WP Einführung in die Computergrafik

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

Aufgabenblatt 4 - Halbkanten-Datenstruktur

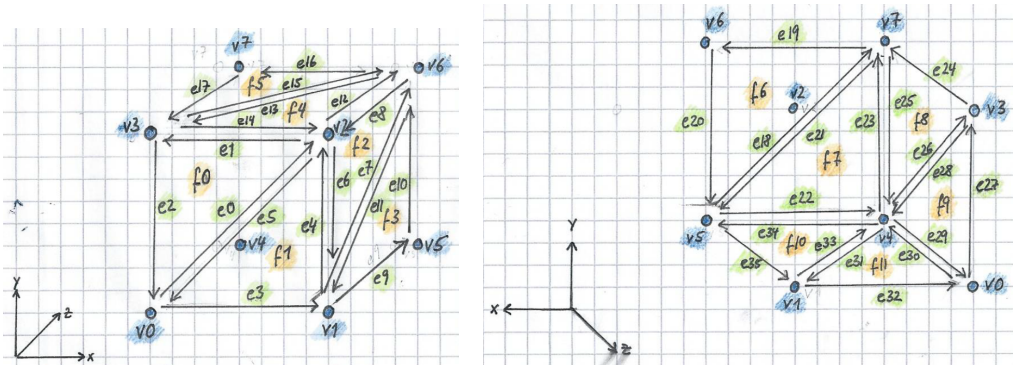

Abbildung: Halbkanten-Darstellung eines Würfels (links: Ansicht von vorne-oben, rechts: Ansicht von hinten-unten). Die gleiche Benennung der Halbkanten (e), Vertices (v) und Facetten (f) wurde in den Testklassen verwendet.

In diesem Aufgabenblatt implementieren Sie eine Halbkanten-Repräsentation für Dreiecksnetze und zugehörige Nachbarschaftsabfragen. In Abbildung 1 ist eine Halbkanten-Darstellung eines Würfels aufgezeigt. Diese Darstellung ist auch in den Testklassen implementiert und wir dort eingesetzt.

Als Rahmen für die Bearbeitung der Aufgaben ist einiger Quellcode vorgegeben:

Halbkanten-Datenstruktur

- Repräsentation eines Vertex: `HalfEdgeVertex`
- Repräsentation einer Dreiecksfacette: `IHalfEdgeFacet` (Interface) und `IHalfEdgeTriangle`
- Repräsentation einer Halbkante: `HalfEdge`
- Container für eine komplette Datenstruktur bestehend aus Halbkanten, Facetten und Vertices: `IHalfEdgeDatastructure` (Interface) und `HalfEdgeDatastructure`

Operationen (Nachbarschaftsabfragen) auf der Datenstruktur

- `IHalfEdgeDatastructureOperations` (Interface)

Tests

- Test der Konsistenz einer Halbkantendatenstruktur: `TestHalfEdgeDatastructure`
- Test der Nachbarschaftsabfragen: `TestHalfEdgeDatastructureOperations`

Aufgabe 1: Konvertierung eines Dreiecksnetzes in eine Halbkanten-Darstellung.

Implementieren Sie Funktionalität zum Konvertieren eines Dreiecksnetzes (Klasse `TriangleMesh` aus den vorherigen Aufgaben) in eine Halbkantendatenstruktur. Wichtig ist, dass alle relevanten Referenzen der Komponenten Vertex, Dreieck und Halbkante korrekt sind (z.B. `next`- und `prev`-Referenz der Halbkanten). Welchen Algorithmus Sie für die Transformation verwenden und welche Laufzeitkomplexität dieser hat, ist Ihnen freigestellt.

Testen Sie die Funktionalität mit mindestens einem eigenen Dreiecksnetz (z.B. ein Tetraeder) und der Testklasse.

Aufgabe 2: Implementierung von Nachbarschaftsabfragen.

Schreiben Sie eine Klasse `HalfEdgeDatastructureOperations`, die das Interface `IHalfEdgeDatastructureOperations` implementiert. Dazu müssen Sie die folgenden Nachbarschaftsabfragen auf Basis der Halbkanten-Repräsentation implementieren:

- Vertex → adjazente Vertices
- Vertex → inzidente Facetten
- Vertex → inzidente Halbkanten (sowohl eingehende als auch ausgehende)
- Facette → adjazente Vertices
- Facette → inzidente Facetten
- Facette → inzidente Halbkanten

Die Implementierungen der Abfragen müssen optimale Laufzeit haben. Es ist also nicht erlaubt, linear die vollständige Datenstruktur zu durchlaufen, um eine Anfrage zu beantworten. Testen Sie Ihre Implementierung mit der Testklasse.