Department of Mathematics and Computer Science
University of Southern Denmark
RF
August 14, 2018

Change of Basis

Let E and F be two coordinate systems for $\mathbb{R}^3$ with same origo, having the orthonormal bases $\vec{e}_1, \vec{e}_2, \vec{e}_3$ and $\vec{f}_1, \vec{f}_2, \vec{f}_3$, respectively. Orthonormal means unit length and pairwise orthogonal, i.e.,

$$\vec{e}_i \cdot \vec{e}_j = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \quad \text{and} \quad \vec{f}_i \cdot \vec{f}_j = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

Let the coordinates in E of the $\vec{f}_i$'s be known:

$$\begin{aligned}
\vec{f}_1 &= f_{11} \cdot \vec{e}_1 + f_{12} \cdot \vec{e}_2 + f_{13} \cdot \vec{e}_3 \\
\vec{f}_2 &= f_{21} \cdot \vec{e}_1 + f_{22} \cdot \vec{e}_2 + f_{23} \cdot \vec{e}_3 \\
\vec{f}_3 &= f_{31} \cdot \vec{e}_1 + f_{32} \cdot \vec{e}_2 + f_{33} \cdot \vec{e}_3
\end{aligned}$$

Given a point $\vec{x} \in \mathbb{R}^3$, it has a set of coordinates in each system. Let these be $\vec{v} = (v_1, v_2, v_3)$ and $\vec{w} = (w_1, w_2, w_3)$, respectively. Then

$$\begin{aligned}
\vec{x} = {} & w_1 \cdot \vec{f}_1 + w_2 \cdot \vec{f}_2 + w_3 \cdot \vec{f}_3 \\
& = w_1 \cdot (f_{11} \cdot \vec{e}_1 + f_{12} \cdot \vec{e}_2 + f_{13} \cdot \vec{e}_3) \\
& + w_2 \cdot (f_{21} \cdot \vec{e}_1 + f_{22} \cdot \vec{e}_2 + f_{23} \cdot \vec{e}_3) \\
& + w_3 \cdot (f_{31} \cdot \vec{e}_1 + f_{32} \cdot \vec{e}_2 + f_{33} \cdot \vec{e}_3) \\
& = \vec{e}_1 \cdot (w_1 f_{11} + w_2 f_{21} + w_3 f_{31}) \\
& + \vec{e}_2 \cdot (w_1 f_{12} + w_2 f_{22} + w_3 f_{32}) \\
& + \vec{e}_3 \cdot (w_1 f_{13} + w_2 f_{23} + w_3 f_{33}).
\end{aligned}$$

Thus, the values in parentheses after the last equality are the coordinates of $\vec{x}$ in E, that is, $\vec{v} = (v_1, v_2, v_3)$.

So for

$$A = \begin{bmatrix} f_{11} & f_{21} & f_{31} \\ f_{12} & f_{22} & f_{32} \\ f_{13} & f_{23} & f_{33} \end{bmatrix} = \begin{bmatrix} \vec{f_1} & \vec{f_2} & \vec{f}3 \end{bmatrix}$$

we have

$$\vec{v} = A \cdot \vec{w}.$$

The equation tells us how to get the coordinates of $\vec{x}$ in E if we know its coordinates in F.

Since the $\vec{f_i}$'s are orthonormal, we have

$$A^T \cdot A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = I.$$

Hence,

$$A^T \cdot \vec{v} = A^T \cdot (A \cdot \vec{w}) = (A^T \cdot A) \cdot \vec{w}) = I \cdot \vec{w} = \vec{w}.$$

In other words,

$$\vec{w} = A^T \cdot \vec{v}.$$

The equation tells us how to get the coordinates of $\vec{x}$ in F if we know its coordinates in E.

Application 1

Assume we want the camera to be positioned in $\vec{c} = (c_1, c_2, c_3)$ looking towards at target position $\vec{t} = (t_1, t_2, t_3)$ and with an additional "up" direction

$\vec{u} = (u_1, u_2, u_3)$ specified to fix the orientation of the camera. We require that $\vec{u}$ is not parallel to the line of sight vector $\vec{l} = \vec{t} - \vec{c}$.

Decompose $\vec{u}$ into two components $\vec{u} = \vec{u}_1 + \vec{u}_2$ with $\vec{u}_1$ parallel and $\vec{u}_2$ orthogonal to $\vec{l}$:

$$
\begin{aligned}
\vec{u}_1 &= \frac{\vec{u} \cdot \vec{l}}{||\vec{l}||} \cdot \frac{\vec{l}}{||\vec{l}||} = \frac{\vec{u} \cdot \vec{l}}{||\vec{l}||^2} \cdot \vec{l} = \frac{\vec{u} \cdot \vec{l}}{\vec{l} \cdot \vec{l}} \cdot \vec{l} \\
\vec{u}_2 &= \vec{u} - \vec{u}_1
\end{aligned}
$$

The camera in OpenGL is positioned in $(0, 0, 0)$ looking down the negative z-axis and with the y-axis as the “up” direction. Hence, we consider the following three vectors:

$$
\begin{aligned}
\vec{f}_3 &= \frac{-\vec{l}}{||\vec{l}||} \\
\vec{f}_2 &= \frac{\vec{u}_2}{||\vec{u}_2||} \\
\vec{f}_1 &= \vec{f}_2 \times \vec{f}_3
\end{aligned}
$$

These vectors are orthonormal by construction and form a (right-handed) coordinate system F. In this coordinate system, the camera is positioned as required by OpenGL. Thus, we want to express all points of the world in this coordinate system.

1. First translate the world by the vector $-\vec{c}$. This makes the world be expressed in a coordinate system which has the same orientation as the original world system E (same base vectors), but with an origo which is the same as the system F.

2. Then change points of the world to be expressed in the coordinate system just constructed (using the method earlier in this note). In this coordinate system, the camera sits as required by OpenGL.

The two steps above are performed on a point $\vec{v}$ by the calculation

$$A^T \cdot (T_{-\vec{c}} \cdot \vec{v}) = (A^T \cdot T_{-\vec{c}}) \cdot \vec{v},$$

where A is the matrix from the method above (based on the vectors $\vec{f}_1$, $\vec{f}_2$, and $\vec{f}_3$) and $T_{-\vec{c}}$ is a translation matrix with displacement vector $-\vec{c}$. [As translation matrices are 4×4 matrices, the 3×3 matrix A should be embedded in a 4×4 matrix in the usual way.]

This matrix $A^T \cdot T_{-\vec{c}}$ is the same construction as the look-at matrix from Section 3.9 in the book.

Application 2

If we want to rotate an angle α around an arbitrary axis $\vec{a}$ (through origo), we can create an coordinate system F $(= \vec{f}_1, \vec{f}_2, \vec{f}_3)$ whose z-axis $\vec{f}_3$ is parallel to $\vec{a}$. This can be done as above, with $-\vec{a}$ in $\vec{l}$'s place, and any vector not parallel to $\vec{a}$ in $\vec{u}$'s place (for instance one of the coordinate system vectors must fulfill this, as at most one of them can be parallel to $\vec{a}$). We then express points $\vec{v}$ in that system, rotate an angle α around its z-axis and afterwards express the rotated points in the original system. This is the calculation

$$A \cdot (R_{z,\alpha} \cdot (A^T \cdot \vec{v})) = (A \cdot R_{z,\alpha} \cdot A^T) \cdot \vec{v},$$

where $R_z(\alpha)$ is a matrix rotating an angle α around the z-axis (which we know already how to make). Hence, such a general rotation matrix can be found as

$$A \cdot R_{z,\alpha} \cdot A^T.$$