Raytracing Aufgabe 05 - Szenengraph

Benjamin Schurian, Clemens Pade, Robert Dimitrov
Computergraphik I

WS2015/16

1 Einführung

In der fünften Übungsaufgabe des Semesters werden die Themen Transformationen und Szenengraph behandelt. Am Anfang ist die Mathe-Bibliothek um eine Klasse zu erweitern, die eine Matrix mit vier Zeilen und vier Spalten darstellt. Danach sollte eine Klasse implementiert werden, die die verschiedenen Grundtransformationen darstellt. Zu schreiben ist außerdem eine von `Geometry` abgeleitete Klasse (`Node`), mit deren Hilfe komplexere Figuren erstellt werden können, die aus mehreren Geometrien zusammengebaut sind. Abschließend sind "als Beweis" zwei vom Dozenten beschriebenen Szenen zu erstellen, in denen verschiedene Objekte transformiert werden. Daneben muss auch ein UML-Diagramm gezeichnet werden, das die veränderten bzw. neu erstellten Klassen und deren Verbindungen veranschaulicht.

2 Teilaufgabe 1

Analog zu der `Mat3x3` aus der ersten Übungsaufgabe ist eine Klasse zu schreiben, die eine 4x4 Matrix darstellt und gewisse Rechenoperationen zur Verfügung stellt. Diese bekommt im Konstruktor 16 Werte, die den verschiedenen Elementen der Matrix zugewiesen werden. Da es umständlich ist, für all diese Elemente setter-Methoden zu schreiben (zu hoher Aufwand, keine Parameterüberprüfung nötig), sind alle Attribute der Matrix öffentlich und final gesetzt. Die Klasse bietet verschiedene Methoden zur Multiplikation mit jeweils einem `Vector3`, einem `Point3` und einer anderen `Mat4x4`. Darüber hinaus wird die Methode `transpose()` zur Verfügung gestellt, die die transponierte Matrix zurückgibt.

Die Implementierung dieser Teilaufgabe war relativ schlicht. Natürlich musste man viel bei den Berechnungen wegen der vielen Attributen vorsichtig vorgehen, da wegen ihrer Anzahl leicht Tippfehler entstehen können. Problematisch war die Multiplikation mit einem Vektor/Punkt, da aus mathematischer Sicht dies nicht zugelassen ist (4x4 mal 3x1), weswegen auf die letzte Zeile und Spalte ignoriert werden musste.

- **Teammitglied:** Robert
- **Einlesen:** 10 Minuten
- **Programmieren:** 40 Minuten

3 Teilaufgabe 2

Ziel der zweiten Teilaufgabe ist die Implementierung der Klasse, die für die Transformationen zuständig ist. Die Klasse verfügt über zwei Attribute vom Typ `Mat4x4` - m (für die Transformationsmatrix) und i (für ihre Inverse). Erzeugt werden folgende Operationen - Translation, Skalierung, Rotation (um die X-, Y- und Z-Achse). Der öffentliche Konstruktor dieser Klasse nimmt keine Parameter entgegen und initalisiert die beiden Attribute mit der Einheitsmatrix. Ein weiterer privater Konstruktor nimmt zwei Matrizen als Parameter für die Initialisierung der Werte. Neben den oben aufgeführten Transformationen

werden auch Methoden zur Multiplikation mit einem Strahl und einer Normalen angeboten. Diese führen die notwendigen Rechenoperationen während der Transformation des Strahls und der Rücktransformation der Normalen aus.

Bei der Implementierung dieser Klasse sind wir auf keine wesentlichen Probleme gestoßen.

- **Teammitglied:** Robert
- **Einlesen:** 30 Minuten
- **Programmieren:** 2 Stunden

4 Teilaufgabe 3

Die dritte Teilaufgabe dieser Übung beschäftigt sich mit dem Erstellen eines Szenengraphen, der durch die Klasse `Node` dargestellt wird. Diese nimmt im Konstruktor ein `Transform`-Objekt und eine Liste der Geometrien entgegen, aus denen die Figur besteht. Da `Node` von `Geometry` erbt, musste entsprechend die `hit()`-Methode überschrieben werden, indem der übergebene Strahlt transformiert wird und es nach dem Treffer mit dem kleinsten positiven `t` gesucht wird. Dieser Treffen resultiert in einem neuen `Hit`-Objekt, welches mit dem gleichen t auf die gleiche Geometrie, jedoch mit dem untransformierten Strahl sowie der korrigierten Normalen zeigt.

Als Fortsetzung dieser Teilaufgabe wurden die weiteren Geometrien entsprechend angepasst und die Demoszenen erstellt.

- **Teammitglied:** Benjamin
- **Einlesen:** 20 Minuten
- **Programmieren:** 4 Stunden

5 Teilaufgabe 4

Im Gegensatz zu den ersten vier Aufgaben, standen uns bei dieser Übung keine Fachklassendiagrammen zur Verfügung, die uns bei der Arbeit erleichtern sollten. Diesmal sollten wir selbst ein UML-Diagramm erstellen, das die Veränderungen bzw. die neu erstellten Klassen widerspiegelt. Da solche Inhalte auch in anderen Fächern auftauchen, in denen wir bald Klausuren schreiben werden, halten wir diese Teilaufgabe für sehr hilfreich.

- **Teammitglied:** Clemens
- **Bearbeitung:** 1 Stunde

6 Probleme

In der Demo-Klasse sind wir auf ein paar Probleme gestoßen. Erstens hatten wir bestimmte Schwierigkeiten mit der Reihenfolge, in der Transformationen ausgeführt werden müssen, bis wir die erwünschten Figuren erstellen könnten.

Der zweite Fehler, den wir leider nicht beheben konnten, war die falsche Beleuchtung der skalierten Kugel. Bei keiner bzw. einer Skalierung um (1,1,1) wird die `Sphere` richtig beleuchtet. Sobald einer dieser Werte verändert wird, verliert die Kugel ihren Glanz und sieht so aus, als wäre sie mit `SingleColorMaterial` definiert. Dabei wird die Skalierung fehlerfrei ausgeführt.

7 Quellen und Literatur

- Computergraphik I Folien – Stephan Rehfeld