Prof. G. Zachmann
David Mainzer

TU Clausthal
Institut für Informatik
7. Dezember 2011

Wintersemester 2011/12

Übungen zu Virtuelle Realität und Simulationa - Blatt 5

Abgabe am 30. 12. 2011

Senden Sie bitten bis zum Abgabedatum die Lösungen an dm@tu-clausthal.de.

Aufgabe 1 (Feder-Masse-System, *10+5+(5) Punkte*)

Laden Sie sich von der Webseite das Framework zum Feder-Masse-System herunter.

Es soll eine Menge von Punktmassen m_i mit $i = 1 \dots N * N$ durch eine Menge von Federn $s_{ij} = (i, j, k_s, k_d)$ (Ruhelänge l_0, Federkonstante (= Steifigkeit) k_s und den Dämpfungskoeffizienten k_d) verbunden werden.

Das Framework besteht aus:

- `Vector3.java` – eine Klasse für 3D Vektoren
- `Spring.java` – Klasse, welche die Federn im Feder-Masse-System beschreibt
- `MassPoint.java` – Klasse, welche die Punktmassen im Feder-Masse-System beschreibt
- `SpringMassSystem.java` – das Feder-Masse-System
- `tuch256.wrl` – `wrl`-File zum Starten des Feder-Masse-Systemes (geringe Auflösung)
- `tuchHigh.wrl` – `wrl`-File zum Starten des Feder-Masse-Systemes (höhere Auflösung)

Das Framework lädt das Mesh, welches ein Tuch simuliert (siehe Abbildung 1 und 2), aus der Datei `tuch256.wrl` und übergibt dieses an das Feder-Masse-System. Dies erzeugt eine Menge von Punktmassen und alle Federn zwischen den Punktmassen.

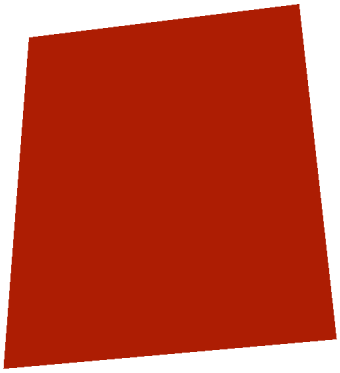

Abbildung 1: "Tuch" in seiner Ausgangslage (ohne Kräfte)

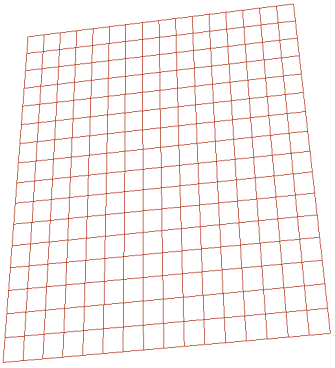

Abbildung 2: "Tuch" als Drahtgitter (jeder Gitterpunkt stellt eine Punktmasse dar)

Ihre Aufgabe im einzelnen:

a) Integrieren Sie die "Federn" korrekt in das System (Funktion **`public void simulate()`**, Zeile 376). Hierfür muss der Abstand zwischen zwei Punktmassen und die daraus resultierende Kraft mit entsprechender Richtung bestimmt werden. Es soll auch die Federkonstante sowie der Dämpfungskoeffizienten in die Berechnung eingehen. Die resultierende Kraft muss dann der entsprechenden Punktmasse zu gewiesen werden.
   Tip: Achten Sie darauf, dass die Ruhelänge der Federn (wird pro Feder bei der Initialisierung gesetzt) zu Beginn gleich dem Abstand der Punktmassen ist. Das bedeutet, dass das System nach korrekter Implementierung der Federn das Tuch nicht verändern wird, solange keine weitere Kraft von Außen auf das Tuch wirkt. Wie in der Vorlesung (siehe 10 - mass spring systems) besprochen, muss Δt klein genug gewählt werden, damit das System nicht explodiert (Overshooting). Um die Gravitationskraft auf das Feder-Masse-System wirken zu lassen, entfernen Sie die Kommentarezeichen vor den Zeilen 411 und 412. Bei korrekter Implementierung der Feder sollte das Tuch jetzt nicht einfach nach unten fallen, sondern sollte sich, wie in Abbildung 3 zu sehen ist, verhalten.

b) In der Funktion **`public void simulate()`**, Zeile 399f, werden von ihnen die Kräfte, welche auf die zwei Punktmassen einer Feder wirken bestimmt. Ihre Aufgabe ist, dafür zu Sorgen, dass die Federn zwischen den Punktmassen mit den IDs 287, 159, 9 und 84 immer die Ausgangslänge der Feder, welche sich zwischen den Punktmassen befindet, als Abstand haben. Verwenden Sie hierfür die Verlet-Integration (Folie 17 im Foliensatz 10 - mass spring systems), da dieses Integrationsverfahren Constraints sehr leicht behandeln kann.

c) Bonusaufgabe: Bis hierher haben alle Federn dieselben Konstanten k_s und k_d (obwohl in der Klasse eine Instanzvariable dafür pro Feder vorgesehen ist). Ihre Aufgabe ist hier, diese Konstanten für die Federn nach dem Einladen (aber vor der Simulation) verschieden zu setzen (verwenden Sie einen anderen Konstruktor oder die entsprechenden Setter-Methode der Klasse) und damit zu experimentieren. Z.B. könnten Sie die Konstanten positionsabhängig setzen, z.B. die oberen steifer als die unteren, oder von der Mitte nach außen von steif zu sehr flexibel, oder umgekehrt.

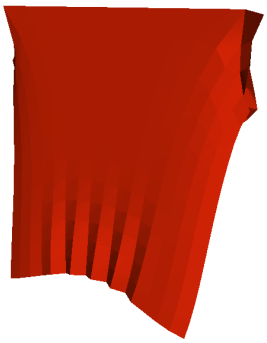

Abbildung 3: "Tuch" auf welches die Schwerkraft wirkt (bis auf einige Punktmassen)

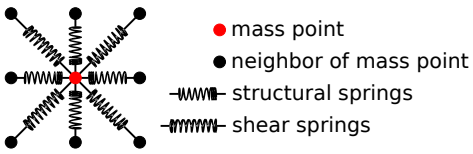

Abbildung 4: Verbindung zwischen den Punktmassen

Die Verbindungen zwischen zwei Punktmassen werden in der Abbildung 4 dargestellt. Hier ist zu sehen, dass zwei Arten von Federn im System auftreten. Dies dient der besseren Strukurerhaltung des Tuches. Jede Punktmasse (siehe Klasse `MassPoint.java`) besitzt eine Position, eine Kraft, welche auf sie wirkt, und ein Gewicht. Jede Feder (sieht Klasse `Spring.java`) besitzt eine Steifigkeit, einen Dämpfungskoeffizienten, eine Ausgangslänge und Indizies, welche für die Punktmassen, welche an der Feder hängen benötigt werden. Achtung: Diese IDs sind nicht gleich der IDs (Indizies der Eckpunkte des Meshes) aus dem `wrl`-File!!