

## Abgabemodalitäten

**Abgabetermin:** Freitag 29.05.2006 - 12.00 Uhr

Abgabe des Programmierteils via E-Mail an [dominik.wodniok@gris.informatik.tu-darmstadt.de](mailto:dominik.wodniok@gris.informatik.tu-darmstadt.de).  
Die Mail sollte folgenden Inhalt haben:

- Gruppennummer und Namen der Mitglieder
- Nummern der bearbeiteten Teilaufgaben
- Eine kurze Erläuterung des Programms (wenige, kurze Sätze)
- Verwendetes Betriebssystem und Entwicklungsumgebung
- Weitere Besonderheiten des Programms (zusätzliche Features, etc)
- Ein Zip-Archiv mit der ausführbaren Datei (*.exe* unter Windows) des Programms mit dem Namen *cg2\_exe2-GRUPPENNR* sowie dem fehlerfrei kompilierbaren Quellcode bzw. Arbeitsbereich

Der Zweck dieser E-Mail besteht darin, eine gewisse Planbarkeit zum Testatstermin herzustellen. Die E-Mail muss nicht die endgültige Programmversion enthalten, Sie können bis zum Testat noch gerne Verbesserungen und Ergänzungen vornehmen.

Benutzen Sie für Fragen, Anregungen und Diskussionen bitte das Forum [\[1\]](#).

## Zweite Programmieraufgabe - Projekt Kalbe & Wodniok

In dieser Übung ist zunächst eine einfache Methode zur direkten Visualisierung aus Volumendaten mit Hilfe der Grafikkarte umzusetzen. Weiterhin sollen Sie effiziente Dreiecksnetz-Datenstrukturen implementieren bzw. die Nutzung entsprechender Bibliotheken erlernen. Zuletzt werden Sie Dreiecksnetze mittels Subdivision-Verfahren unterteilen.

### 10. Direkte Volumenvisualisierung

In einem einfachen Ansatz zur Volumenvisualisierung wird ein Stapel (*Stack*) von 2D-Texturen geladen und zur Visualisierung übereinander geblendet. Implementieren Sie diesen Algorithmus für zumindest eine der drei Hauptrichtungen. Um einen Volumendatensatz der Größe  $i \times j \times k$  beispielsweise entlang der  $z$ -Richtung zu visualisieren, werden zunächst  $k$  2D-Texturen der Größe  $i \times j$  erzeugt. Zur Visualisierung werden nun, bei aktiviertem *blending* und deaktiviertem  $z$ -Buffer, von hinten nach vorne  $k$  Quadrate mit Eckpunkten  $x = 0, y = 0$  und  $x = 1, y = 1$  im Abstand  $h_z$  gezeichnet, wobei die Quadrate mit den entsprechenden Texturen überlegt werden. Nähere Informationen können Sie unter [2] finden (S. 38-42 in den course notes oder in der Präsentation *part2\_textureBased*).

Verwenden Sie 2D-Texturen im Format RGBA. Im einfachsten Fall entsprechen die Werte für R,G,B den Dichte- bzw. Funktionswerten im jeweiligen Punkt (Grauwertverlauf). Experimentieren Sie ein wenig mit den Werten für A, z.B.  $A = \text{Dichtewert}$  oder  $A = 1.0 - \text{Dichte}$ .

Begründen Sie, warum das Blending der Texturen eine Näherung des Volumenintegrals darstellt.

**4 Punkte**

11. Schönere Ergebnisse können erzielt werden, wenn die Farb- und Alphawerte der Texturen modifiziert werden. Überlegen Sie sich eine einfache Methode, diese Werte durch Benutzereingabe zu verändern und implementieren Sie diese (z.B. mit einem Widget).

**4 Punkte**

### 12. Dreiecksnetz-Datenstrukturen

Beim Implementieren des Marching Cubes-Algorithmus aus der vorherigen Übung werden Sie vermutlich so vorgegangen sein, dass jedes der Dreiecke die Koordinaten der drei Eckpunkte speichert. Das bedeutet, dass die Koordinaten eines Eckpunkts mehrfach gespeichert werden, jeweils für jedes angrenzende Dreieck („Dreieckssuppe“). In dieser Aufgabe sollen Sie – falls noch nicht geschehen, andernfalls lassen Sie sich die Punkte gutschreiben – einen Algorithmus implementieren, der die Netze in eine *Indexed Face Set*-Datenstruktur überführt (alternativ: den MC-Algorithmus entsprechend modifizieren, so dass ein Indexed Face Set entsteht).

Überlegen Sie sich, wie Sie ein *set* aus der STL STD-Bibliothek auf einfache Weise für diese Zwecke einsetzen können (schreiben Sie einen passenden Vergleichsoperator für die Vertices).

**2 Punkte**

13. Implementieren Sie – soweit noch nicht geschehen, ansonsten siehe oben – effizientes Rendering der Dreiecksnetze in OpenGL direkt von der Datenstruktur aus Aufgabe 12. Erstellen Sie dazu zunächst Vertex Arrays und in einem zweiten Schritt Vertex Buffer Objects.

**2 Punkte**

14. Machen Sie sich mit einer Bibliothek vertraut, die fortgeschrittenere Netzdatenstrukturen wie z.B. Half Edge zur Verfügung stellt. Sie können hier auf eine Vielzahl vorhandener Bibliotheken zurückgreifen. Unterstützung können wir für CGAL [3] anbieten. Weitere Beispiele sind Halfedge Mesh Library [4], Advanced Surface Library [5] oder das Visualization Toolkit VTK [6]. Da VTK hiervon wahrscheinlich am schwierigsten zu verwenden ist (jedenfalls bis die Bibliotheken kompiliert und das erste lauffähige Programm geschrieben wurde), erhalten Sie bei Verwendung von VTK 2 Zusatzpunkte. Schreiben Sie mit Hilfe der Libraries Routinen, um Dreiecksnetze in Form von Index Face Sets in die Half Edge-Repräsentation zu überführen. Schreiben Sie ebenfalls Routinen, um aus der Half Edge-Repräsentation wieder Indexed Face Sets zu erhalten. Wenn Sie VTK verwenden, starten Sie die Suche nach geeigneten Datenstrukturen und Routinen in der Klasse `vtkPolyData`, zu finden in der Doxygen-generierten Dokumentation (<http://www.vtk.org/doc/nightly/html/classvtkPolyData.html>).

**4 Punkte**

15. **Subdivision**

Implementieren Sie mit Hilfe der von Ihnen ausgewählten Bibliothek mindestens 2 Subdivision-Verfahren für Dreiecksnetze, z.B. Loop, Butterfly oder  $\sqrt{3}$  Subdivision. Die Bibliotheken erlauben es zum Teil, die Verfahren ohne größere Kenntnisse direkt umzusetzen. Machen Sie sich dennoch mit den Verfahren vertraut, ausführliche Erklärungen finden Sie beispielsweise in [7].

**4 Punkte**

## Literatur

- [1] GDV2 Forum <http://www.fachschaft.informatik.tu-darmstadt.de/forum/viewforum.php?f=284>
- [2] M. Hadwiger et. al. *High-Quality Volume Graphics on Consumer PC Hardware*, Siggraph 2002 Course Notes 42. [http://www.cs.utah.edu/~jmk/sigg\\_crs\\_02/courses\\_0067.html](http://www.cs.utah.edu/~jmk/sigg_crs_02/courses_0067.html)
- [3] CGAL Computational Geometry Algorithms Library <http://www.cgal.org/>
- [4] Halfedge Mesh Library <http://www.cs.sunysb.edu/~gu/software/MeshLib/index.html>
- [5] Advanced Surface Library <http://www.cs.sunysb.edu/~gu/software/MeshLib/index.html>
- [6] VTK Visualization Toolkit <http://www.vtk.org/>
- [7] Siggraph 2000 course notes on subdivision <http://mrl.nyu.edu/~dzorin/sig00course/>