Fachhochschule Gießen-Friedberg

Prof. Dr.-Ing. A. Christidis

**Fachbereich Mathematik,
Naturwissenschaften und Informatik**

11. Oktober 2007

Klausur
Computergrafik
SS 2007

– Lösungshilfe –

Personalien:

Name, Vorname: .....

Matrikelnummer: .....

Hinweise:

- **Die Bearbeitungszeit beträgt 90 Minuten.**
- **Alle schriftlichen Hilfsmittel sind zugelassen; andere Hilfsmittel, insb. elektronische Rechen- und Kommunikationsapparate dürfen nicht verwendet werden.**
- **Die Aufgaben sollen nur auf diesen Aufgabenblättern bearbeitet werden. Bei Bedarf kann zusätzliches Papier zur Verfügung gestellt werden.**
- **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 (15 Punkte)

a) Ihr Lieblingsautor hat ein Buch über Computergrafik und eins über Bildverarbeitung geschrieben. Sie finden beide Bücher aufgeschlagen auf einem Tisch in der Bibliothek und stellen fest, daß in beiden ein Kapitel mit der Überschrift „Figuren und ihre Beschreibung“ enthalten ist, das jeweils mit Abb.1 beginnt. Erwarten Sie darunter den gleichen Text?

Wenn ja: Welche Fragen könnte dieser Text behandeln?

Wenn nein: Welche Unterschiede erwarten Sie zwischen den beiden Büchern?

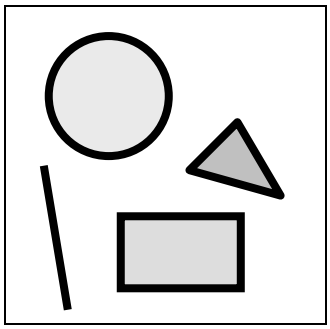

Abb. 1 Figuren und ihre Beschreibung

Nein. Das Buch über CG wird die Abbildung (Erzeugung) einer Figur aus ihrer Beschreibung enthalten, das Buch über BV wird die Gewinnung einer Beschreibung (oder die Erkennung) einer bereits abgebildeten Figur zum Thema haben.

b) In einem Grafik-System werden die Punkte $p_i=[x_i, y_i, z_i]$ einer Punktwolke mit zufälligen Koordinaten um die z-Achse gedreht. Dadurch verändern sich ...

	... alle Punkt-Koordinaten
X	... nur die x- und y-Koordinaten
	... nur die y- und z-Koordinaten
	... nur die x- und z-Koordinaten
	... nur die z-Koordinaten

(Bitte richtige Antwort/en ankreuzen!)

c) Aus wievielen Pixeln besteht die Diagonale eines Quadrats mit einer Kantenlänge von 25 Pixeln?

Aus 25.

d) Verstehen Sie unter einer „Brute-Force-Methode“ ein Verfahren, ...

	... das mit „roher Gewalt“ die Gesetze weder der Mathematik noch der Ergonomie beachtet, um für eine kurze Vorführung täuschend echt die Ergebnisse einer exakten Rechnung zu imitieren?
X	... das alle Möglichkeiten ausprobiert und dann zum Ergebnis führt, ohne auf Ressourcen-Verbrauch zu achten?
	... das schnell zu programmieren ist und erste Eindrücke liefert, ohne aber die Sicherheit zu bieten, daß es immer zum Ergebnis führt?

(Bitte richtige Antwort/en ankreuzen!)

e) Der Bresenham-Algorithmus zur Erzeugung gerader Linien (von „Südwest“ nach „Nordost“) auf digitalen Ausgabemedien verwendet eine sog. „Entscheidungsvariable“. Welche Entscheidung wird anhand dieser Variablen getroffen?

Ob das nächste zu setzende Pixel im Osten oder Nordosten des zuletzt gesetzten liegt.

f) Zur mathematischen Beschreibung des gesamten Verlaufs einer Geraden reicht es aus, die Koordinaten von Anfangs- und Endpunkt zu wissen. Welche weitere/n Größe/n geht/gehen in den Zahlenwert der Entscheidungsvariable ein – bzw.: Warum wird der Bresenham-Algorithmus nicht in Büchern für klassische Mathematik oder Geometrie erwähnt? (Kurze Antwort genügt)

Als weitere Größen gehen in den Zahlenwert der Entscheidungsvariable ein die Anzahl der vorausgegangenen Entscheidungen, Pixel im Osten oder Nordosten zu setzen; sie sind nicht berechenbar, sondern nur abzählbar – und insofern für Bücher der klassischen Mathematik oder Geometrie uninteressant.

2. Aufgabe (15 Punkte)

Eine Linie wird mit der in der Vorlesung behandelten Version des Bresenham-Algorithmus zwischen den Punkten mit den Koordinaten (5,10) und (15,17) gezogen.

a) Auf welchen Oktanten bezieht sich diese Anwendung des Linien-Algorithmus? (Nennung genügt.)

Auf den 1.

b) Tragen Sie bitte in die nachstehende Liste die Koordinaten-Werte der ersten zwei Punkte ein, die nach dem Startpunkt gesetzt werden, sowie den jeweils aktuellen Wert der Entscheidungsvariablen.

(Bei Bedarf kann die karierte Fläche auf dem letzten Blatt als Schmierpapier genutzt werden; sie wird nicht ausgewertet.)

Punkt-Nr.	„decision variable“	x-Koordinate	y-Koordinate
1.	***+4***	5	10
2.	***-2***	***6***	***11***
3.	***+12***	***7***	***11***

c) Der Linienalgorithmus ist in seiner Grundform für nur einen Oktanten konzipiert und wird unter Nutzung von Symmetrien auf die anderen Oktanten angewandt. Für welchen Start- und welchen Endpunkt würde dieser Algorithmus im 8. Oktanten arbeiten und der Entscheidungsvariablen in derselben Reihenfolge dieselben Zahlenwerte zuweisen?

	x-Koordinate	y-Koordinate
Startpunkt	***+5***	***-10***
Endpunkt	***+15***	***-17***

3. Aufgabe (40 Punkte)

Zwei Punkte mit den Koordinaten (3, 2, 0) und (8, 7, 0) werden nacheinander

- um den Winkel θ_x=180° um die x-Achse gedreht (Transformation $\mathbf{T_{Rx}}$);
- um t_y=4 Längeneinheiten in (positiver) y-Richtung verschoben (Transformation $\mathbf{T_{Ty}}$);
- um eine Achse parallel zur z-Achse durch den Punkt (3, 2, 0) um den Winkel θ_z= – 90° gedreht (Transformation $\mathbf{T_{Rz}}$);
- um t_x=5 in x-, um t_{y2}=5 in y-Richtung verschoben (Transformation $\mathbf{T_{Txy}}$).

(Die Werte der trigonometrischen Winkelfunktionen zu den hier vorkommenden Winkeln werden als bekannt vorausgesetzt.)

Beantworten Sie bitte folgende Fragen:

a) Bedeutet die Angabe „Winkel θ_z= – 90°“ einen Winkel im Uhrzeigersinn oder gegen den Uhrzeigersinn?

Im Uhrzeigersinn

b) Von welcher Seite muß man eine Koordinaten-Achse betrachten, damit die mathematische Aussage „positiver/negativer Rotationswinkel“ mit dem visuellen Eindruck „Drehung gegen den / im Uhrzeigersinn“ übereinstimmt? (Schaut man dabei z.B. von der Seite positiver oder negativer Werte auf den Koordinaten-Ursprung?)

Schaut man vom Zahlenstrahl positiver Werte auf den Koordinaten-Ursprung einer Achse, so deckt sich die Aussage „positiver (negativer) Winkel“ mit der Aussage „Rotation gegen den (im) Uhrzeigersinn“.

Berechnen Sie nun die Endposition der beiden Punkte, indem Sie folgende Fragen behandeln:

c) Geben Sie die Transformatiosmatrix $\mathbf{T_{Rx}}$ (symbolisch und arithmetisch) an:

$$\mathbf{T_{Rx}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x & 0 \\ 0 & \sin\theta_x & \cos\theta_x & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

d) Geben Sie die Transformatiosmatrix $\mathbf{T_{Ty}}$ an (Zahlenform genügt):

$$\mathbf{T_{Ty}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

e) Ermitteln Sie (erst symbolisch, dann arithmetisch) die Transformationsmatrix $\mathbf{T_{Rz}}$:

$$\mathbf{T_{Rz}} = \begin{pmatrix} 1 & 0 & 0 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta_z & -\sin\theta_z & 0 & 0 \\ \sin\theta_z & \cos\theta_z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & -3 \\ 0 & 1 & 0 & -2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & -3 \\ 0 & 1 & 0 & -2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 0 & 1 & 0 & 1 \\ -1 & 0 & 0 & 5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

f) Geben Sie die Transformatiosmatrix $\mathbf{T_{Txy}}$ an (Zahlenform genügt):

$$\mathbf{T_{Txy}} = \begin{pmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

g) Wie lassen sich die o.a. Transformationen zu einer Gesamttransformation zusammenfassen? Bilden Sie den dazugehörigen mathematischen Ausdruck (Formel) und werten Sie ihn aus:

$$T_{gesamt} = T_{Txy} \bullet T_{Rz} \bullet T_{Ty} \bullet T_{Rx}$$

$$= \begin{pmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 1 \\ -1 & 0 & 0 & 5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 1 \\ -1 & 0 & 0 & 5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 4 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & -1 & 0 & 5 \\ -1 & 0 & 0 & 5 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 0 & -1 & 0 & 10 \\ -1 & 0 & 0 & 10 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

h) Berechnen Sie nun bitte die Koordinaten der zwei Punkte nach allen o.a. Transformationen:

$$\begin{pmatrix} 0 & -1 & 0 & 10 \\ -1 & 0 & 0 & 10 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 3 & 8 \\ 2 & 7 \\ 0 & 0 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 8 & 3 \\ 7 & 2 \\ 0 & 0 \\ 1 & 1 \end{pmatrix}$$

4. Aufgabe (15 Punkte)

Sie helfen einem Freund, ein Grafik-Programm zu entwickeln und klären zunächst einige Fragen bezüglich der GLUT-Bibliothek, die er in `main()` einsetzt:

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

Beantworten Sie bitte folgende Fragen:

a) Wofür steht das Akronym „GLUT“, und was bedeutet das auf deusch?

„GLUT” steht für „OpenGL Utility Toolkit”; das bedeutet etwa: „OpenGL-Alltagswerkzeug“ oder auch „-Rüstzeug“.

b) Was bewirkt die Anweisung `glutCreateWindow()` und was ist von dem Fenster zu sehen, nachdem diese Programmzeile ausgeführt wurde (d.h.: während der Ausführung der darauffolgenden Zeile)?

`glutCreateWindow()` richtet das GLUT-Fenster ein, d.h., es bereitet das System (Speicher etc.) darauf vor, ein (weiteres) Fenster zu verwalten. Bis zum Aufruf von `glutMainLoop()` ist vom Fenster nichts zu sehen.

c) Wie groß ist das Fenster, das GLUT in `main()` erzeugt, und woran erkennen Sie das?

300 x 300 Pixel; das ist die Voreinstellung (default) und gilt hier, weil nichts anderes vereinbart wurde (kein Aufruf von `glutInitWindowSize()`).

d) Verwendet das Programm Double Buffering? Woran erkennen Sie das?

Nein, es setzt Single Buffering ein; das ist die Voreinstellung (default) und gilt hier, weil nichts anderes vereinbart wurde (kein Aufruf von `glutInitDisplayMode()`).

e) Was bedeutet der Aufruf `glutKeyboardFunc(key)`?

`key()` ist die Callback-Funktion der Tastatur; sie wird immer dann aufgerufen, wenn eine (reguläre Schriftzeichen-)Taste gedrückt wird.

f) `main()`-Funktionen, die GLUT einsetzen, enthalten praktisch nie einen `exit()`-Aufruf. Woran liegt das?

Das hängt mit der Arbeitsweise von `glutMainLoop()` zusammen: Vor ihrem Aufruf würde `exit()` das Programm beenden, bevor ein Grafik-Fenster zu sehen wäre. Nach ihrem Aufruf kehrt die Kontrolle nicht mehr an `main()` zurück, `exit()` bliebe wirkungslos.

g) Das Programm soll offenbar eine Funktion namens `draw()` verwenden.

(i) Was soll GLUT mit `draw()` tun, und wie heißen Funktionen, die eine solche Rolle gegenüber Programmierschnittstellen oder -umgebungen übernehmen?

(ii) Erklären Sie kurz, was die Ausführung die Zeile:

```
glutDisplayFunc(draw);
```

bewirkt. Wann wird `draw()` zum ersten Mal aufgerufen?

(i) `draw()` ist die Callback-Funktion zur Auffrischung des aktuellen Fensters; sie wird immer aufgerufen, wenn das GLUT-Fenster neu gezeichnet werden soll (z.B. zur Animation oder nach Verdeckung).

(ii) Dies wird GLUT mit dem Aufruf `glutDisplayFunc(draw);` gemeldet. Der Aufruf von `draw()` erfolgt erst, wenn das Fenster aufgebaut werden soll (also nach `glutMainLoop();`).

5. **Aufgabe** (15 Punkte)

Zum o.a. `main()` schreiben Sie nun die Funktion `draw()` und wollen damit schwarz-weiße Drahtmodelle (Abb.2) und Flächenmodelle nach Abb. 3 erzeugen.

Abb. 2

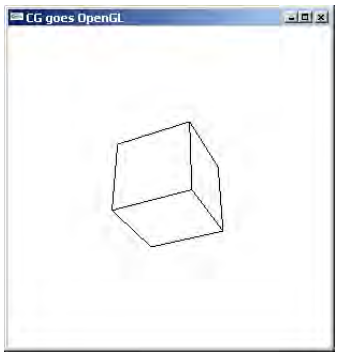

Abb. 3

Abb. 2: Gewünschte Draht-modell-Darstellung

Abb. 3: Gewünschte Flächen-modell-Darstellung

Sie experimentieren mit `draw()`und gehen dabei auf folgende Fragen ein, wobei es unwichtig ist, ob die betrachteten Variablen lokal oder global (und deshalb nicht im Auszug von `draw()` auf der nächsten Seite) sind:

a) Mit welchen Anweisungen werden die Zeichen- und die Löschfarbe in `draw()` festgelegt? Schreiben Sie sie bitte so, wie sie benötigt werden, damit Abb. 3 entstehen kann:

```
glClearColor (1.0, 1.0, 1.0, 1.0);
glColor3f(0.0, 0.0, 0.0);
```

b) Welche Anweisung bewirkt, daß die vom Augenpunkt abgewandten Flächen eliminiert werden? Welche Variable aktiviert diese Anweisung, und was für Werte hat sie dann?

Die Anweisung: `if(backF) glEnable(GL_CULL_FACE);`

Dazu muß die Variable `backF` Werte ungleich Null haben.

Sie stellen fest, daß nach Klärung der obigen Fragen schon die u.a. Version ein Bild nach Abb. 3 erzeugt. Die Umschaltung auf Drahtmodell (mit Darstellung auch der abgewandten Flächen) bereitet aber Probleme, denn sie hat Abb. 4 zum Ergebnis, weshalb Sie nun dieser Frage nachgehen müssen:

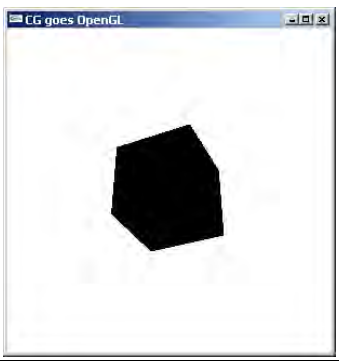

Abb. 4 Fehlerhafte Drahtmodell-Ausgabe

c) Sie erkennen schnell, daß genau eine Zeile dafür verantwortlich ist.
Wie sollte sie richtig lauten?

glPolygonMode(GL_FRONT_AND_BACK, GL_LINE);

d) Erklären Sie kurz das Programm-Verhalten, das zum Ergebnis nach Abb. 4 führte. Wieso ist die Darstellung des Flächenmodells korrekt, das Drahtmodell aber falsch? Welche Rolle spielt dabei die Realisierung von OpenGL als Zustandsautomat mit Voreinstellungen?

Der Aufruf glPolygonMode(GL_FRONT, GL_LINE) stellt die Vorderflächen auf Liniendarstellung ein. Damit ist nichts über die Rückseiten ausgesagt, und somit gilt die Voreinstellung des Zustandsautomaten; diese ist GL_FILL (bei Zeichenfarbe Schwarz).

```
void draw(void)
{ int jj;
  nah=eyez; fern=nah+2*diag;

  glClearColor (1.0, 1.0, 1.0, 1.0);
  glColor3f(1.0, 1.0, 1.0);

  glMatrixMode(GL_PROJECTION);
  glLoadIdentity();
  glFrustum (-diag, diag, -diag, diag, nah, fern);

  if (backF) glEnable  (GL_CULL_FACE);
  else       glDisable (GL_CULL_FACE);

  /*Vorerst nur als Drahtmodell:*/
  glPolygonMode(GL_FRONT, GL_LINE);

  glClear(GL_COLOR_BUFFER_BIT);

  /*...*/
}
```

Abb. 5 Auszug aus der Funktion `draw()` zur Behandlung dieser Aufgabe

Platz für Notizen: