Ludwig-Maximilians-Universität München
LFE Medieninformatik
Prof. Dr. Heinrich Hußmann
Henri Palleis

Übungsblatt 10: Animation und Partikelsysteme

Abgabe:

Dieses Übungsblatt ist einzeln zu lösen. Die Lösung ist bis **Montag**, den **16.Juli 2012, 12:00 Uhr s.t.** über UniWorx (https://uniworx.ifi.lmu.de/) abzugeben.

Aufgabe 1: Pfadanimation

Diese Aufgabe baut auf den Klassen Casteljau.java und DrawCasteljau.java von Übungsblatt 5 auf. Ziel ist es, einen beliebigen dreidimensionalen Körper entlang der angenäherten Beziér-Kurve zu animieren. Der Körper soll dabei gleichmäßig und in einer Endlosschleife entlang der Kurve bewegt werden (ist das Ende der Kurve erreicht, fängt die Animation nahtlos von vorne an).

Hinweise: Die Klasse Animator (`com.jogamp.opengl.util.Animator;`) kann zur Animation verwendet werden. Sorgen Sie zudem dafür, dass der Animator beim Schließen des Programms beendet wird.

Aufgabe 2: Ein einfaches Partikelsystem

Ziel dieser Aufgabe ist es, ein einfaches Partikelsystem zu programmieren.

i. Was versteht man in der Computergrafik unter einem Partikelsystem?

ii. Implementieren Sie zunächst eine Klasse Particle.java, die Partikel-Objekte erzeugen kann. Ein Partikel zeichnet sich durch seine Position im Raum (posx, posy, posz), seine Farbe (r,g,b) und seine Geschwindigkeit in alle Richtungen (vecx, vecy, vecz) aus. Weiterhin besitzt es einen Wert, der seine Lebenszeit repräsentiert (lifetime).

iii. Implementieren sie eine Methode in der Klasse Particle.java, welche die Evolution des Partikels berechnet. Hier soll folgendes passieren:

   a. Die Position im Raum muss aktualisiert werden. Addieren sie hierzu zum Positionsvektor den Geschwindigkeitsvektor. Modellieren Sie zudem eine Annäherung an die Schwerkraft. Dazu wird die Geschwindigkeit in der y-Richtung bei jedem Evolutionsschritt um einen Faktor von 0.0007 verringert.

   b. Die Lebenszeit soll verringert werden (z.B. um den Faktor 0.0001). Ist die Lebenszeit abgelaufen, soll das Partikel mit neuen Werten wieder neu belebt

werden.

c. Kommt das Partikel auf dem Boden auf, soll es abprallen. Um einen realistisch wirkenden Effekt zu erzielen, soll hier die Geschwindigkeit in der y-Richtung mit einem Faktor von -0.6 multipliziert werden.

d. Wie die Werte für Position, Geschwindigkeit, Farbe und Lebenszeit initialisiert werden, hängt davon ab, welchen Effekt man erzielen möchte. Möchte man beispielsweise einen Springbrunnen-Effekt erzeugen (siehe Abbildung 1), dann sollten sich die Partikel zu Beginn (und bei der erneuten Initialisierung nach Ablauf der Lebenszeit) z.B. am Ursprung befinden. Die Geschwindigkeit in y-Richtung kann zufällig gewählt werden. Die Geschwindigkeit in x- und z-Richtung kann so gewählt werden, dass sich die Bewegungen der Partikel in einem Kreis um den Ursprung verteilen.

iv. Implementieren Sie eine Klasse ParticleRenderer.java, welche das Partikelsystem animiert. Erzeugen Sie eine große Menge an Partikeln und speichern Sie diese in einer geeigneten Datenstruktur. Iterieren Sie in der Display()-Methode über die Partikel: Die Partikel sollen als TRIANGLE_STRIP gerendert werden und bei jeder Iteration sollen alle Partikel aktualisiert werden (Evolution).

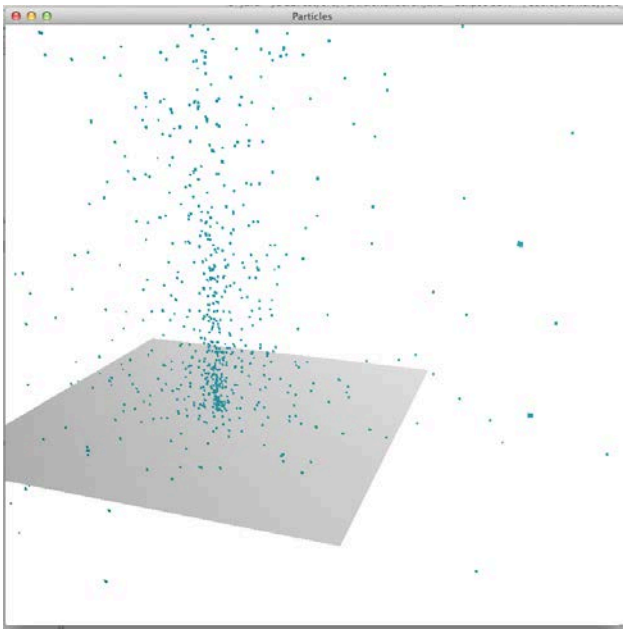

Abbildung 1 Ein Frame der animierten Partikelföntäne

Viel Erfolg.