellt den nächsten darzustellenden Simulationsschritt bereit. Es nimmt auch die Interaktionsbefehle aus der Darstellung entgegen und behandelt diese entsprechend.

Das Darstellungsmodul erhält einen Zeitschritt und benutzt die grafische Oberfläche um ihn zu zeichnen.

6 Fazit

Alles in allem haben wir fast jedes der uns gesteckten Ziele erreicht. Auf dem Weg dorthin waren aber einige Probleme zu lösen.

Mit den Informationen aus [4] hatten wir zwar ein gutes theoretisches Modell als Basis. Allerdings wussten wir oft nicht, was genau dieses Modell leistet und wo wir selbst etwas tun müssen. Diese Informationen haben wir erst nach und nach in Erfahrung bringen können, wodurch sich Verzögerungen ergeben haben.

Mit der Betrachtung der Impulsübertragung am Ende kam auch die Frage auf, ob das gewählte Modell, die Würfel als Punktmenge zu behandeln, die richtige Wahl war oder ob eine andere Betrachtung besser gewesen wäre.

An dieser Stelle sei noch die sehr gute Zusammenarbeit in der Gruppe erwähnt. In wöchentlichen Treffen wurden aktuelle Probleme und Lösungen besprochen und vorgestellt und die Arbeitspakete für die darauf folgende Woche vergeben, die dann selbständig bearbeitet wurden. Trotz der Einzelarbeiten wurde regelmäßig über E-Mail kommuniziert. Dadurch wurde nicht gegeneinander gearbeitet und es entstand kein Chaos im Quellcode.

Literatur

- [1] *Institut für Informatik, Freie Universität Berlin*. <http://www.inf.fu-berlin.de>
- [2] CGAL, *Computational Geometry Algorithms Library*. <http://www.cgal.org>
- [3] FISCHER, Kaspar ; GÄRTNER, Bernd ; SCHÖNHERR, Sven ; WESSENDORP, Frans: Linear and Quadratic Programming Solver. Version: 3.3, 2007. http://www.cgal.org/Manual/3.3/doc.html/cgal_manual/packages.html#Pkg:QPSolver. In: BOARD, CGAL E. (Hrsg.): *CGAL User and Reference Manual*. 3.3. 2007
- [4] ROTE, Günter: *Mitschriften vom ersten Treffen zum Projekt „Fallende Würfel“*. Mai 2007
- [5] ROTE, Günter: *Praktikum über effiziente Algorithmen*. <http://www-depreciated.inf.fu-berlin.de/lehre/SS07/Praktikum-Algorithmen>. Version: 2007
- [6] WANG, Miao ; RICHTER, Robert ; THIEL, Florian ; SCHULZE, Manuel: *Homepage zum Projekt „Fallende Würfel“*. <http://page.mi.fu-berlin.de/mschulze/fcs/>. Version: Juli 2007