Übungen zur Vorlesung "Architektur und Programmierung von Grafik- und Koprozessoren"
Übungsblatt 5

Sommersemester 2018

5 Computergrafik

Aufgabe 5.1

a.)
Quaternionen sind eine Erweiterung der komplexen Zahlen $\mathbb{C}$ und sind definiert als

$$\mathbb{H} = \{q_0 + q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k} : q_0, q_1, q_2, q_3 \in \mathbb{R}\}, \tag{1}$$

wobei für **i**,**j** und **k** gilt:

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1. \tag{2}$$

Zeigen Sie, dass daraus die Identitäten

$$\mathbf{ij} = \mathbf{k}, \mathbf{jk} = \mathbf{i}, \mathbf{ki} = \mathbf{j} \tag{3}$$

sowie

$$\mathbf{ij} = -\mathbf{ji}, \mathbf{jk} = -\mathbf{kj}, \mathbf{ki} = -\mathbf{ik} \tag{4}$$

folgen. (2 Punkte)

b.)
Die (i. Allg. nicht kommutative) Multiplikation von zwei Quaternionen $p = p_0 + p_1\mathbf{i} + p_2\mathbf{j} + p_3\mathbf{k}$ und $q = q_0 + q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k}$ ist definiert als

$$\begin{aligned} pq = {} & (p_0q_0 - p_1q_1 - p_2q_2 - p_3q_3) \\ & + (p_0q_1 + p_1q_0 + p_2q_3 - p_3q_2)\,\mathbf{i} \\ & + (p_0q_2 - p_1q_3 + p_2q_0 + p_3q_1)\,\mathbf{j} \\ & + (p_0q_3 + p_1q_2 - p_2q_1 + p_3q_0)\,\mathbf{k}. \end{aligned} \tag{5}$$

Zeigen Sie dies mit Hilfe der Rechenregeln aus a.). (4 Punkte)

c.)
Das Quaternion

$$\bar{q} = q_0 - q_1\mathbf{i} - q_2\mathbf{j} - q_3\mathbf{k} \tag{6}$$

nennt man das *Konjugierte* zu q. Weiterhin sei das das *Einheitsquaternion* für $q \neq 0$ gegeben als

$$\frac{q}{|q|} = \frac{q_0}{|q|} + \frac{q_1}{|q|}\mathbf{i} + \frac{q_2}{|q|}\mathbf{j} + \frac{q_3}{|q|}\mathbf{k}, \tag{7}$$

wobei

$$|q| := \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}. \tag{8}$$

Gegeben sind die Eckpunktsvektoren der Box

$$\begin{aligned}\mathbf{v_1} &= (1, 0, 0)\\ \mathbf{v_2} &= (2, 0, 0)\\ \mathbf{v_3} &= (2, 2, 0)\\ \mathbf{v_4} &= (1, 2, 0).\end{aligned}$$

Führen Sie für jeden dieser Vektoren die Operation $qv\bar{q}$ durch. Dafür sind das Einheitsquaternion

$$q = cos\left(\frac{30^\circ}{2}\right) + s_1 sin\left(\frac{30^\circ}{2}\right)\mathbf{i} + s_2 sin\left(\frac{30^\circ}{2}\right)\mathbf{j} + s_3 sin\left(\frac{30^\circ}{2}\right)\mathbf{k}$$

sowie der Vektor $\mathbf{s} = (s_1, s_2, s_3) = (0, 0, 1)$ gegeben. Das Quaternion v bilden Sie, indem Sie dessen *Realteil* v_0 auf 0 setzen und die drei *Imaginärteile* v_1, v_2 und v_3 jeweils mit dem ersten, zweiten und dritten Eintrag der Eckpunktvektoren initialisieren.

Zeichnen Sie die ursprünglichen Eckpunkte der Box, sowie die nun transformierten Eckpunkte, in ein gemeinsames Diagramm ein. Was bedeutet die Operation $qv\bar{q}$ mit Bezug auf die vorgegebenen Parameter geometrisch? (4 Punkte)

Aufgabe 5.2

Im Gerüstprogramm zu dieser Aufgabe finden Sie einen kleinen 2D Ray Tracer, den Sie erweitern sollen.
Um das Gerüstprogramm zu übersetzen, benötigen Sie das Cross Platform Build Tool *CMake* (`https://cmake.org/`). Mit CMake lassen sich Projekte organisieren, die aus mehreren *compilation units* bestehen, welche anschließend gelinkt werden. Mit CMake empfehlen sich sogenannte *out-of-source builds*: legen Sie ein Verzeichnis, z. B. als Unterverzeichnis des Gerüstprogramms, an. Dieses können Sie z. B. `build` nennen. Rufen Sie nun *von dort aus* das CMake Kommandozeilenprogramm auf und übergeben Sie den Ordner, in dem sich die Datei `CMakeLists.txt` befindet, als Kommandozeilenparameter, also mit `bash` etwa:

```
mkdir build
cd build
cmake ..
```

Platformspezifische Parameter lassen sich über den *CMake Cache* setzen. Diesen können Sie etwa mit dem Tool `ccmake` beeinflussen, oder durch direktes Editieren der Datei `CMakeCache.txt` im Verzeichnis `build`. Diese Parameter können dem Programm `cmake` auch direkt auf der Kommandozeile mit übergeben werden, z. B.:

```
cmake .. -DCMAKE_BUILD_TYPE=Release -DCMAKE_CXX_FLAGS="-std=c++1y"
```

Das Gerüstprogramm erfordert ausnahmsweise, dass Sie mit dem C++14 Standard übersetzen.

a.)
Erweitern Sie den 2D Ray Tracer aus dem Gerüstprogramm. Implementieren Sie zunächst das Lichtbrechungsverfahren aus der Vorlesung in der Funktion `refract()`. Das Gerüstprogramm definiert bereits eine einfache 2D Geometrie mit unterschiedlichen Brechungseigenschaften. An den 2D Plots, die das Programm in eine `.pnm` Datei schreibt, können Sie nachvollziehen, ob Sie die Methode richtig implementiert haben. (6 Punkte)

b.)
Behandeln Sie außerdem den Fall der totalen internen Reflektion. Diesen Fall sollte Ihre Implementierung von `refract()` erkennen und in diesem Fall einen 0-Vektor zurückgeben. Der Ray Tracer geht im weiteren davon aus, dass totale interne Reflektion aufgetreten ist, wenn `refract()` einen 0-Vektor zurückgegeben hat und ruft in dem Fall die Funktion `reflect()` auf, um einen neuen Richtungsvektor zu erzeugen. Die Funktion `reflect()` ist von Ihnen noch zu implementieren. Was ist im Gerüstprogramm zu ändern, sodass bei der gegebenen Geometrie und den vorgebenen Primärstrahlen totale interne Reflektion auftreten kann? (4 Punkte)

Bemerkungen: bei der Implementierung der beiden Funktionen sollten Sie mit einfachen Vektoroperationen auskommen, das Aufrufen tatsächlicher trigonometrischer Funktionen sollte nicht nötig sein. Bedenken Sie außerdem, dass bei den in der Vorlesung behandelten Modellen *alle Vektoren*, also die Normale, der Vektor zum Licht sowie der Richtungsvektor zum Betrachter *Einheitsvektoren* sind, die vom Schnittpunkt mit der Oberfläche *wegzeigen*.

Abgabe bitte bis zum 13.06.2018, 22:00h in Ilias.