HOCHSCHULE KONSTANZ TECHNIK, WIRTSCHAFT UND GESTALTUNG
Fakultät für Informatik
Labor für Computergrafik
Prof. Dr. G. Umlauf

Konstanz, 11.09.2017

Assignment 3

„Computer graphics“

Deadline 19.12.2017, F033.

Preliminary remark: Do not use for this assignment OpenGL, GLUT or GLAUX functions for the projection or rotations! You can use the provided vector and matrix classes.

Exercise 5 (Central projection) — 2+1+1+2 points

Implement an application that computes the central projection along z-axis of a simple 3d scene of cuboids:

a. Implement the function

```
CVec4f projectZ(float fFocus, CVec4f pSicht)
```

for the central projection of an arbitrary 3d point `pSicht` in homogenous coordinates onto the projection plane. Use the setting as shown in Figure 1

- The eye point is on the positive z-axis, i.e. (`0,0,fFocus`).
- The view direction is anti-parallel to the die z-axis.
- The view-up-vector (y-axis of the image plane) is parallel to the y-axis.
- The image plane is the xy-plane.

The focal distance `fFocus` is a variable parameter of the function.

b. Implement the function

```
void drawProjektedZ (CVec3f Points[8]),
```

which takes eight 2d points and draws the wireframe of a projected 3d cuboid. To this end, simply connect corresponding points with edges using the Bresenham algorithm.

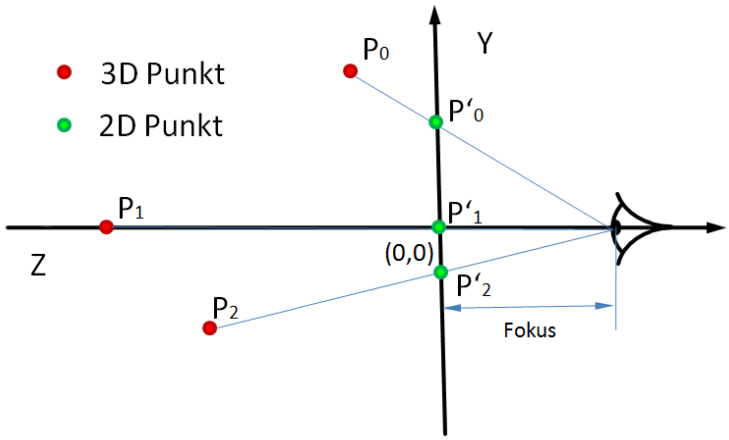

Figure 1 Central projection along the z-axis.

c. Define 3d-points for at least three 3d cuboid. Per cuboid only eight 3d points need to be stored. Hence, use an array or a structure or a simple class for the representation of cuboids.

d. Implement the function

```
void drawQuader(CVec3f Cuboid[8],float fFocus, Color c),
```

- which takes as parameter a cuboid,
- which projects the 3d-points using `projectZ(...)` onto the projection plane, and
- which draws the respective edges.

Implement these functions in the `display`-function, to display your scene of cuboids.

Exercise 6 (General view) 7+1 points

So far the scene can only be drawn from one perspective. In this part a general view transformation should be implemented. Thus, exercise 5 must be extended to use the following parameters:

- A general position of the eye points (in homogenous coordinates) `EyePoint`.
- A general view direction `ViewDir`.
- A general view-up-vector `ViewUp`.

The eye point and the two vectors `ViewDir` and `ViewUp` define a complete 3d coordinate system (view system). The missing x-axis is computed via $= y \times z$.

a. Implement the function

```
CMat4f getTransform( CVec4f viewOrigin,
                     CVec4f viewDir,
                     CVec4f viewUp),
```

which computes the complete transformation as 4×4-matrix for the coordinate from the view coordinate system tot world coordinate system, see Figure 2. The inverse of this matrix transforms points in world coordinates to points in view coordinates.

b. Implement the function

```
CVec4f projectZallg( CMat4f matTransf,
                     float  fFocus,
                     CVec4f pWorld),
```

which transforms he point `pWorld` with `matTransf` into view coordinates and projects it onto the image plane using `projectZ`.

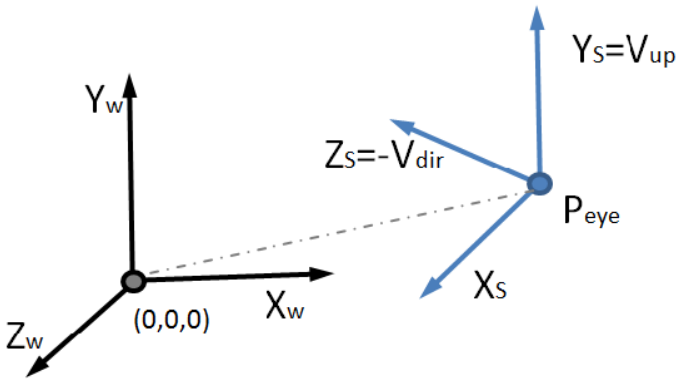

Figure 2 General view transformation.

HOCHSCHULE KONSTANZ TECHNIK, WIRTSCHAFT UND GESTALTUNG
Fakultät für Informatik
Labor für Computergrafik
Prof. Dr. G. Umlauf

Exercise 7 (Combination of 5&6) 1+3+3+2+1 points

Combine the functions from parts 5 and 6 into one application. Using the functions from part 6 the scene from part 5 can be visualized from an arbitrary perspective given the view coordinate system. The only function missing is an approach to manipulate the view coordinate system. Use here a simple keyboard-interaction realizing the following key assignments:

a. F increases the focal length and f decreases the focal length.

b. X, Y and Z rotate the view coordinate system in positive direction around x-, y- and z-axes of the world coordinate system and x, y and z rotate in negative direction around the respective axes.

c. A, B and C respectively a, b and c rotate the view coordinate system in respective direction around the respective axes of the view coordinate system (A, a: view direction, B, b: view-up-vector, . . .).

d. U, V, W, u, v and w translate the view coordinate system along the ayes of the world coordinate system in respective directions (U, u: x-axis, V, v: y- axis, W, w: z- axis).

e. R resets the view coordinate system to its initial position (congruent to the world coordinate system).

Initially the world and view coordinate systems should be congruent.

Deadline 19.12.2017, F033.