Geometric Transformations

Transformations like translation, rotation, scaling, mirroring etc. of objects within a given coordinate system or between coordinate systems are called geometric transformations. In the following, we will describe the necessary transformation rules for points only. From this, all other objects can then be transformed easily by transforming all vertices of these objects.

Simple 2D Transformations

Translation

The translation of a point (x,y) by some vector (t_x,t_y) results in the transformed point

$$(x',y') = (x+t_x,y+t_y) \; .$$

Rotation

When rotating an object around the point of origin by an angle θ, the point (x,y) ends up at

$$(x',y') = (x\cdot\cos\theta - y\cdot\sin\theta,\; x\cdot\sin\theta + y\cdot\cos\theta).$$

Scaling (magnification or miniaturization)

When scaling an object from the point of origin by the factor s, the point (x,y) is mapped to

$$(x',y') = (s\cdot x,\; s\cdot y).$$

If we use different scaling factors s_x and s_y in x- respectively y-direction, we get

$$(x',y') = (s_x\cdot x,\; s_y\cdot y) \; .$$

Reflection (Mirroring)

Reflecting a point about a coordinate axis is a special case of scaling with $s_x = -1$ oder $s_y = -1$.

All other transformations can be obtained by consecutive application of the base transformations described above. All these transformations (except the translation) can also be described by ***transformation matrices***. When using transformation matrices we have to describe our points as vectors in order to apply the matrix operations:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} \qquad \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} s_x & 0 \\ 0 & s_y \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} \qquad \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} \qquad (x'\; y') = (x+dx,\; y+dy) \;\; \textbf{... ?}$$

rotation (counter-clockwise) scaling reflection about x-axis translation

Homogeneous Coordinates

In order to be able to describe translations in matrix notation too, we use *homogeneous coordinates*. To each point we assign an additional coordinate h, where the conversion to 2D coordinates is done by dividing the x- and y-coordinates by h. Thus h=1 is mostly used. For a point (x,y) we now have (x,y,1), and the transformation matrices are extended by an additional row and column with identity values:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \qquad \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \qquad \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

2D rotation 2D scaling 2D translation

What is the advantage of being able to formulate all transformations in this unified matrix notation? In most cases, larger objects (models, images) containing a lot of points are transformed as a whole, which means that the same sequence of transformations is being applied to each point of those objects. This corresponds to a sequential multiplication of a point P with the matrices M_1, M_2, M_3…: $P' = M_1 \cdot P$, $P'' = M_2 \cdot P'$, $P''' = M_3 \cdot P''$, and so on. Here we can take advantage of the associativity of matrix multiplications [i.e. $(M_1 \cdot M_2) \cdot M_3 = M_1 \cdot (M_2 \cdot M_3)$] to reduce the computing time massively.

Instead of $\mathbf{P^{(n)} = M_n \cdot (M_{n-1} \cdot \ldots (M_3 \cdot (M_2 \cdot (M_1 \cdot P))))}$ **we write** $\mathbf{P^{(n)} = (M_n \cdot M_{n-1} \cdot \ldots \cdot M_3 \cdot M_2 \cdot M_1) \cdot P}$ **.**

Now we can *precalculate* the combined product $M = (M_n \cdot M_{n-1} \cdot \ldots \cdot M_3 \cdot M_2 \cdot M_1)$ and then apply this *one* single combined matrix to all points.

For a better illustration we will label the basic transformation matrices as follows:

$\mathbf{T(t_x,t_y)}$ = translation by the vector (t_x,t_y)
$\mathbf{R(\theta)}$ = rotation about the angle θ
$\mathbf{S(s_x,s_y)}$ = scaling with the factors s_x and s_y.

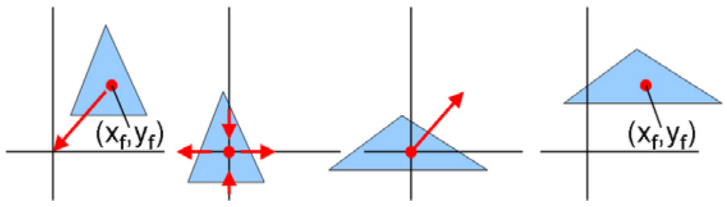
The inverse transformations of these basic transformations are:

$$T^{-1}(t_x,t_y) = T(-t_x,-t_y) \qquad R^{-1}(\theta) = R(-\theta) \qquad S^{-1}(s_x, s_y) = S(1/s_x, 1/s_y)$$

From these basic transformations we can now build more complex transformations.
As an example we examine the

Scaling with respect to a point other than the origin:

1st step = translation of the scaling center into the point of origin: $T(-x_f,-y_f)$
2nd step = scaling of the object with respect to the point of origin: $S(s_x,s_y)$
3rd step = translation of the object back to its original location: $T^{-1}(-x_f,-y_f) = T(x_f,y_f)$



So we obtain the generalized scaling matrix with (x_f,y_f) as scaling center by:

$$S(x_f,y_f,s_x,s_y) = T(x_f,y_f) \cdot S(s_x,s_y) \cdot T(-x_f,-y_f)$$

Our next example is the

***Reflection about an arbitrary axis y = mx+b*:**

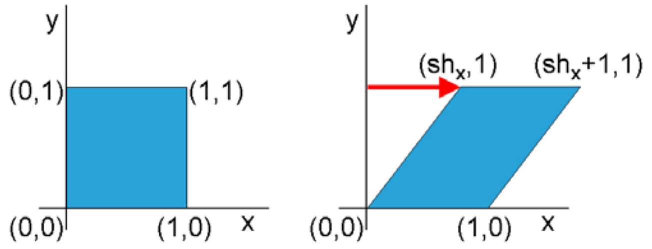
1st step = translation to move the reflection axis through the point of origin: $T(0,-b)$
2nd step = rotation to align the reflection axis with e.g. the x-axis: $R(-\theta)$ $[m = \tan\theta]$
3rd step = reflection about the x-axis: $S(1,-1)$
4th step = rotation back to the original angle: $R^{-1}(-\theta) = R(\theta)$
5th step = translation back to move the reflection axis to its original location: $T^{-1}(0,-b) = T(0,b)$

So the generalized matrix for reflection about the axis y = mx+b can be obtained by:

$$X(m,b) = T(0,b) \cdot R(\theta) \cdot S(1,-1) \cdot R(-\theta) \cdot T(0,-b)$$

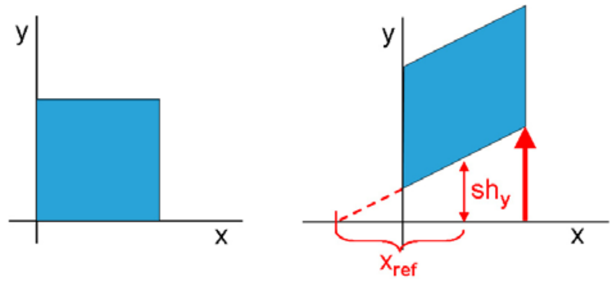
Another important transformation is *shearing*. The simplest case, i.e. shearing in x-direction with fixed x-axis has the form:

$$\begin{pmatrix} 1 & sh_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$



More generally, shearing can also be performed along any line parallel to a coordinate axis, which would e.g. in y-direction look like this:

$$\begin{pmatrix} 1 & 0 & 0 \\ sh_y & 1 & -sh_y \cdot x_{ref} \\ 0 & 0 & 1 \end{pmatrix}$$



Of course, also here we can deduce the general matrix for a shearing transformation not parallel to any coordinate axis: First rotation to axis-parallel orientation, then shearing, then rotation back to original orientation.

Also the general window-viewport transformation from chapter “2D-Viewing” can easily be described by transformation matrices. The used operations are: translation of one coordinate origin into another, rotation of the viewport into the axis-directions of the window and scaling along the axes. But in contrast to the combined transformations examined above, the back-translation and back-rotation are omitted here.

Affine Transformations
All transformations mentioned so far are *affine transformations*, which means that the coordinates can be converted into each other by linear functions plus a translation term. Affine mappings conserve co-linearity, i.e. 3 points lying on a common line before the transformation also lie on a common line afterwards, and proportionality of distances along a straight line, i.e. relations of distances on a line are preserved. Furthermore, parallel lines are always mapped to parallel lines, and finite points stay finite. All affine transformations (including the shearing!) can be obtained from a combination of the basic translation-, rotation- and scale-transformations. Furthermore, affine transformations which only contain rotation, translation and reflection are distance- and angle-preserving.

3D Transformations

All concepts from 2D can easily be extended to 3D. Again a homogeneous component is required in order to gain 4x4 matrices that can be applied to 4-dimensional vectors. Later we will see that projections can be described by 4x4 matrices too.

In the following we list the most important 3D transformations:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

3D translation

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

3D scaling

$$\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

reflection about the yz- xz- xy-plane

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

3D rotation about x-axis

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

3D rotation about y-axis

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

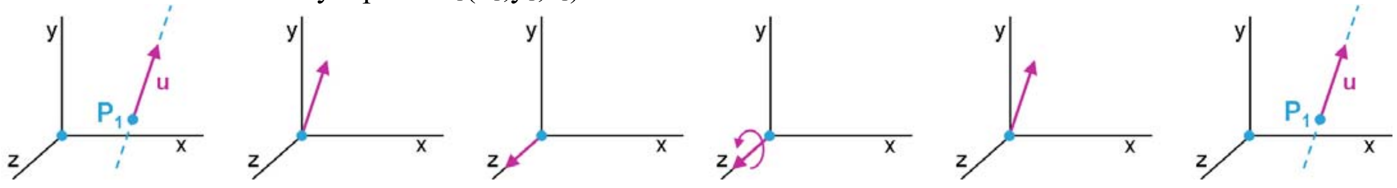
3D rotation about z-axis

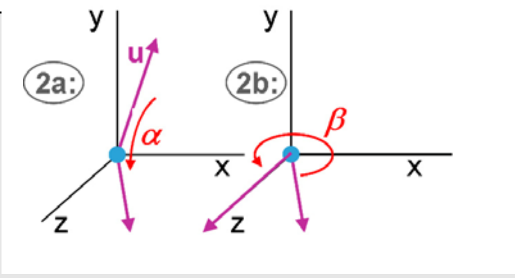
We label these basic 3D transformation matrices as follows:

$T(t_x,t_y,t_z)$ = translation by the vector (t_x,t_y,t_z)
$R_x(\theta)$ = rotation through the angle θ about the x-axis (y- and z-axis analogously)
$S(s_x,s_y,s_z)$ = scaling by the factors s_x, s_y and s_z.

As an example for a more complex transformation we will deduce a

Rotation through an angle θ about an arbitrary axis in 3D space
Let the axis be defined by a point $P_1(x_1,y_1,z_1)$ and a direction vector u.





1st step = translate the point P_1 into the point of origin:
$T(-x_1,-y_1,-z_1)$
2nd step = rotate vector u into the z-axis
2a. rotate vector u about the x-axis into the xz-plane: $R_x(\alpha)$
Let u = (a,b,c), then u´=(0,b,c) is the projection of u onto the yz-plane. The rotation angle α about the x-axis can be obtained from $\cos \alpha = c/d$ with $d = \sqrt{(b^2+c^2)}$
2b. rotate vector u about the y-axis into the z-axis: $R_y(\beta)$
The rotation angle β about the y-axis is calculated from $\cos \beta = d$ (respectively $\sin \beta = -a$)
3rd step = perform rotation through θ about the z-axis: $R_z(\theta)$
4th step = rotate vector u back into its original orientation: first $R_y(-\beta)$, then $R_x(-\alpha)$
5th step = translate point P_1 back to its original position: $T(x_1,y_1,z_1)$

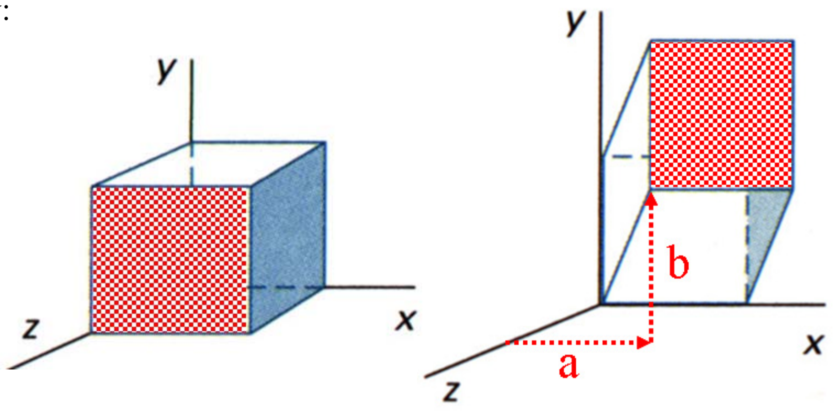
So the resulting matrix can be calculated as follows:

$$R(\theta) = T^{-1}(-x_1,-y_1,-z_1)\cdot R_x^{-1}(\alpha)\cdot R_y^{-1}(\beta)\cdot R_z(\theta)\cdot R_y(\beta)\cdot R_x(\alpha)\cdot T(-x_1,-y_1,-z_1) =$$
$$= T(x_1,y_1,z_1)\cdot R_x(-\alpha)\cdot R_y(-\beta)\cdot R_z(\theta)\cdot R_y(\beta)\cdot R_x(\alpha)\cdot T(-x_1,-y_1,-z_1)$$

A ***shearing in 3D*** can also be depicted easily:

A shearing parallel to the xy-plane with the parameters a in x-direction and b in y-direction is calculated by

$$\begin{bmatrix} 1 & 0 & a & 0 \\ 0 & 1 & b & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



A shearing along a fixed plane other than one of the principal planes of the coordinate system can also be deduced easily.