

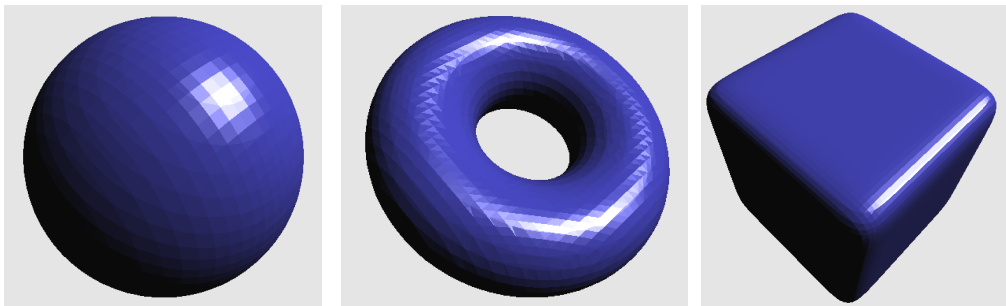
# WP Einführung in die Computergrafik

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



## Aufgabenblatt 4 - Marching Cubes

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



Darstellung von Oberflächen-Dreiecksnetzes (Kugel, Torus, Superquadrik).

### a) Tessellierung eines Würfels

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

```
private void createTriangles(Vector3 p1, Vector3 p2,  
    ..., Vector3 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

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

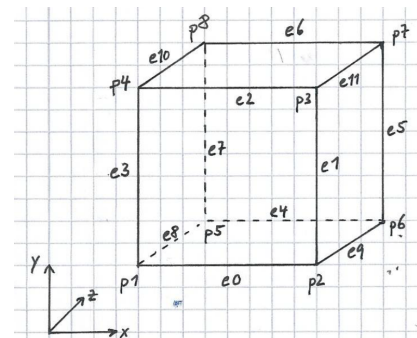
Es gibt demnach 256 verschiedene Fälle. In jedem der Fälle werden zwischen 0 und 5 Dreiecke erzeugt. Die Eckpunkte jedes Dreiecks liegen auf je einer der 12 Kanten des Würfels. Um festzustellen, wie viele und welche Dreiecke benötigt werden, gibt es eine Lookup-Tabelle in der Datei *casesLookupTable.txt*, 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. Die Position eines Dreieck-Eckpunktes auf einer Kante berechnen Sie mit der Formel aus den Vorlesungsfolien:

$$p = (1 - t) * p_{i,j,k} + t * p_{i+1,j,k} \text{ (für Kanten in x-Richtung)}$$
$$t = (\tau - v_{i,j,k}) / (v_{i+1,j,k} - v_{i,j,k})$$

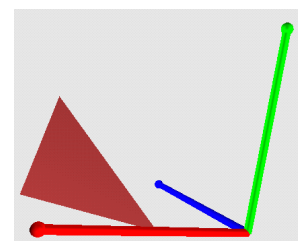
$v_{i,j,k}$  ist dabei der Funktionswert an der Stelle  $p_{i,j,k}$  und  $\tau$  ist der Isowert. Zum Testen können Sie zunächst auch die Mittelpunkte der Kanten verwenden ( $t=0.5$ ).

Beispiel:

- $v_2$  ist größer als 0, alle anderen  $v_i$  sind kleiner als 0  $\rightarrow$  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



Würfel mit Bezeichnungen der Eckpunkte ( $p_1$ - $p_8$ ) und der Kanten ( $e_0$ - $e_{11}$ ).



Beispielfall im Würfel  $[0;1]^3$

Dreieck erzeugt werden.

- Die Eckpunkte des Dreiecks liegen auf den Kanten 0, 1 und 9.
- Bei der Kante  $e_0$  bedeutet das: Dreiecks-Eckpunkt =  $0.5 * (p_1 + p_2)$  (einfach Approximation des Eckpunktes)

## b) Tessellierung 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.  $25 \times 25 \times 25$ . 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. Testen Sie Ihre Implementierung mit mindestens drei verschiedenen impliziten Funktionen.

### Anhang: Implizite Funktionen

Hier finden Sie einige implizite Funktionen zum Testen. Am Besten kapseln Sie die Auswertung einer impliziten Funktion in eine extra Datenstruktur/Klasse:

Kugel:

$$f(x, y, z) = x^2 + y^2 + z^2 - r^2$$

Radius  $r$ , z.B.  $r = 1$

Torus:

$$f(x, y, z) = (x^2 + y^2 + z^2 + R^2 - r^2)^2 - 4R^2(x^2 + y^2)$$

äußerer Radius  $R$ , innerer Radius  $r$ , z.B.  $R = 1$  und  $r = 0.5$

Weitere implizite Funktionen (insbesondere die Superquadriken in Kapitel 3.2) finden Sie unter [1]. Die Arbeit liegt als PDF auf der EMIL-Seite.

### Quellen

[1] Eldar Sultanow (Hasso-Plattner-Institut an der Universität Potsdam): Implizite Flächen