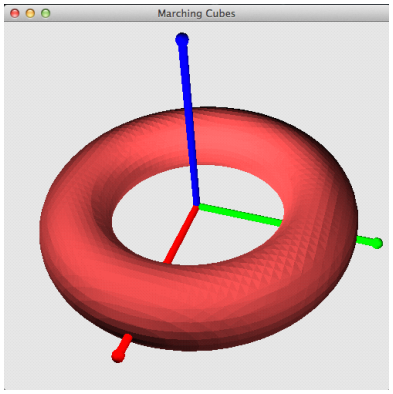
WP Einführung in die Computergrafik

WS 2013/2014, Hochschule für Angewandte Wissenschaften (HAW), Hamburg
Prof. Dr. Philipp Jenke, Lutz Behnke

Aufgabenblatt 3 - Marching Cubes



Darstellung eines Oberflächen-Dreiecksnetzes.

In diesem Aufgabenblatt implementieren Sie den Marching Cubes Algorithmus zur Tesselierung von Impliziten Funktionen in 3D.

Aufgabe 1: Tesselierung eines Würfels

Beginnen Sie die Implementierung mit der Tesselierung eines einzelnen Würfels. Schreiben Sie dazu eine Methode

```
private void createTriangles(Point3d p1, Point3d p2,
      ..., Point3d p8, double v1, double v2, ... ,
      double v8);
```

die für die vier Eckpunkte des Würfels und die dazugehörigen Funktionswerte die notwendigen Dreiecke erzeugt. Die Dreiecke werden in einem TriangleMesh-Objekt (aus dem vergangenen Aufgabenblatt) abgelegt. Gehen Sie folgendermaßen vor:

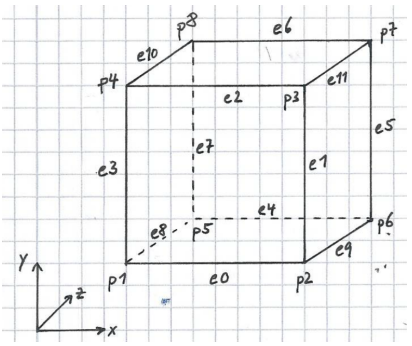
Berechnen Sie für jeden Eckpunkt den zugehörigen Fall:

b_i = (v_i>0)? 1 : 0

und

caseIndex = b_1*1+b_2*2+b_3*4+b_4*8+b_5*16+b_6*32+ b_7*64+b_8*128.

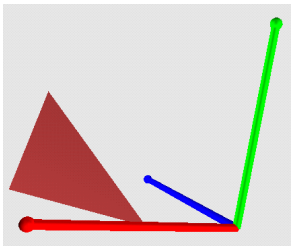
Es gibt demnach 256 verschiedene Fälle. In jedem der Fälle werden zwischen 0 und 5 Dreiecke erzeugt. Die Eckpunkte jedes Dreiecks liegen auf je einer der 12 Kanten des Würfels. Um festzustellen, wie viele und welche Dreiecke benötigt werden, gibt es eine Lookup-Tabelle in der Datei casesLookupTable.txt, die einfach in Ihre Java-Klasse integriert werden kann. In der Tabelle ist jeder Fall also mit 5 (Anzahl Dreiecke) * 3 (Knoten pro Dreieck) = 15 Indizes dargestellt. Der Index -1 gibt an, dass kein Dreieck benötigt wird, alle anderen Indizes geben die zugehörige Kante an. Zunächst verwenden wir die Mittelpunkte der Kanten, um von der Kante auf den Eckpunkt zu schließen.



Würfel mit Bezeichnungen der Eckpunkte (p1-p8) und der Kanten (e0-e11).

Beispiel:

- v_2 ist größer als 0, alle anderen v_i sind kleiner als 0 → caseIndex = 2
- In der Tabelle sind also die Einträge mit den Indizes caseIndex*15 bis (caseIndex+1)*15-1 relevant, also 30 bis 44: 0, 1, 9, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1.
- Nur die ersten drei Einträge sind ungleich -1, daher muss genau ein Dreieck erzeugt werden.
- Die Eckpunkte des Dreiecks liegen auf den Kanten 0, 1 und 9.
- Bei der Kante e_0 bedeutet das: Dreiecks-Eckpunkt = 0.5*(p_1+p_2) (siehe Skizze)



Beispielfall im Würfel $[0;1]^3$

Aufgabe 2: Tesselierung eines Volumens

Erweitern Sie jetzt im zweiten Schritte das Verfahren auf ganze Volumen. Hier gehen wir von einem konstanten Volumen im Bereich $[-2;2]^3$ aus. Um das Volumen in Würfel zu unterteilen, verwenden Sie eine angemessene Auflösung, z.B. 25x25x25. Nun müssen Sie einfach die in der ersten Aufgabe entwickelte Logik auf jeden Würfel einzeln anwenden. Die Werte an den Eckpunkten jedes Würfels werten Sie für eine implizite Funktion aus. Dafür wird das Interface ImplicitFunction3D vorgegeben. Über die Methode

```
public double f(Point3d p);
```

berechnen Sie den Funktionswert an einem beliebigen Punkt im Raum. Im Vorgabe-Quellcode finden Sie zwei Implementierungen dieses Interfaces, Sphere und Torus, zum Testen.