Interactive Computer Graphics
Coursework – Task 2

December 13, 2020

Task 2: Projections and Transformations

In Computer Graphics transformations and projections are defined through matrix operations as discussed during the lecture. In this exercise you will learn how to use these matrices. For this task you may want to use wireframe mode for better visibility.

A 3D point P is represented in homogeneous coordinates by a 4-dimensional vector

$$p = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{1}$$

A full 4×4 transformation matrix in homogeneous coordinates can be separated into individual parts steering trVianslation T, rotation R, and the affine parameters scaling A_{sc}, reflection A_{re}, and shearing A_{sh} ($A = A_{sc}A_{re}A_{sh}$). The full transformation can be defined as

$$p' = T \cdot R \cdot A \cdot p. \tag{2}$$

Translation T can be defined as

$$T = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{3}$$

Rotation R can be defined as

$$R = R_x \cdot R_y \cdot R_z = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{4}$$

Scaling A_{sc} can be defined as

$$A_{sc} = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{5}$$

where s_x, s_y, s_z are real values defining a scale factor along each axis.

Reflection through a specific plane A_{re} can be achieved by inverting components of the diagonal, *e.g.*, a reflection through the xy plane would look like this:

$$A_{re} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{6}$$

Shear effects can be achieved through manipulating the rotation parameters in a non-orthogonal way:

$$A_{sh} = \begin{pmatrix} 1 & a & b & 0 \\ c & 1 & d & 0 \\ e & f & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{7}$$

a, b, c, d, e, f changes each coordinate as a linear combination of all three.

A combined transformation matrix can be used as *ModelMatrix* to manipulate a 3D object in 3D space or to define the position of the camera plane as *ViewMatrix*.

The projection on the camera plane is defined through the *ProjectionMatrix*. In case of orthographic projection this matrix simply removed the z-coordinate and looks like

$$A_{re} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{8}$$

For perspective transformation we can add the focal length of the camera and use

$$A_{re} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/f & 0 \end{pmatrix} \tag{9}$$

as *ProjectionMatrix*. Note that the framework will provide the matrices as used by OpenGL and pre-multiply internal scaling (for a nice viewing frustum of the teapot) and enforces a viewing direction along the negative z-axis by putting negative values at locations (4,3) and (3,3).

In `http://shaderlabweb.doc.ic.ac.uk/` you can framework provides and interface to all of these matrices. The matrices can be found in the `Uniforms` tab. `mMatrix` is the *model matrix* and does not need to be attached to anything since we are in full control of this in our case. The default model matrix centers the teapot at the origin. `vMatrix` is the *view matrix* and is attached to the according matrix in the host program. `pMatrix` is the *projection matrix* and attached to the view's projection matrix.

Your task is to

- rotate the object 45° around axis $(0.5, 0.5, 0.75)$.
- scale the object by 50%.
- translate the object to $(0, 5, 0)$ followed by a 30° rotation around $(0, 0, 1)$ (z-axis).
- reflect the object through a plane defined by its normal vector $(0.7071, 0.7071, 0)$.
- shear the object along the x-axis to a general parallelepiped so that the top left edge of the cube is translated to $(1, 0, 0)$.
- change to orthographic projection.
- use perspective projection with focal length $f = 20mm$. The height and width of your field of view are shown on the perspective matrix widget.

Generate a separate screen shot for each of these tasks. Reset your matrices to *default* between the tasks.

HAVE A LOT OF FUN!!