



Bildgenerierung

Wintersemester 2023 / 2024

Übungsblatt 4

Aufgabe 10 (Perspektivische Projektion)

Programmieren Sie die perspektivische Projektion, die Überführung in normalisierte Koordinaten sowie die Umwandlung in Gerätekoordinaten. Ergänzen Sie hierzu im Rahmenprogramm proj1.cc im Verzeichnis /home/bildgen/Aufgaben/projektion-1 die entsprechenden Teile der Funktionen

```
Matrix4x4 berechneTransformation(const Vec3D& cop, const Vec3D& vrp,  
                                const Vec3D& vup, int w, int h,  
                                double& un, double& vn)
```

und

```
void maleLinien(Drawing& pic, const vector<Kante>& kanten,  
               const Matrix4x4& t, double un, double vn)
```

Alle für die Berechnung benötigten Operationen sind bereits vorgegeben, Sie können also z. B. die Matrix-Multiplikation direkt mittels `*`-Operator verwenden. Unterstützt werden ausschließlich 4×4 -Matrizen, geben Sie deshalb bei der Projektion und den nachfolgenden Transformationen für die n -Koordinate eine Nullzeile in der Matrix an.

Gehen Sie für die Berechnung der Transformation in `berechneTransformation()` in den folgenden Schritten vor und erzeugen Sie zunächst jeweils einzelne Matrizen mit dem gewünschten Effekt. `vpn`, `umin`, `umax`, `vmin` und `vmax` sind bereits vorgegeben.

1. Verschiebung des Aufpunktes der Projektionsebene (`vrp`) in den Ursprung.
2. Überführung in das (u, v, n) -Koordinatensystem.
 - a) Bestimmung des (u, v, n) -Koordinatensystems aus der Normalen der Projektionsebene, `vpn`, und der Aufwärtsrichtung, `vup`.
 - b) Rotation $z \mapsto n$, $x \mapsto u$, $y \mapsto v$.
3. Verschieben der transformierten Augenposition (`cop` überführt in das (u, v, n) -Koordinatensystem) in den Koordinatenursprung.
4. Standard perspektivische Projektion auf die (u, v) -Ebene.
5. Normalisierung der Koordinaten.

- a) Translation der unteren linken Ecke des Projektionsfensters in den Ursprung.
- b) Skalierung des Projektionsfensters, so dass es in das Einheitsquadrat passt.
- c) Speichern der Ausdehnung des Bildes im Einheitsquadrat, u_n und v_n (der entsprechende Code-Abschnitt ist vorgegeben).

Berechnen Sie nun die Gesamttransformation durch Matrix-Multiplikation in der richtigen Reihenfolge. Die Skalierung auf Gerätekoordinaten ist nicht Teil der Matrix, sie wird im Folgenden separat durchgeführt. Praktischer Grund hierfür ist das einfache 3D-Clipping in normalisierten Koordinaten.

Zum Zeichnen des Bildes ist dann noch Folgendes in `maleLinien()` zu tun:

1. Anwenden der Transformationsmatrix auf Anfangs- und Endpunkt der einzelnen Linien (der entsprechende Code-Abschnitt ist vorgegeben).
2. Umwandlung der homogenen Koordinaten in 2D-Koordinaten.
3. Skalierung auf Fenstergröße (Gerätekoordinaten) unter Verwendung von u_n und v_n .

Testen Sie Ihre Transformation mit den *.in-Dateien, z. B. „proj1 < Colosseum.in“.

Aufgabe 11 (*Strecken-Clipping nach Cohen und Sutherland*)

Sei das Kappungsfenster gegeben durch das achsenparallele Rechteck $[2; 8] \times [1; 5]$ sowie Linien mit Anfangspunkt P_1 und Endpunkt P_2 .

a) $P_1 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, P_2 = \begin{pmatrix} 10 \\ 4 \end{pmatrix}$ b) $P_1 = \begin{pmatrix} 3 \\ 3 \end{pmatrix}, P_2 = \begin{pmatrix} 6 \\ 0 \end{pmatrix}$

c) $P_1 = \begin{pmatrix} 7 \\ 0 \end{pmatrix}, P_2 = \begin{pmatrix} 10 \\ 2 \end{pmatrix}$ d) $P_1 = \begin{pmatrix} 5 \\ 6 \end{pmatrix}, P_2 = \begin{pmatrix} 11 \\ 4 \end{pmatrix}$

Bestimmen Sie mit dem Cohen-Sutherland-Verfahren jeweils Anfangs- und Endpunkt der gekappten Linie von Hand.

Aufgabe 12 (*Strecken-Clipping nach Cyrus, Beck, Liang und Barsky*)

Gegeben seien das gleiche achsenparallele Rechteck und die gleichen Punkte wie in Aufgabe 11. Bestimmen Sie mit dem Cyrus-Beck-Liang-Barsky-Verfahren jeweils Anfangs- und Endpunkt der gekappten Linie. Sie können die Lösung wahlweise von Hand berechnen oder ein kleines Hilfsprogramm schreiben.

Abgabe: Do., 19.11.2022, 16:15 Uhr

Senden Sie Ihre Lösungen der Theorie-Aufgaben und Ihre Programme per E-Mail an bildgen@studs.math.uni-wuppertal.de.