Übung Nr. 4:

Im Konsole-Fenster sollen mit ASCII-Zeichen dreidimensionale Grafik-Modelle dargestellt werden (Abb.1).

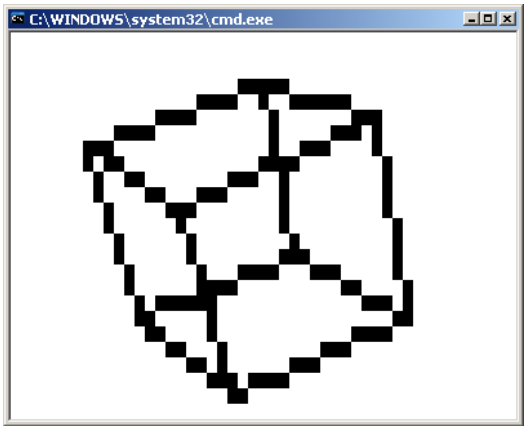

Abb. 1: `WireCull.exe`

Unter http://homepages.thm.de/christ/ kann das dazugehörige MS-VC-Projekt „`WireCull`“ heruntergeladen werden. Interessant für diese Übung sind, neben den bereits bekannten, die unten aufgeführten Dateien. Sie sind zu vervollständigen (insg. ca. 20 Zeilen Code), damit das Programm die beschriebene Funktionalität aufweist.

Zur Erleichterung der Arbeit sind die Lücken gegenüber der Original-Version des Autors mit Präprozessor-Statementpaaren (`#ifdef` und `#endif`) gekennzeichnet, wobei die Auswahl der konkreten Stellen für das Einfügen der Code-Zusätze je nach eigenem Verständnis der jeweiligen Aufgabe eine individuelle Entscheidung bleibt.

`GrafOps.c`
`GrafOps.h`
	beinhalten die Funktionen aus der 2. Übung, zu denen zwei weitere (insg. ca. 4 Zeilen Code, s. Bezeichner `MORE_GRAPH2`) hinzutreten sollen:
	`int perspTrf_z0 (float eyez, float *posMat)`
	soll die Positionierungsmatrix eines Objekts mit der Projektionsmatrix multiplizieren (s. Vorlesungsfolien);
	`int backFcull (float *vrtx1, float *vrtx2, float *vrtx3, float *eye)`
	soll anhand dreier Punkte einer Objektfläche entscheiden, ob die Fläche dem Augenpunkt zu- oder abgewandt (und somit nicht sichtbar) ist.

`MatrOps.c` `MatrOps.h`	Darin sind die Matrizen-Operationen aus der 2. Übung. Hinzu kommen drei weitere, die noch z.T. komplettiert werden müssen (Bezeichner `MORE_MATR2`). Bereits „gebrauchsfertig“ ist die Funktion zur gewichteten Addition zweier Matrizen: `int matAdd (float f1, float *mi1, int z12, int s12, float f2, float *mi2, float *mout)`

	Ergänzt werden müssen: die Funktion für das gewichtete Skalarprodukt zweier Vektoren (in der eine einzeilige `for` -Schleife fehlt):
	`int spV (float f1, float *vi1, float f2, float *vi2, float *out)`
	sowie die Berechnung des gewichteten Vektorprodukts (in der auch eine einzeilige `for` -Schleife und zwei weitere Statements fehlen):
	`int vpV (float f1, float *vi1, float f2, float *vi2, float *vout)`

`WireCull.c`
`WireCull.h`
	Hier ist das eigentlich zu entwickelnde Programm; folgende Ergänzungen sind vorzunehmen (Bezeichner `MORE_WIRE`):
	In der Funktion
	`void conViewport (float *posMat)`
	fehlt ein (im Kommentar angekündigter) Aufruf der Skalierungsfunktion. In
	`int conDrawCGFobj (CGFobject *obj, float eyez)`
	fehlt zudem die Überprüfung der Sichtbarkeit für die zu zeichnenden Flächen (ca. 5 Zeilen). Vor allem fehlt aber die darauf folgende Zeichnung des Objekts (ca. 6 Zeilen).
	Aus der Funktion
	`void conBBox(CGFobject *obj)`
	fehlen schließlich die (ca. 6) Zeilen zur Ermittlung der maximalen Ausdehnung des geladenen Objektes.

Folgende Vorgehensweise wird empfohlen:

Ausbau von `WireCull.c` zum „Draht- und Flächenmodell-Viewer“

Hier muß zwangsläufig mit dem Ausbau von `conDrawCGFobj()` begonnen werden. Die Flächenpunkte des geladenen Objekts müssen durch Linien verbunden werden. Damit wird erstmalig etwas vom geladenen Objekt sichtbar.

Nun sollte mit der Funktion `conBBox()` fortgefahren werden, weil hier die Voraussetzung für eine korrekte Skalierung geschaffen wird. Zur Unterstützung der eigenen Anschauung sollte man nachvollziehen, wie die Variable `viewScale` eingesetzt wird; sie wird in `conViewport()` benötigt. Bei korrekter Skalierung ist das geladene Dreieck wie in Abb. 3 bildfüllend zu sehen und zu animieren. (Versuchen Sie, nachzuvollziehen, wo und wie dies geschieht!)

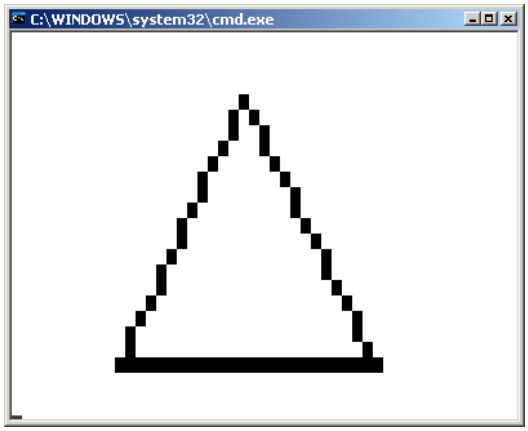

Abb. 3: `WireCull.exe`

Druck auf die ‚h'-Taste zeigt die im Programm implementierten Möglichkeiten. Von ihnen läßt sich u.a. die Veränderung des Augenpunkts noch nicht bedienen. Aktiviert man die Option `#define MIT_LOADCGF`, so wird beim Laden dreidimensionaler Objekte deutlich, daß momentan eine Parallelprojektion dargestellt wird. Abhilfe schafft Vervollständigung von `perspTrf_z0()`. (Was passiert da?)

Die verbleibende Funktionalität der Eliminierung der vom Augenpunkt abgewandten Objektflächen (Backface Culling) macht die o.a. Ergänzungen in `MatrOps.c` notwendig. Die Fertigstellung sollte nach den oben und den im Code gewährten Hinweisen und unter Zuhilfenahme der Vorlesungsfolien ohne weitere Tips möglich sein.

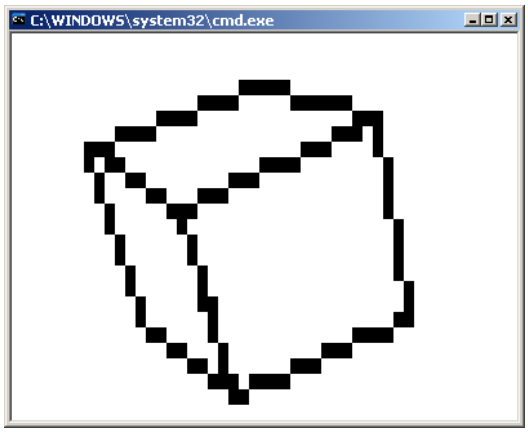

Abb. 4: `WireCull.exe`

Grafik-Hedonist/inn/en sollten die Gelegenheit nutzen, (freiwillig) im Editor mit den mitgelieferten CGF-Modellen (im Verzeichnis `_Dat`) zu experimentieren: Versuchen Sie, dem Würfel Spielwürfel-Augen zuzufügen, bauen Sie für das Haus-Modell eine Tür, Fenster, einen Vorgarten, einen Schornstein o.ä.!