3D-Computergrafik I

Sommersemester 2015 – Aufgabenblatt 4

Hasso-Plattner-Institut
Fachgebiet Computergrafische Systeme

Aufgabe 12: Zusammenhängende Transformationen (3 Punkte)

Ein gelber Würfel soll sich periodisch um einen roten Würfel bewegen. Bei dieser Bewegung sollen sich die Würfel stets mit mind. einer zur z-Achse parallelen Kante berühren (siehe Abbildung 1).

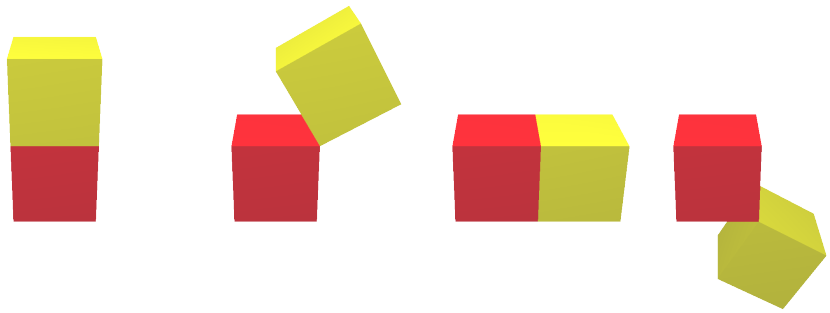

Abbildung 1: Animation (Zwischenresultat) bei Takt 0, 30, 90, 150.

Die Würfel haben eine Kantenlänge von 1. Der rote Würfel liegt im Ursprung und ist entlang der Hauptachsen ausgerichtet. Als Eingabe steht die Frame-Nummer, welche die Zahl der gerenderten Frames seit Programstart angibt, zur Verfügung. Implementieren Sie eine Taktung, welche periodisch von 0 bis 359 läuft. Bei Takt 0, 90, 180 und 270 sollen sich die Würfel jeweils an einer Fläche berühren.

Die hierfür notwendigen Transformation sollen in der Methode `rotateClockwise` in `exercise12.cpp` mittels geeigneter Matrix Anweisungen (`QMatrix4x4` mit den Methoden `rotate`, `translate` und `scale`) umgesetzt werden.

Aufgabe 13: Animation einer elastischen Kugel (4 Punkte)

Ziel dieser Aufgabe ist es, ein Modell einer Kugel durch geometrische Transformationen zu animieren. Der bereitgestellte Programmrahmen definiert eine texturierte Kugel, die sich mit konfigurierbarer, konstanter Geschwindigkeit `fX` in positiver x-Richtung ein Plateau entlang bewegt. Bei Übertreten der Plateaukante fällt die Kugel auf den Boden herab, springt wieder hoch und erreicht ein zweites, tiefer liegendes Plateau (siehe Abbildung 2).

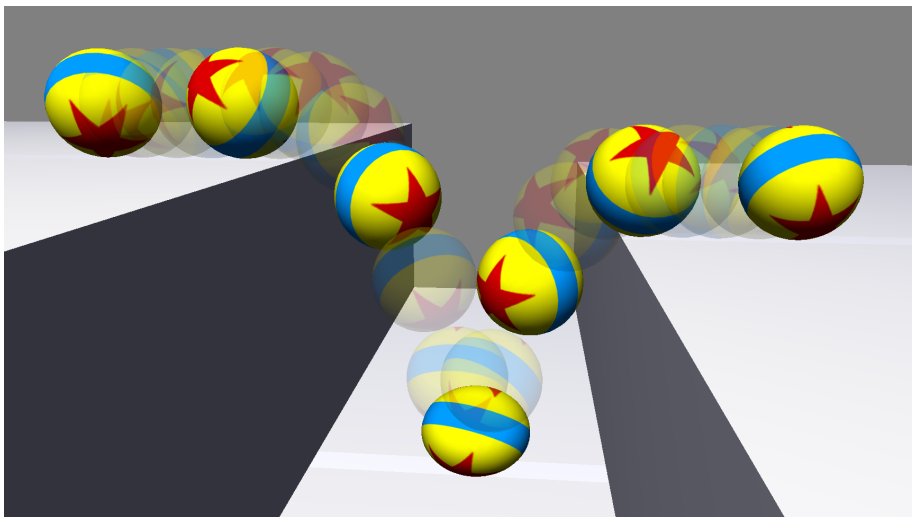

Abbildung 2: Beispielanimation einer Kugel

Die Bewegung der Kugel soll möglichst realistisch gestaltet werden, d.h. sie soll (1) entsprechend ihrer horizontalen Bewegung korrekt rotieren und (2) sich beim Auftreffen auf der Bodenplatte verformen. Die Kugel hat einen Radius von $r = 0.3$ und sollte (3) kurz hinter der Mitte des Spaltes auf der Bodenplatte aufschlagen. Dabei soll der Ball bei Auftreffen auf die Bodenplatte um $d = 0.3r$ gestaucht werden. Ergänzen Sie (4) eine entsprechende Skalierung, welche ein tatsächliches Durchdringen der Bodenplatte verhindert und den Aufprall stattdessen als Verformung nachwirken lässt.

Stellen Sie für die Translation, Skalierung und Rotation *um die z-Achse* je eine Matrix vom Typ `QMatrix4x4` in der Datei `exercise13.cpp` bereit und multiplizieren Sie diese in korrekter Reihenfolge zur Berechnung der kombinierten Transformationsmatrix.

Aufgabe 14: Interpolationsverfahren für Rotationsbewegungen (8 Punkte)

Ein 3D Objekt bewegt sich von links nach rechts und soll dabei unter Verwendung unterschiedlicher Interpolationsverfahren rotiert werden:

a) Lineare Interpolation (*lerp*) der Elemente der Rotationsmatrix (2 Punkte).

b) Lineare Interpolation (*lerp*) mittels Eulerwinkeln (2 Punkte).

c) Spherical Linear Interpolation (*slerp*) mittels Quaternionen (4 Punkte).

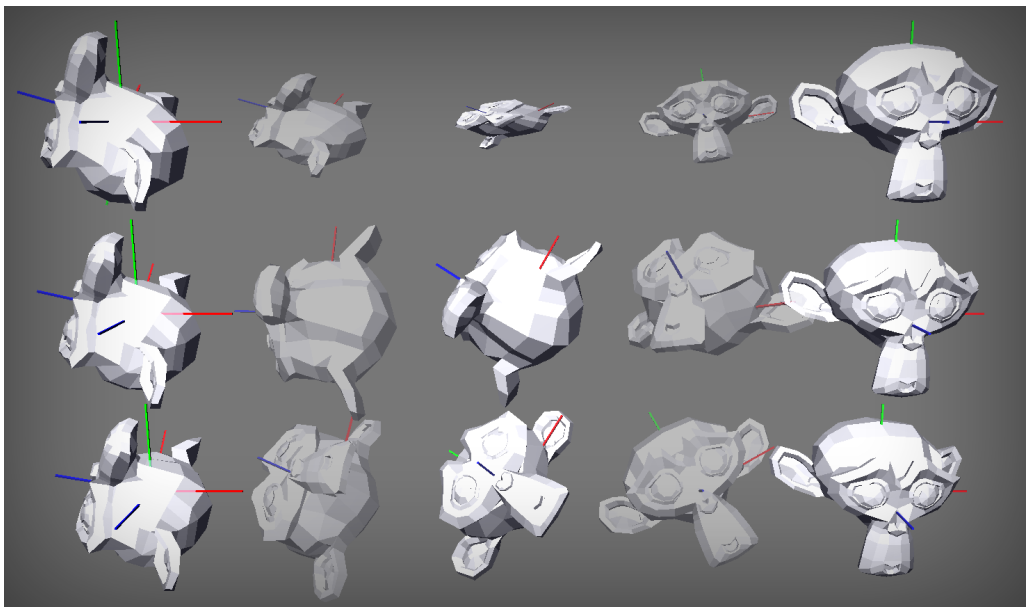

Abbildung 3: Schrittweise Animation einer Rotation mittels Interpolation der Matrixelemente (oben), der Eulerwinkel (mitte) und schließlich Interpolation mit Quaternionen (unten).

Die Variable t läuft während eines Durchlaufs von 0.0 bis 1.0 und definiert den aktuellen Stand der Interpolation. Die Start- und Endorientierung werden durch Eulerwinkel beschrieben und in den Variablen `m_angles0` und `m_angles1` gespeichert.

Implementieren Sie (1) eine lineare Interpolation der Matrixelemente in der Methode `interpolateMatrix`, (2) eine lineare Interpolation mittels Eulerwinkeln in `interpolateEuler`, und schließlich (3) eine sphärische Interpolation mit Hilfe von Quaternionen in der Methode `interpolateQuaternion`. Implementieren Sie zudem die Hilfsmethoden `lerp` und `slerp`. Die Funktionen sollen in der Methode `applyBallTransformation` in `exercise14.cpp` benutzt werden.

Allgemeine Hinweise:

- Die Aufgaben sollen maximal zu zweit bearbeitet werden; Ausnahmen müssen mit den Übungsleitern abgesprochen werden.
- Bitte reichen Sie Ihre Lösungen bis Dienstag, den **02.06.2015**, um **13.15** Uhr ein.
- Tragen Sie für ihre Matrikelnummern in die Quellcode-Dateien ein. Beachten Sie, dass nur die vollständigen Quelltexte, Projektdateien und Mediadateien geschickt werden sollen. Senden Sie uns keinesfalls ausführbare Dateien oder bereits kompilierte Binär- oder Temporärdateien (*.obj, *.pdb, *.ilk, *.ncb etc.)! Testen Sie vor dem Verschicken, ob die Projekte aus den Zipdateien fehlerfrei kompiliert und ausgeführt werden können.
- Zippen Sie ihre Lösungen in **eine** Zip-Datei. Geben Sie der Zip-Datei einen Namen nach folgendem Schema:

  cg1_blatt4_matrikelnummer1_matrikelnummer2.zip.
- Die Zip-Datei laden Sie dann bei moodle hoch.