APPENDIX C

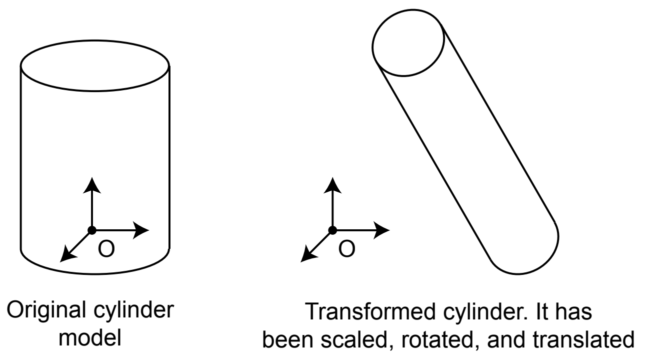
Affine Transformations

CONTENTS

C.1 THE NEED FOR GEOMETRIC TRANSFORMATIONS

One could imagine a computer graphics system that requires the user to construct everything directly into a single scene. But, one can also immediately see that this would be an extremely limiting approach. In the real world, things come from various places and are arranged together to create a scene. Further, many of these things are themselves collections of smaller parts that are assembled together. We may wish to define one object relative to another – for example we may want to place a hand at the end of an arm. Also, it is often the case that parts of an object are similar, like the tires on a car. And, even things that are built on scene, like a house for example, are designed elsewhere, at a scale that is usually many times smaller than the house as it is built. Even more to the point, we will often want to animate the objects in a scene, requiring the ability to move them around relative to each other. For animation we will want to be able to move not only the objects, but also the camera, as we render a sequence of images as time advances to create an illusion of motion. We need good mechanisms within a computer graphics system to provide the flexibility implied by all of the issues raised above.

The figure below shows an example of what we mean. On the left, a cylinder has been built in a convenient place, and to a convenient size. Because of the requirements of a scene, it is first scaled to be longer and thinner than its original design, rotated to a desired orientation in space, and then moved to a desired position (i.e. translated). The set of operations providing for all such transformations, are known as the *affine transforms*. The affines include translations and all linear transformations, like scale, rotate, and shear.


O
Original cylinder model
O
Transformed cylinder. It has been scaled, rotated, and translated

C.2 AFFINE TRANSFORMATIONS

Let us first examine the affine transforms in 2D space, where it is easy to illustrate them with diagrams, then later we will look at the affines in 3D.

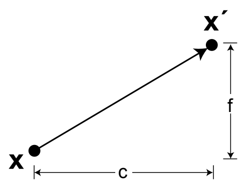
Consider a point $\mathbf{x} = (x, y)$. Affine transformations of $\mathbf{x}$ are all transforms that can be written

$$\mathbf{x}' = \begin{bmatrix} ax + by + c \\ dx + ey + f \end{bmatrix},$$

where a through f are scalars.

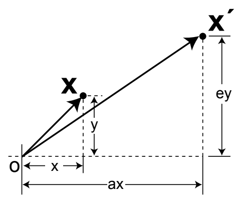
For example, if $a, e = 1$, and $b, d = 0$, then we have a pure translation

$$\mathbf{x}' = \begin{bmatrix} x + c \\ y + f \end{bmatrix}.$$



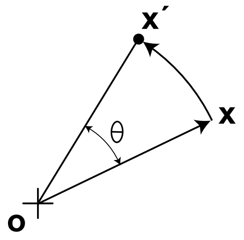
If $b, d = 0$ and $c, f = 0$ then we have a pure scale.

$$\mathbf{x}' = \begin{bmatrix} ax \\ ey \end{bmatrix}$$



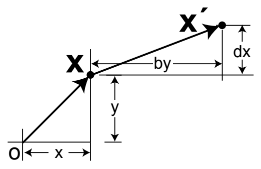
And, if $a, e = \cos\theta$, $b = -\sin\theta$, $d = \sin\theta$, and $c, f = 0$, then we have a pure rotation about the origin

$$\mathbf{x}' = \begin{bmatrix} x\cos\theta - y\sin\theta \\ x\sin\theta + y\cos\theta \end{bmatrix}.$$



Finally if $a, e = 1$, and $c, f = 0$ we have the shear transforms

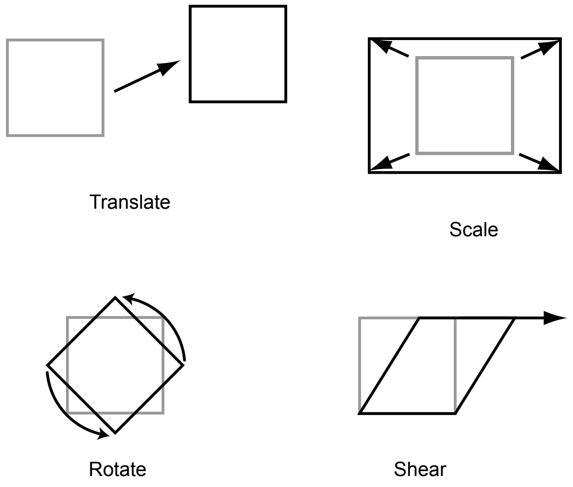
$$\mathbf{x}' = \begin{bmatrix} x + by \\ y + dx \end{bmatrix}.$$



In summary, we have the four basic affine transformations shown in the figure below:

- Translate moves a set of points a fixed distance in x and y,
- Scale scales a set of points up or down in the x and y directions,
- Rotate rotates a set of points about the origin,
- Shear offsets a set of points a distance proportional to their x and y coordinates.

Note that only shear and scale change the shape determined by a set of points.



C.3 MATRIX REPRESENTATION OF THE LINEAR TRANSFORMATIONS

The affine transforms scale, rotate and shear are actually linear transforms and can be represented by a matrix multiplication of a point represented as a vector,

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} ax + by \\ dx + ey \end{bmatrix} = \begin{bmatrix} a & b \\ d & e \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix},$$

or $\mathbf{x}' = M\mathbf{x}$, where M is the matrix.

One very nice feature of the matrix representation is that we can use it to factor a complex transform into a set of simpler transforms. For example, suppose we want to scale an object up to a new size, shear the object to a new shape, and finally rotate the object. Let S be the scale matrix, H be the shear matrix and R be the rotation matrix. Then

$$\mathbf{x}' = R(H(S\mathbf{x}))$$

defines a sequence of three transforms: 1st-scale, 2nd-shear, 3rd-rotate. Because matrix multiplication is associative, we can remove the parentheses and multiply the three matrices together, giving a new matrix $M = RHS$. Now we can rewrite our transform

$$\mathbf{x}' = (RHS)\mathbf{x} = M\mathbf{x}$$

If we have to transform thousands of points on a complex model, it is clearly easier to do one matrix multiplication, rather than three, each time we want to transform a point. Thus, matrices are a very powerful way to encapsulate a complex transform and to store it in a compact and convenient form.

In matrix form, we can catalog the linear transforms as

$$\text{Scale:} \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}, \ \text{Rotate:} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \ \text{Shear:} \begin{bmatrix} 1 & h_x \\ h_y & 1 \end{bmatrix},$$

where s_x and s_y scale the x and y coordinates of a point, θ is an angle of counterclockwise rotation around the origin, h_x is a horizontal shear factor, and h_y is a vertical shear factor.

C.4 HOMOGENEOUS COORDINATES

Since the matrix form is so handy for building up complex transforms from simpler ones, it would be very useful to be able to represent all of the affine transforms by matrices. The problem is that translation is not a linear transform. The way out of this dilemma is to turn the 2D problem into a 3D problem, but in *homogeneous coordinates*.

We first take all of our points $\mathbf{x} = (x, y)$, express them as 2D vectors $\begin{bmatrix} x \\ y \end{bmatrix}$ and make these into 3D vectors with identical (thus the term homogeneous) 3rd coordinates set to 1:

$$\begin{bmatrix} x \\ y \end{bmatrix} \Longrightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}.$$

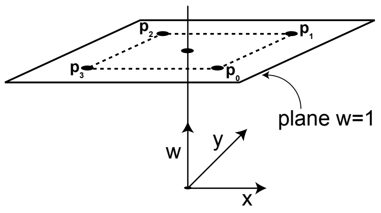
By convention, we call this third coordinate the w coordinate, to distinguish it from the usual 3D z coordinate. We also extend our 2D matrices to 3D homogeneous form by appending an extra row and column, giving

$$\text{Scale:} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}, \text{Rotate:} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}, \text{Shear:} \begin{bmatrix} 1 & h_x & 0 \\ h_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Note what happens when we multiply our 3D homogeneous matrices by 3D homogeneous vectors:

$$\begin{bmatrix} a & b & 0 \\ d & e & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} ax + by \\ dx + ey \\ 1 \end{bmatrix}.$$

This is the same result as in 2D, with the exception of the extra w coordinate, which remains 1. All we have really done is to place all of our 2D points on the plane $w = 1$ in 3D space, and now we do all the operations on this plane. Really, the operations are still 2D operations.



But, the magic happens when we place the translation parameters c and f in the matrix in the 3rd column:

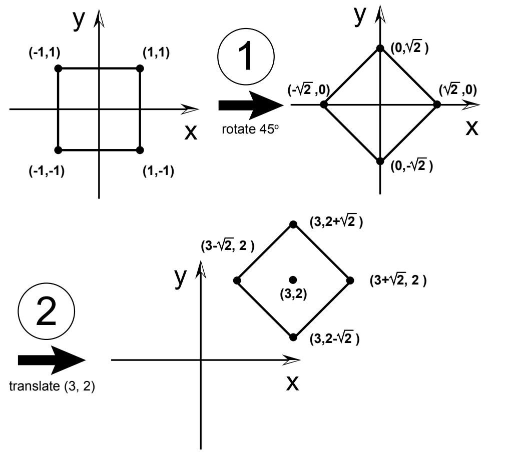
$$\begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} ax + by + c \\ dx + ey + f \\ 1 \end{bmatrix}$$

We can now do translations as linear operations in homogeneous coordinates! So, we can add a final matrix to our catalog:

$$\text{Translate:} \begin{bmatrix} 1 & 0 & \triangle x \\ 0 & 1 & \triangle y \\ 0 & 0 & 1 \end{bmatrix},$$

where $\triangle x$ is the translation in the x direction and $\triangle y$ is the translation in the y direction. The astute reader will see the trick behind the magic – 2D translation is now being expressed as a shear in 3D space.

Now, suppose we have a 2×2 square centered at the origin and we want to first rotate the square by 45° about its center and then move the square so its center is at $(3, 2)$. We can do this in two steps, as shown in the diagram to the right.



In matrix form:

$$M = T_{(3,2)}R_{45^\circ} = \begin{bmatrix} 1 & 0 & 3 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 45^\circ & -\sin 45^\circ & 0 \\ \sin 45^\circ & \cos 45^\circ & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos 45^\circ & -\sin 45^\circ & 3 \\ \sin 45^\circ & \cos 45^\circ & 2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \sqrt{2}/2 & -\sqrt{2}/2 & 3 \\ \sqrt{2}/2 & \sqrt{2}/2 & 2 \\ 0 & 0 & 1 \end{bmatrix}.$$

Note that

$$M \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 2 + \sqrt{2} \\ 1 \end{bmatrix}, \text{and } M \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 - \sqrt{2} \\ 2 \\ 1 \end{bmatrix},$$

verifying that we get the same result shown in the figure.

C.5 3D FORM OF THE AFFINE TRANSFORMATIONS

Now, we can extend all of these ideas to 3D in the following way:

1. Convert all 3D points to homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \Longrightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}.$$

The extra (4th) coordinate is again called the w coordinate.

2. Use matrices to represent the 3D affine transforms in homogeneous form.

The following matrices constitute the basic affine transforms in 3D, expressed in homogeneous form:

$$\text{Translate:}\begin{bmatrix}1 & 0 & 0 & \triangle x\\0 & 1 & 0 & \triangle y\\0 & 0 & 1 & \triangle z\\0 & 0 & 0 & 1\end{bmatrix},\ \text{Scale:}\begin{bmatrix}s_x & 0 & 0 & 0\\0 & s_y & 0 & 0\\0 & 0 & s_z & 0\\0 & 0 & 0 & 1\end{bmatrix},$$

and

$$\text{Shear:}\begin{bmatrix}1 & h_{xy} & h_{xz} & 0\\h_{yx} & 1 & h_{yz} & 0\\h_{zx} & h_{zy} & 1 & 0\\0 & 0 & 0 & 1\end{bmatrix}.$$

In addition, there are three basic rotations in 3D,

$$\text{Rotation about the } x \text{ axis:}\begin{bmatrix}1 & 0 & 0 & 0\\0 & \cos\theta_x & -\sin\theta_x & 0\\0 & \sin\theta_x & \cos\theta_x & 0\\0 & 0 & 0 & 1\end{bmatrix},$$

$$\text{Rotation about the } y \text{ axis:}\begin{bmatrix}\cos\theta_y & 0 & \sin\theta_y & 0\\0 & 1 & 0 & 0\\-\sin\theta_y & 0 & \cos\theta_y & 0\\0 & 0 & 0 & 1\end{bmatrix},$$

and

$$\text{Rotation about the } z \text{ axis:}\begin{bmatrix}\cos\theta_z & -\sin\theta_z & 0 & 0\\\sin\theta_z & \cos\theta_z & 0 & 0\\0 & 0 & 1 & 0\\0 & 0 & 0 & 1\end{bmatrix}.$$

The rotations, specified in this way, determine an amount of rotation about each of the individual axes of the coordinate system. The angles θ_x, θ_y, and θ_z of rotation about the three axes are called the Euler angles. They can be used to describe an off-axis rotation, by combining Euler angle rotations via matrix multiplication. Note, however, that the order of rotation affects the end result, so besides specifying Euler angles, an order of rotation must be specified. In general, affine transformations are associative but are not commutative, so the order in which operations are done is highly important. One can see this for rotations by computing the product $R_{\theta_x}R_{\theta_y}R_{\theta_z}$, and comparing with the result obtained by the product $R_{\theta_z}R_{\theta_y}R_{\theta_x}$. Please see Appendix D for a more powerful and general look at rotation.