Fachhochschule Gießen-Friedberg

Prof. Dr.-Ing. A. Christidis

**Fachbereich Mathematik,
Naturwissenschaften und Informatik**

8. August 2008

Klausur
Computergrafik
SS 2008

– Lösungshilfe –

Personalien:

Name, Vorname: .....

Matrikelnummer: .....

Hinweise:

- **Die Bearbeitungszeit beträgt 90 Minuten.**
- **Alle schriftlichen Hilfsmittel sind zugelassen; andere Hilfsmittel, insb. elektr. Rechen- und Kommunikationsapparate, dürfen nicht verwendet werden.**
- **Ausgesprochene Folgefehler (durch Übertragung falscher Zwischenergebnisse) werden in Folgerechnungen als richtig gewertet.**
- **Die Aufgaben sollen nur auf diesen Blättern (inkl. Rückseite) bearbeitet werden. Bei Bedarf wird zusätzliches Papier zur Verfügung gestellt.**
- **Zur sicheren Zuordnung aller Lösungen wird um eine persönliche Kennung (Name u./o. Matrikelnr.) auf allen Blättern gebeten.**
- **Auf Wunsch darf auch Bleistift verwendet werden.**

Zur leichteren Lesbarkeit werden Substantive nur in einem Geschlecht („Nutzerin“) verwendet.

1. Aufgabe (30 Punkte)

a) Aufgrund Ihrer hervorragenden Ergebnisse in dieser Klausur werden Sie gebeten, die Eröffnungsrede zu einer Tagung über Bild- und Grafikverarbeitung zu übernehmen. Die Organisatoren bitten Sie, besonders für die Anfänger im Publikum zu betonen, daß die beiden Technologien weitgehend mit den gleichen Verfahren arbeiten, diese aber jeweils „in umgekehrter Richtung“ einsetzen.

Was ist damit gemeint? (Kurze Erklärung)

Beide Technologien verbinden Bilder mit Bildbeschreibungen. Die Bildverarbeitung erzeugt Bildbeschreibungen aus Bildern, die Computergrafik erzeugt umgekehrt Bilder aus Bildbeschreibungen.

b) Sie probieren ein neu gekauftes Zeichenprogramm für hochauflösende Grafik und ziehen damit eine Linie. Jemand, der Ihre guten Kenntnisse in Computergrafik schätzt, fragt Sie, ob die Linie nach dem Bresenham-Algorithmus gezogen wurde. Sie erklären ihm, daß man weder den Zeitvorteil bei nur einer Linie messen kann, noch die Linienführung mit bloßem Auge beurteilen kann. Dann vergrößern Sie das Bild so stark, daß es wie Abb. 1 aussieht.

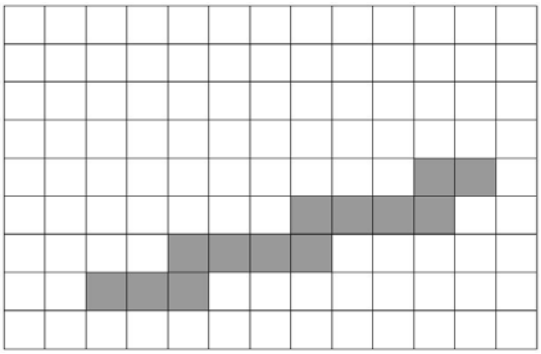


Abb. 1

Kann man anhand der Vergrößerung etwas über den zugrundeliegenden Algorithmus sagen (bestätigen oder ausschließen), oder war das eine nutzlose Maßnahme? (Begründung!)

Das kann nicht der Bresenham-Algorithmus sein: Die Linie hat eine Steigung von <45° und setzt beim Zeilenwechsel zwei Pixel übereinander – d.h., die Entscheidung wurde nicht (wie bei Bresenham) zwischen Ost und Nordost, sondern zwischen Ost und Nord getroffen.

c) Sie lernen einen Informatiker der ersten Generation kennen, der als Datenbank-Spezialist jahrzehntelang sehr erfolgreich im Dienst der internationalen Flugsicherheit gearbeitet hat. Sie erfahren, daß er berühmt wurde, weil er ein Verfahren entwickelte, das nach Eintippen einer Paßnummer mit extremer Geschwindigkeit ein bei den Sicherheitsbehörden gespeichertes Referenzfoto des Reisepaß-Besitzers mit grafisch markierten besonderen Merkmalen (Nasenform, Augenfarbe etc.) abruft.

Handelt es sich bei diesem großen Datenbank-Fachmann gleichzeitig um einen Experten für Bildverarbeitung, für Computergrafik, für beide oder weder für die eine, noch für die andere?

Bitte begründen kurz Sie Ihre Antwort!

Weder für die eine, noch für die andere: Die beschriebene Tätigkeit hat nichts mit visuellen Darstellungen zu tun (weder als Input, noch als Output – die abgerufenen Daten könnten beliebiger Natur sein).

d) Sie arbeiten mit Kollegen an der Entwicklung von Navigationsgrafik auf Displays mit einer Pixelgröße von 0,2mm x 0,2mm und einigen sich, daß das System nach dem Einschalten alle seine Subsysteme überprüfen und als Icons kachelartig auf seinem kleinen Bildschirm abbilden soll. Ihrem Subsystem hat man nur einen Platz von 30 x 40 Pixeln zugeordnet, und Sie beschließen, möglichst viele schräge Linien zu verwenden, damit sie auf dem kleinen Platz länger sind.

Wieviele Millimeter lang ist die längste Linie, die auf dem Platz Ihres Subsystems gezeichnet werden kann?

Diagonale nach Satz von Pythagoras in Längeneinheiten (LE, gleich für Breite und Höhe):

$d = (30^2\ LE^2 + 40^2\ LE^2)^{1/2} = (900 + 1600)^{1/2}\ LE = (2500)^{1/2}\ LE = 50\ LE$

Für LE= 0,2 mm $\Rightarrow$ $\underline{d =}$ 50 * 0,2 mm = $\underline{10\ mm}$ (= 1 cm)

e) Sie überprüfen mit dem Debugger eine Bresenham-Implementierung, indem Sie den Punkt (5,0) festhalten und immer neue Punkte mit ihm über eine Linie verbinden. Als Sie bei einer Stichprobe die Verbindung zu dem Punkt mit den Koordinaten (10,1) überprüfen, stellen Sie fest, daß sich dieser Rechengang nicht auf den 1. Oktanten bezieht; trotzdem erkennen Sie keinen Fehler an der gezogenen Linie.

Haben Sie eine Erklärung für diese Erscheinung?

Die Linie kann nur von (10,1) nach (5,0) gezogen worden sein.

Damit war nicht der 1., sondern der 5. Oktant beteiligt.

f) Die Reihenfolge, in der wir ein Objekt um die Raumachsen drehen, beeinflußt die Lage, in der es sich schließlich befindet: Drehung erst um die x- und anschließend um die y-Achse führt zu einer anderen Lage als Drehung erst um die y- und dann um die x-Achse. Mathematisch entspricht das der Tatsache, daß ...

	... das Distributivgesetz in der Matrizenrechnung gilt
	... nicht alle Matrizen invertierbar sind
	... Drehungen um mehrere Raumachsen nicht als Matrizenprodukt darstellbar sind
X	... die Matrizenmultiplikation nicht kommutativ ist
	... Rotationsmatrizen orthogonal sind
	... das Assoziativgesetz in der Matrizenrechnung uneingeschränkt gilt

(Bitte richtige Begründung/en ankreuzen!)

2. **Aufgabe** (40 Punkte)

Sie arbeiten mit an einem Projekt zur Steuerung eines virtuellen Roboters, der Bücher aus einem Regal holt und stapelt:

Auf einem Regalbrett der Länge **q** und der Tiefe **h**, dessen eine Ecke am Koordinaten-Ursprung und dessen Oberfläche in der x-z-Ebene liegt, steht ein Buch mit der Breite **b** und der Höhe **h**. Der (unendlich schmal angenommene) Buchrücken steht anfänglich am Regal-Rand (z_R=h) an der Stelle x_R=s (Abb. 2).

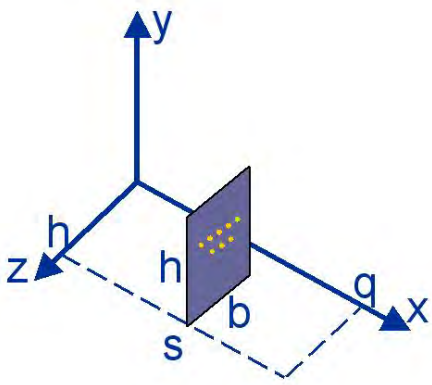


Abb. 2

Ein Roboterarm soll das Buch an dieser Position abholen und so hinlegen, daß der Buchrücken mit der positiven z-Achse zusammenfällt und die anfänglich untere Kante mit dem Regal abschließt (Abb. 3). Er benötigt dazu die Transformationsmatrix für die Koordinaten der Buch-Eckpunkte.

Sie beschließen, die Transformation in folgenden vier Schritten zu berechnen:

(i) Das Buch wird erst (aufrecht) auf dem Regalbrett soweit verschoben, bis es auf der negativen z-Achse steht und der Buchrücken mit der (positiven) y-Achse zusammenfällt. **(ii)** Anschließend wird es um die y-Achse (Winkel α) gedreht, bis es auf der positiven x-Achse steht. **(iii)** Dann dreht es sich um die x-Achse (Winkel β), bis sein Rücken an der negativen z-Achse liegt. **(iv)** Schließlich wird es (liegend) entlang der z-Achse bis zur o.a. Position (Abb. 3) verschoben.

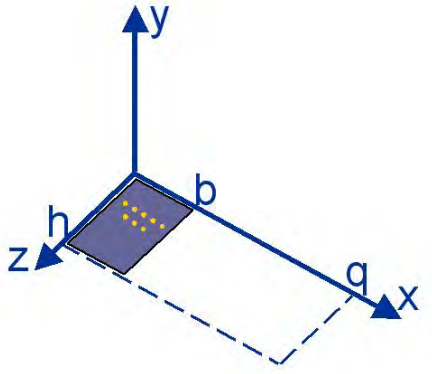


Abb. 3

Lösen Sie diese Aufgabe, indem Sie nacheinander folgende Fragen bearbeiten:

a) Wie lautet die Transformationsmatrix $\underline{\mathbf{T}}_{(i)}$ zum o.a. Schritt (i), mit dem das untere Ende des Buchrückens an den Koordinaten-Ursprung verschoben wird?

$$\underline{\mathbf{T}}_{(i)} = \begin{pmatrix} 1 & 0 & 0 & -s \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -h \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

b) Geben Sie (in Grad, unter Berücksichtigung des Drehsinns) den Winkel **α** an, um den in Schritt (ii) das Buch um die y-Achse gedreht werden muß, bis es auf der positiven x-Achse steht. Wie groß sind dann **sin α** und **cos α**?

α = ***– 90°*** **sin α =** ***–1*** **cos α =** ***0***

c) Wie lautet nun die Transformationsmatrix $\underline{T}_{(ii)}$ zu Schritt (ii), nach welchem das Buch auf der positiven x-Achse steht? Geben Sie sie bitte sowohl in symbolischer (sin α, ...) als auch in arithmetischer (zahlenmäßiger) Form an!

$$\underline{T}_{(ii)}(\alpha) = \begin{pmatrix} \cos\alpha & 0 & \sin\alpha & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\alpha & 0 & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

d) Geben Sie (entsprechend Frage (b), in Grad, unter Berücksichtigung des Drehsinns) den Winkel **β** an, um den in Schritt (iii) das Buch um die x-Achse geklappt wird, bis sein Rücken an der negativen z-Achse liegt. Wie groß sind **sin β** und **cos β**?

β = *– 90°* **sin β =** *–1* **cos β =** *0*

e) Wie lautet nun die Transformationsmatrix $\underline{T}_{(iii)}$ zum o.a. Schritt (iii)? Geben Sie sie bitte wieder in symbolischer und in arithmetischer Form an!

$$\underline{T}_{(iii)}(\beta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\beta & -\sin\beta & 0 \\ 0 & \sin\beta & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

f) Wie lautet schließlich die Transformationsmatrix $\underline{T}_{(iv)}$ zum o.a. Schritt (iv), mit dem das untere Ende des Buchrückens entlang der z-Achse an den Regal-Rand verschoben wird?

$$\underline{T}_{(iv)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & h \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

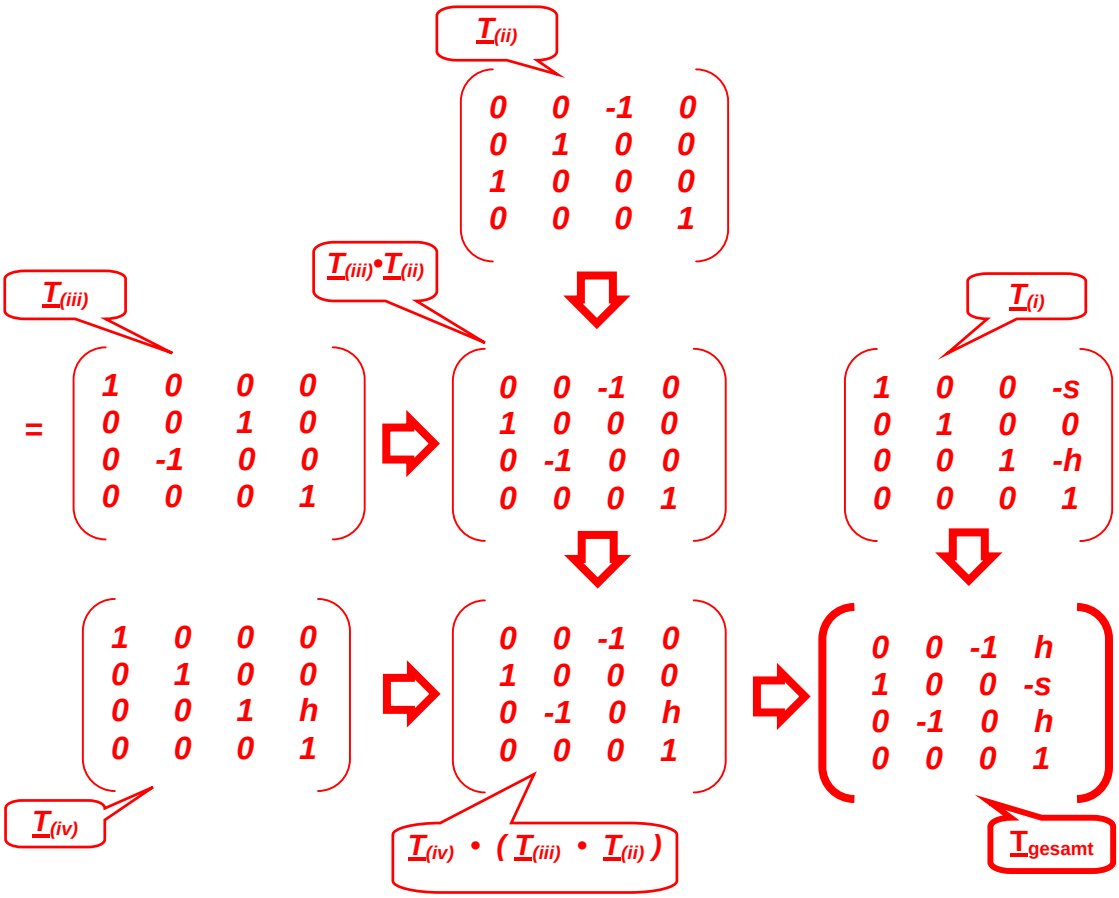
g) Wie berechnet sich die gesuchte Transformationsmatrix für die Koordinaten der Buch-Eckpunkte $\underline{T}_{gesamt}$ aus den bisher besprochenen Transformationen $\underline{T}_{(i)}$ bis $\underline{T}_{(iv)}$?

$$\underline{T}_{gesamt} = \underline{T}_{(iv)} \cdot \underline{T}_{(iii)} \cdot \underline{T}_{(ii)} \cdot \underline{T}_{(i)}$$

h) Berechnen Sie jetzt bitte die gesuchte Transformationsmatrix $\underline{T}_{gesamt}$ nach den obigen Angaben.

(Tip: Es ist meist einfacher, zunächst die zahlenmäßig vorliegenden Matrizen zusammenzufassen und danach dieses Zwischenergebnis erst mit den schwächer, dann mit den stärker besetzten Symbol-Matrizen zu verknüpfen.)

$$\underline{T}_{gesamt} = [\, \underline{T}_{(iv)} \cdot (\underline{T}_{(iii)} \cdot \underline{T}_{(ii)}) \,] \cdot \underline{T}_{(i)}$$



i) Welche Koordinaten x_{iStart}, y_{iStart}, z_{iStart} (I=1,…4) hatten die vier Eckpunkte des Buches vor der Transformation, welche danach?

Lesen Sie bitte die entsprechenden Angaben aus der Aufgabenstellung und der Abb. 2 ab, und weisen Sie durch Anwendung von $\underline{T}_{gesamt}$ nach, daß die von Ihnen ermittelte Matrix die vier Ecken entsprechend Abb. 3 transformiert.

(Sie können die Punkte zu einer 4x4-Matrix zusammenfassen.)

$$\begin{pmatrix} x_{iStart} \\ y_{iStart} \\ z_{iStart} \\ 1 \end{pmatrix} = \begin{pmatrix} s & s & s & s \\ 0 & 0 & h & h \\ h & h-b & h-b & h \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

$$\Downarrow$$

$$\begin{pmatrix} 0 & 0 & -1 & h \\ 1 & 0 & 0 & -s \\ 0 & -1 & 0 & h \\ 0 & 0 & 0 & 1 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & b & b & 0 \\ 0 & 0 & 0 & 0 \\ h & h & 0 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} x_{iEnde} \\ y_{iEnde} \\ z_{iEnde} \\ 1 \end{pmatrix}$$

j) An welchen Stellen in den obigen Berechnungen spielte die Regallänge **q** eine Rolle? Überprüfen Sie die Zusammenhänge, in denen Sie diese Größe verwendet haben, und kommentieren Sie kurz, welche Schlüsse Sie daraus ziehen:

Die Größe q wurde nirgends benötigt: Es ist auch physikalisch (für Roboter wie für Menschen) vollkommen belanglos, wie lang das Regal ist, aus dem ein Buch geholt wird.

3. Aufgabe (15 Punkte)

Sie wollen am PC Trickfilme mit OpenGL und GLUT-Fenstern entwickeln und klären zunächst folgende Frage:

a) GLUT ist als Bibliothek nicht als Open-Source-Gemeinschaftsprojekt, sondern von einem Entwickler (M.J. Kilgard) entstanden. Müssen dafür Gebühren entrichtet werden (i) für die Nutzung bzw. (ii) für den Quellcode, falls er nötig sein sollte? (Antwort ohne Begründung / ohne Gebührenhöhe genügt.)

GLUT ist gebührenfrei inkl. Code erhältlich.

Für Ihr Grafik-Projekt erwarten Sie die Unterstützung von einem erfahrenen Freund, und Sie fühlen sich gleich ermutigt, eine Entwicklung mit zwei Fenstern auszuprobieren:

```
int main(int argc, char **argv)
{ glutInit(&argc, argv);
  glutInitWindowSize(200,200);
  glutCreateWindow("CG goes OpenGL");
  glutDisplayFunc(draw);
  glutKeyboardFunc(key);
  glutCreateWindow("CG goes OpenGL");
  glutDisplayFunc(draw);
  init();
  glutMainLoop();
  return 0;
}
```

Das konfrontiert Sie mit folgenden Fragen:

b) Als Ergebnis Ihres Experimentes erhalten Sie zwei gleich große Fenster, obwohl in Ihrem `main()` nur einmal die Fenstergröße gewählt wird. Ist das Folge der Design-Philosophie von GLUT (wenn ja: welches Grundsatzes?), oder handelt es sich hier um eine Standard-Größe von GLUT?

Das ist Folge der Design-Philosophie: GLUT ist ein zustandshaftes System: einmal eingestellte (z.B.: Fenster-)Größen gelten, solange sie nicht geändert werden.

c) Die beiden Fenster haben nicht nur denselben (zweimal eingestellten) Titel; sie haben vielmehr auch denselben Inhalt (Abb.4). Wie konnte GLUT erkennen, daß dieselbe Funktion für die Auffrischung beider Fenster gedacht war?

Hätte evtl. auch ein Aufruf von `glutDisplayFunc()` gereicht?

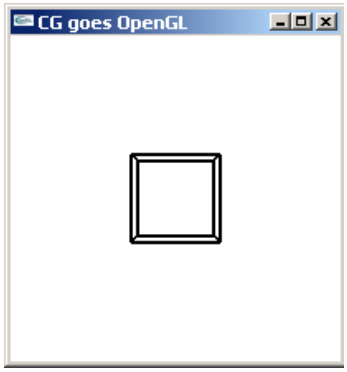


Abb. 4

Als zustandshaftes System registriert GLUT auch das jeweils aktuelle Fenster: Es wird erstmalig mit `glutCreateWindow()` gesetzt, und alle nachfolgenden Einstellungen gelten diesem Fenster. Damit werden im Code die beiden `glutDisplayFunc()`-Aufrufe korrekt zugeordnet.

Beide Aufrufe sind deshalb auch nötig.

d) Sie versuchen, das abgebildete Objekt über die Tastatur zu bewegen (Abb. 5) und merken, daß nur in einem der Fenster etwas geschieht, während der Inhalt des anderen unverändert (wie Abb. 4) bleibt.

Als Ihr Freund rät, eine Ecke des nicht-aktualisierten Fensters zu verdecken, stellen Sie fest, daß danach beide Fenster denselben Inhalt (Abb. 5) haben.

Welches der beiden identisch aussehenden Fenster richtet sich nach der Tastatur, welches nicht – und wie erklären Sie den Rat Ihres Freundes anhand des o.a. Codes?

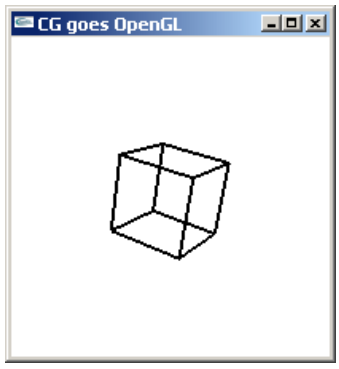


Abb. 5

GLUT hatte (als zustandshaftes System) den einzigen `glutKeyboardFunc()`-Aufruf dem ersten (beim Aufruf aktuellen) Fenster zugeordnet; das zweite war davon nicht betroffen. Unabhängig davon bekamen aber beide Fenster ihren Inhalt von derselben Auffrischungs-Funktion. Der „Trick“ mit der kurzen Verdeckung sollte das Fenster zwingen, sich aufzufrischen.

4. Aufgabe (15 Punkte)

a) In einem Lehr-Projekt mit Computergrafik setzen Sie die Programm-Bibliothek OpenGL ein und wollen möglichst schnell Teile Ihres Programms auf Eingebettete Systeme portieren. Dafür ist die ganze Bibliothek zu groß. Sie brauchen C-Quellcode, um ihn den Bedürfnissen des Projektes anzupassen. Müssen Sie dann die OpenGL-Funktionalitäten selbst programmieren, können Sie auf Open-Source-Code ausweichen (wenn ja: welchen?), oder können Sie den Code von OpenGL frei bekommen (wenn ja: woher)?

Sie können auf anderen Code ausweichen: „The Mesa 3D Graphics Library“ ist die Open-Source-Implementierung der OpenGL-Spezifikation.

b) Ihr OpenGL-Projekt stellt dreidimensionale Flächenmodelle dar und strukturiert ihre Oberflächen mit der Musterungsfunktionalität von OpenGL („Stippling“). Beantworten Sie bitte dazu folgende Fragen:

(i) Welche Fläche (Pixel x Pixel) belegt (ohne Wiederholungen) die Grundform eines solchen Füllmusters? (ii) Wieviele Farben kann es (minimal / maximal) enthalten? (iii) Wieviel Speicherplatz (in Bit oder Byte) beansprucht es (minimal / maximal)? (iv) Wie lauten die Anweisungen zur Aktivierung bzw. Deaktivierung dieser OpenGL-Funktionalität? (Angabe der beiden Befehle genügt.)

i) genau 32 x 32 Pixel

ii) genau 2 (1 Bit / Pixel)

iii) genau 32 x 32 Bit (= 1.024 Bit = 128 Byte)

iv) glEnable (GL_POLYGON_STIPPLE);
glDisable (GL_POLYGON_STIPPLE);

c) In Ihrem Projekt verwenden Sie zwei Muster:

```
GLubyte pattern1[] = {
0xAA, 0xAA, 0xAA, 0xAA, 0x55, 0x55, 0x55, 0x55,
0xAA, 0xAA, 0xAA, 0xAA, 0x55, 0x55, 0x55, 0x55,
       /* (...ueber mehrere Zeilen...) */
0xAA, 0xAA, 0xAA, 0xAA, 0x55, 0x55, 0x55, 0x55,
0xAA, 0xAA, 0xAA, 0xAA, 0x55, 0x55, 0x55, 0x55};

GLubyte pattern2[] = {
0x0F, 0x0F, 0x0F, 0x0F, 0xF0, 0xF0, 0xF0, 0xF0,
0x0F, 0x0F, 0x0F, 0x0F, 0xF0, 0xF0, 0xF0, 0xF0,
       /* (...ueber mehrere Zeilen...) */
0x0F, 0x0F, 0x0F, 0x0F, 0xF0, 0xF0, 0xF0, 0xF0,
0x0F, 0x0F, 0x0F, 0x0F, 0xF0, 0xF0, 0xF0, 0xF0};
```

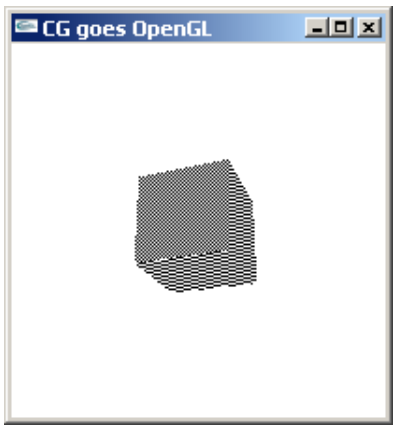


Abb. 6

In Abb. 6 belegen diese Muster jeweils eine (feinkörnig gemusterte) bzw. zwei (grob strukturierte) Objektflächen.

Welche Struktur ordnen Sie `pattern1[]`, welche `pattern2[]` zu? Woran erkennen Sie das?

Es gilt einerseits $A_{16}=1010_2$ u. $5_{16}=0101_2$, andererseits $F_{16}=1111_2$ u. $0_{16}=0000_2$ d.h., `pattern1` ist das fein, `pattern2` das grob strukturierte Muster auf den zwei Flächen (einzelne vs. mehrere gesetzte / ausgelassene Punkte hintereinander)

d) Beide Muster bestehen aus Zeilen, die sich identisch wiederholen. Wie ist es zu erklären, daß sie keine Streifen oder andere Artefakte („Muster im Muster“) bilden?

Die o.a. Muster sind so konzipiert, daß sie zeilenweise abwechselnd Nullen und Einsen enthalten. (Der Muster-Code (`pattern1[]` und `pattern2[]` enthält jeweils zwei Muster-Bildzeilen je Code-Zeile).

Platz für Notizen: