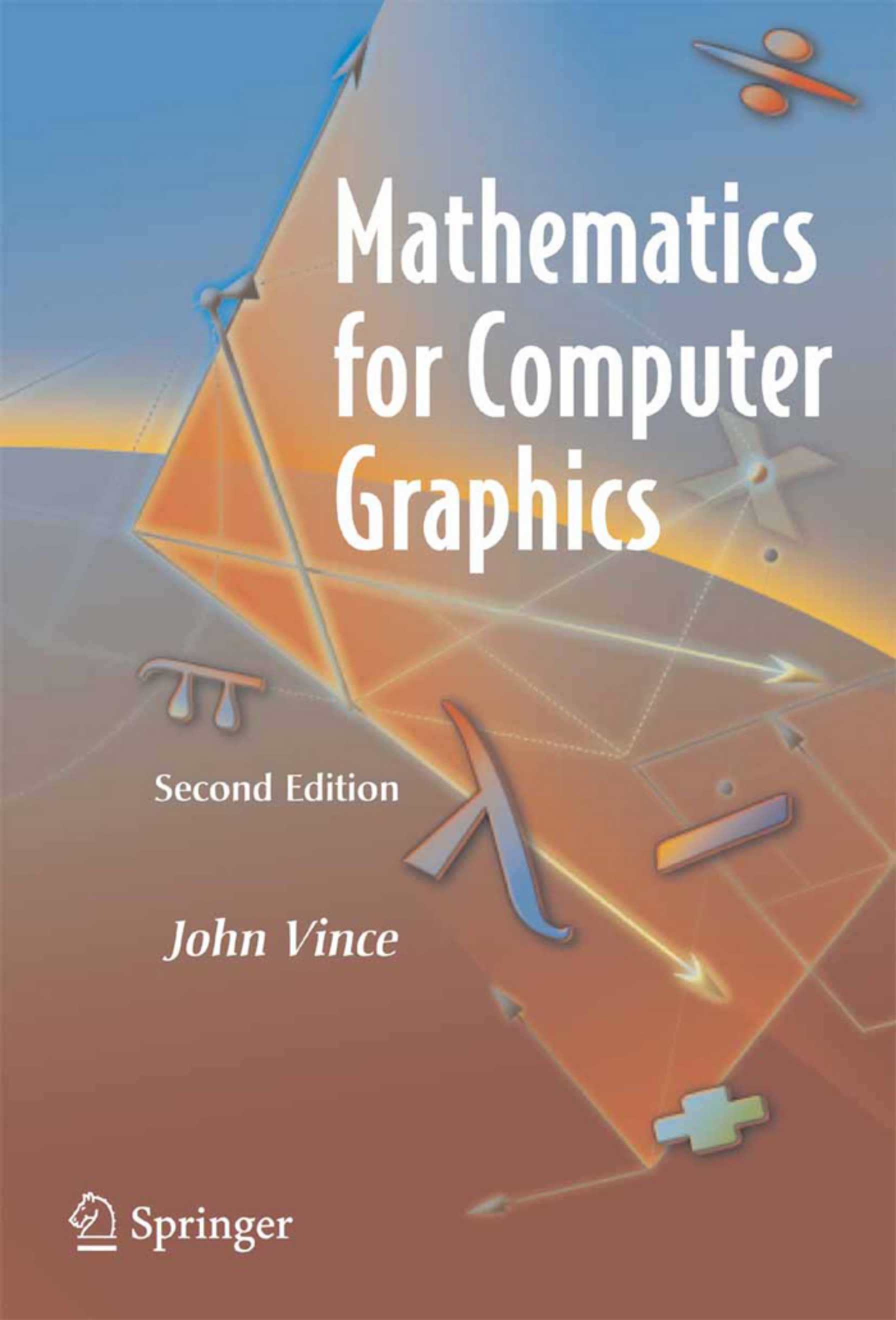
Mathematics for Computer Graphics

Second Edition

John Vince

Springer



Contents

7 Transformation

Transformations are used to scale, translate, rotate, reflect and shear shapes and objects. And, as we shall discover shortly, it is possible to effect this by changing their coordinate values.

Although algebra is the basic notation for transformations, it is also possible to express them as *matrices*, which provide certain advantages for viewing the transformation and for interfacing to various types of computer graphics hardware. We begin with an algebraic approach and then introduce matrix notation.

7.1 2D Transformations

7.1.1 Translation

Cartesian coordinates provide a one-to-one relationship between number and shape, such that when we change a shape's coordinates, we change its geometry. For example, if $P(x, y)$ is a vertex on a shape, when we apply the operation $x' = x + 3$ we create a new point $P'(x', y)$ three units to the right. Similarly, the operation $y' = y + 1$ creates a new point $P'(x, y')$ displaced one unit vertically. By applying both of these transforms to every vertex to the original shape, the shape is displaced as shown in Figure 7.1.

7.1.2 Scaling

Shape scaling is achieved by multiplying coordinates as follows:

$$\begin{aligned} x' &= 2x \\ y' &= 1.5y \end{aligned} \tag{7.1}$$

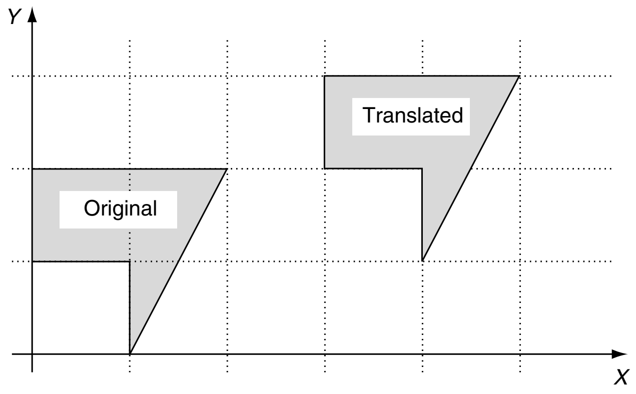


Fig. 7.1. The translated shape results by adding 3 to every x-coordinate, and 1 to every y-coordinate of the original shape.

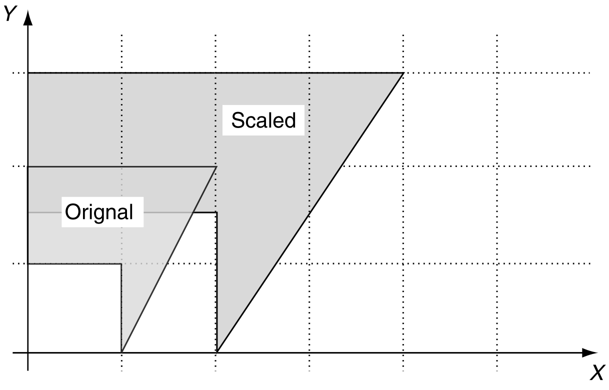


Fig. 7.2. The scaled shape results by multiplying every x-coordinate by 2 and every y-coordinate by 1.5.

This transform results in a horizontal scaling of 2 and a vertical scaling of 1.5, as illustrated in Figure 7.2. Note that a point located at the origin does not change its place, so scaling is relative to the origin.

7.1.3 Reflection

To make a reflection of a shape relative to the y-axis, we simply reverse the sign of the x-coordinate, leaving the y-coordinate unchanged

$$\begin{aligned} x' &= -x \\ y' &= y \end{aligned} \tag{7.2}$$

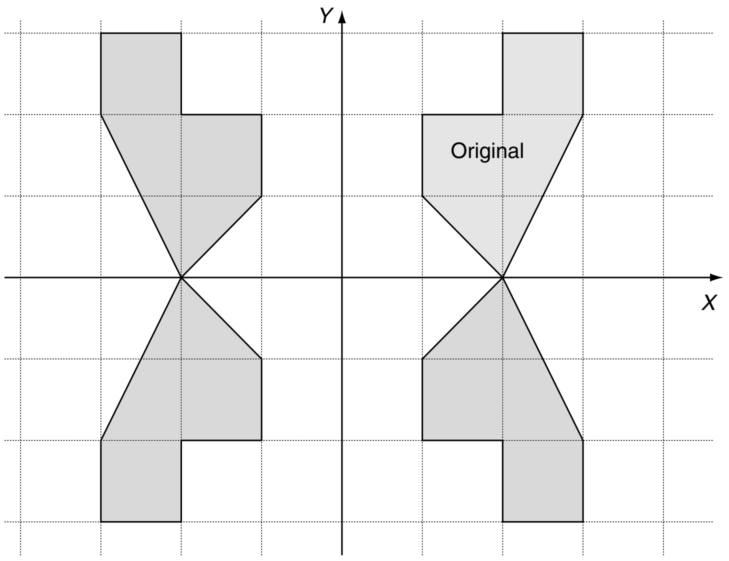


Fig. 7.3. The top right-hand shape can give rise to the three reflections simply by reversing the signs of coordinates.

and to reflect a shape relative to the x-axis we reverse the y-coordinates:

$$\begin{aligned} x' &= x \\ y' &= -y \end{aligned} \tag{7.3}$$

Examples of reflections are shown in Figure 7.3.

Before proceeding, we pause to introduce matrix notation so that we can develop further transformations using algebra and matrices simultaneously.

7.2 Matrices

Matrix notation was investigated by the British mathematician Arthur Cayley around 1858. Caley formalized matrix algebra, along with the American mathematicians Benjamin and Charles Pierce. Also, by the start of the 19th century Carl Gauss (1777–1855) had proved that transformations were not commutative, i.e. $\mathrm{T}_1 \times \mathrm{T}_2 \neq \mathrm{T}_2 \times \mathrm{T}_1$, and Caley's matrix notation would clarify such observations. For example, consider the transformation T_1:

$$\mathbf{T}_1 \begin{array}{ll} x' & = ax + by \\ y' & = cx + dy \end{array} \tag{7.4}$$

and another transformation T$_2$ that transforms T$_1$:

$$\mathbf{T}_2 \times \mathbf{T}_1 \quad \begin{aligned} x'' &= Ax' + By' \\ y'' &= Cx' + Dy' \end{aligned} \tag{7.5}$$

If we substitute the full definition of T$_1$ we get

$$\mathbf{T}_2 \times \mathbf{T}_1 \quad \begin{aligned} x'' &= A(ax + by) + B(cx + dy) \\ y'' &= C(ax + by) + D(cx + dy) \end{aligned} \tag{7.6}$$

which simplifies to

$$\mathbf{T}_2 \times \mathbf{T}_1 \quad \begin{aligned} x'' &= (Aa + Bc)x + (Ab + Bd)y \\ y'' &= (Ca + Dc)x + (Cb + Dd)y \end{aligned} \tag{7.7}$$

Caley proposed separating the constants from the variables, as follows:

$$\mathbf{T}_1 \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} \tag{7.8}$$

where the square matrix of constants in the middle determines the transformation. The algebraic form is recreated by taking the top variable x', introducing the = sign, and multiplying the top row of constants $[a\ b]$ individually by the last column vector containing x and y. We then examine the second variable y', introduce the = sign, and multiply the bottom row of constants $[c\ d]$ individually by the last *column* vector containing x and y, to create

$$\begin{aligned} x' &= ax + by \\ y' &= cx + dy \end{aligned} \tag{7.9}$$

Using Caley's notation, the product T$_2 \times$ T$_1$ is

$$\begin{bmatrix} x'' \\ y'' \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \cdot \begin{bmatrix} x' \\ y' \end{bmatrix} \tag{7.10}$$

But the notation also intimated that

$$\begin{bmatrix} x'' \\ y'' \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \cdot \begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} \tag{7.11}$$

and when we multiply the two *inner matrices* together they must produce

$$\begin{aligned} x'' &= (Aa + Bc)x + (Ab + Bd)y \\ y'' &= (Ca + Dc)x + (Cb + Dd)y \end{aligned} \tag{7.12}$$

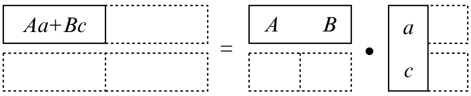
or in matrix form

$$\begin{bmatrix} x'' \\ y'' \end{bmatrix} = \begin{bmatrix} Aa + Bc & Ab + Bd \\ Ca + Dc & Cb + Dd \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} \tag{7.13}$$

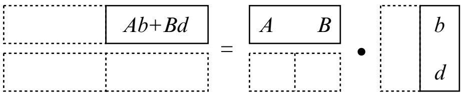
otherwise the two systems of notation will be inconsistent. This implies that

$$\begin{bmatrix} Aa+Bc & Ab+Bd \\ Ca+Dc & Cb+Dd \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \cdot \begin{bmatrix} a & b \\ c & d \end{bmatrix} \tag{7.14}$$

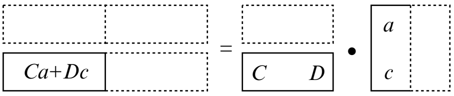
which demonstrates how matrices must be multiplied. Here are the rules for matrix multiplication:



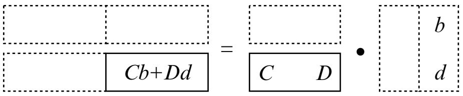
1 The top left-hand corner element $Aa+Bc$ is the product of the top row of the first matrix by the left column of the second matrix.



2 The top right-hand element $Ab+Bd$ is the product of the top row of the first matrix by the right column of the second matrix.



3 The bottom left-hand element $Ca+Dc$ is the product of the bottom row of the first matrix by the left column of the second matrix.



4 The bottom right-hand element $Cb+Dd$ is the product of the bottom row of the first matrix by the right column of the second matrix.

It is now a trivial exercise to confirm Gauss's observation that $\mathrm{T}_1 \times \mathrm{T}_2 \neq \mathrm{T}_2 \times \mathrm{T}_1$, because if we reverse the transforms $\mathrm{T}_2 \times \mathrm{T}_1$ to $\mathrm{T}_1 \times \mathrm{T}_2$ we get

$$\begin{bmatrix} Aa+Bc & Ab+Bd \\ Ca+Dc & Cb+Dd \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} A & B \\ C & D \end{bmatrix} \tag{7.15}$$

which shows conclusively that the product of two transforms is not commutative.

One immediate problem with this notation is that there is no apparent mechanism to add or subtract a constant such as c or f:

$$\begin{aligned} x' &= ax + by + c \\ y' &= dx + ey + f \end{aligned} \tag{7.16}$$

Mathematicians resolved this in the 19th century, by the use of *homogeneous coordinates*. But before we look at this idea, it must be pointed out that currently there are two systems of matrix notation in use.

7.2.1 Systems of Notation

Over the years, two systems of matrix notation have evolved: one where the matrix multiplies a column vector, as described above, and another where a *row vector* multiplies the matrix:

$$[x' \quad y'] = [x \quad y].\begin{bmatrix} a & c \\ b & d \end{bmatrix} \tag{7.17}$$

Note how the elements of the matrix are transposed to accommodate the algebraic correctness of the transformation. There is no preferred system of notation, and you will find technical books and papers supporting both. For example, *Computer Graphics: Principles and Practice* (Foley *et al.*, 1990) employs the column vector notation, whereas the *Gems* books (Glassner *et al.*, 1990) employ the row vector notation. The important thing to remember is that the rows and columns of the matrix are transposed when moving between the two systems.

7.2.2 The Determinant of a Matrix

The *determinant* of a 2×2 matrix is a scalar quantity computed. Given a matrix

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

its determinant is $ad - cb$ and is represented by

$$\begin{vmatrix} a & b \\ c & d \end{vmatrix} \tag{7.18}$$

For example, the determinant of $\begin{bmatrix} 3 & 2 \\ 1 & 2 \end{bmatrix}$ is $3 \times 2 - 1 \times 2 = 4$

Later, we will discover that the determinant of a 2×2 matrix determines the change in area that occurs when a polygon is transformed by the matrix. For example, if the determinant is 1, there is no change in area, but if the determinant is 2, the polygon's area is doubled.

7.3 Homogeneous Coordinates

Homogeneous coordinates surfaced in the early 19th century, when they were independently proposed by Möbius (who also invented a one-sided curled band, the Möbius strip), Feuerbach, Bobillier, and Plücker. Möbius named them *barycentric coordinates.* They have also been called *areal coordinates* because of their area-calculating properties.

Basically, homogeneous coordinates define a point in a plane using three coordinates instead of two. Initially, Plücker located a homogeneous point relative to the sides of a triangle, but later revised his notation to the one employed in contemporary mathematics and computer graphics. This states that for a point P with coordinates $(x,\ y)$ there exists a homogeneous point $(x,\ y,\ t)$ such that $X = x/t$ and $Y = y/t$. For example, the point (3, 4) has homogeneous coordinates (6, 8, 2), because $3 = 6/2$ and $4 = 8/2$. But the homogeneous point (6, 8, 2) is not unique to (3, 4); (12, 16, 4), (15, 20, 5) and (300, 400, 100) are all possible homogeneous coordinates for (3, 4).

The reason why this coordinate system is called 'homogeneous' is because it is possible to transform functions such as $f(x,\ y)$ into the form $f(x/t,\ y/t)$ without disturbing the degree of the curve. To the non-mathematician this may not seem anything to get excited about, but in the field of projective geometry it is a very powerful concept.

For our purposes, we can imagine that a collection of homogeneous points of the form $(x,\ y,\ t)$ exist on an xy-plane where t is the z-coordinate, as illustrated in Figure 7.4. The figure shows a triangle on the $t = 1$ plane, and a similar triangle, much larger, on a more distant plane. Thus instead of working in two dimensions, we can work on an arbitrary xy-plane in three dimensions. The t- or z-coordinate of the plane is immaterial because the x- and y-coordinates are eventually scaled by t. However, to keep things simple it seems a good idea to choose $t = 1$. This means that the point $(x,\ y)$ has homogeneous coordinates $(x,\ y,\ 1)$, making scaling unnecessary.

If we substitute 3D homogeneous coordinates for traditional 2D Cartesian coordinates, we must attach a 1 to every $(x,\ y)$ pair. When a point $(x,\ y,\ 1)$ is transformed, it will emerge as $(x', y', 1)$, and we discard the 1. This may seem a futile exercise, but it resolves the problem of creating a translation transformation.

Consider the following transformation on the homogeneous point $(x,\ y,\ 1)$:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.19}$$

This expands to

$$\begin{aligned} x' &= ax + by + c \\ y' &= dx + ey + f \\ 1 &= 1 \end{aligned} \tag{7.20}$$

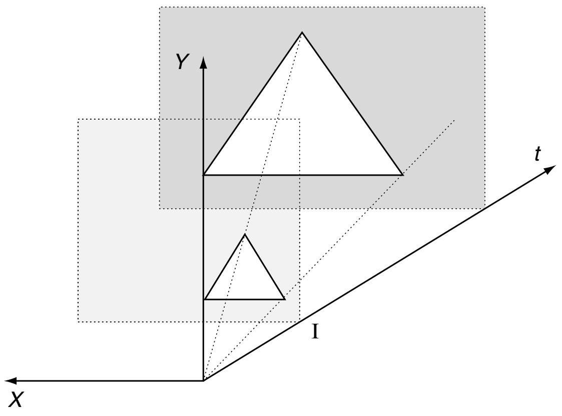


Fig. 7.4. 2D homogeneous coordinates can be visualized as a plane in 3D space, generally where $t = 1$, for convenience.

which solves the above problem of adding a constant.

Let's now go on to see how homogeneous coordinates are used in practice.

7.3.1 2D Translation

The algebraic and matrix notation for 2D translation is

$$\begin{aligned} x' &= x + t_x \\ y' &= y + t_y \end{aligned} \tag{7.21}$$

or, using matrices,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.22}$$

7.3.2 2D Scaling

The algebraic and matrix notation for 2D scaling is

$$\begin{aligned} x' &= s_x x \\ y' &= s_y y \end{aligned} \tag{7.23}$$

or, using matrices,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.24}$$

The scaling action is relative to the origin, i.e. the point (0,0) remains (0,0) All other points move away from the origin. To scale relative to another point (p_x, p_y) we first subtract (p_x, p_y) from (x, y) respectively. This effectively translates the reference point (p_x, p_y) back to the origin. Second, we perform the scaling operation, and third, add (p_x, p_y) back to (x, y) respectively, to compensate for the original subtraction. Algebraically this is

$$\begin{aligned} x' &= s_x(x - p_x) + p_x \\ y' &= s_y(y - p_y) + p_y \end{aligned} \tag{7.25}$$

which simplifies to

$$\begin{aligned} x' &= s_x x + p_x(1 - s_x) \\ y' &= s_y y + p_y(1 - s_y) \end{aligned} \tag{7.26}$$

or in a homogeneous matrix form

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & p_x(1 - s_x) \\ 0 & s_y & p_y(1 - s_y) \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.27}$$

For example, to scale a shape by 2 relative to the point (1, 1) the matrix is

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 2 & 0 & -1 \\ 0 & 2 & -1 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

7.3.3 2D Reflections

The matrix notation for reflecting about the y-axis is:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.28}$$

or about the x-axis

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.29}$$

However, to make a reflection about an arbitrary vertical or horizontal axis we need to introduce some more algebraic deception. For example, to make a reflection about the vertical axis $x = 1$, we first subtract 1 from the x-coordinate. This effectively makes the $x = 1$ axis coincident with the major y-axis. Next we perform the reflection by reversing the sign of the modified

x-coordinate. And finally, we add 1 to the reflected coordinate to compensate for the original subtraction. Algebraically, the three steps are

$$\begin{aligned} x_1 &= x - 1 \\ x_2 &= -(x-1) \\ x' &= -(x-1)+1 \end{aligned}$$

which simplifies to

$$\begin{aligned} x' &= -x + 2 \\ y' &= y \end{aligned} \tag{7.30}$$

or in matrix form,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.31}$$

Figure 7.5 illustrates this process.

In general, to reflect a shape about an arbitrary y-axis, $y = a_x$, the following transform is required:

$$\begin{aligned} x' &= -(x - a_x) + a_x \quad = -x + 2a_x \\ y' &= y \end{aligned} \tag{7.32}$$

or, in matrix form,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 2a_x \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.33}$$

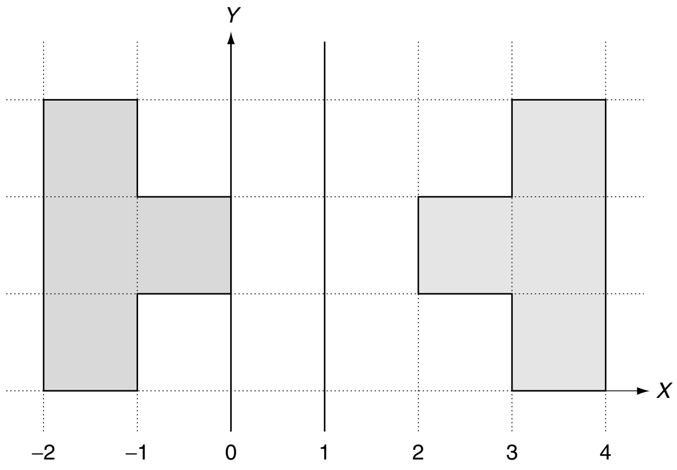


Fig. 7.5. The shape on the right is reflected about the $x = 1$ axis.

Similarly, this transform is used for reflections about an arbitrary x-axis, $y = a_y$:

$$\begin{aligned} x' &= x \\ y' &= -(y - a_y) + a_y = -y + 2a_y \end{aligned} \tag{7.34}$$

or, in matrix form,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 2a_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.35}$$

7.3.4 2D Shearing

A shape is sheared by leaning it over at an angle β. Figure 7.6 illustrates the geometry, and we see that the y-coordinate remains unchanged but the x-coordinate is a function of y and $\tan(\beta)$.

$$\begin{aligned} x' &= x + y\tan(\beta) \\ y' &= y \end{aligned} \tag{7.36}$$

or, in matrix form,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & \tan(\beta) & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.37}$$

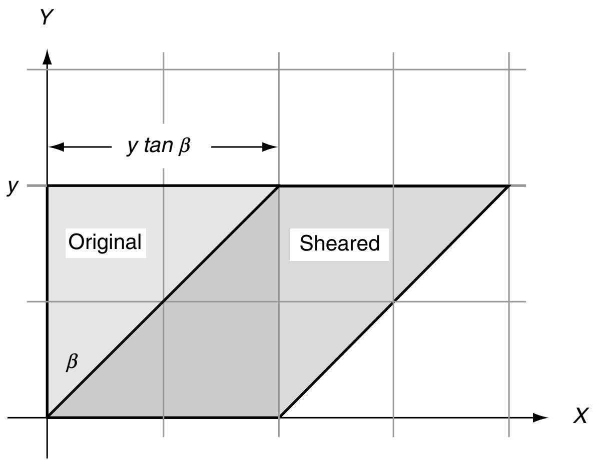


Fig. 7.6. The original square shape is sheared to the right by an angle β, and the horizontal shift is proportional to $ytan(\beta)$.

7.3.5 2D Rotation

Figure 7.7 shows a point $P(x, y)$ which is to be rotated by an angle β about the origin to $P'(x', y')$. It can be seen that

$$\begin{aligned} x' &= R\cos(\theta+\beta) \\ y' &= R\sin(\theta+\beta) \end{aligned} \tag{7.38}$$

therefore

$$\begin{aligned} x' &= R(\cos(\theta)\cos(\beta) - \sin(\theta)\sin(\beta)) \\ y' &= R(\sin(\theta)\cos(\beta) + \cos(\theta)\sin(\beta)) \\ x' &= R\left(\frac{x}{R}\cos(\beta) - \frac{y}{R}\sin(\beta)\right) \\ y' &= R\left(\frac{y}{R}\cos(\beta) + \frac{x}{R}\sin(\beta)\right) \\ x' &= x\cos(\beta) - y\sin(\beta) \\ y' &= x\sin(\beta) + y\cos(\beta) \end{aligned} \tag{7.39}$$

or, in matrix form,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\beta) & -\sin(\beta) & 0 \\ \sin(\beta) & \cos(\beta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.40}$$

For example, to rotate a point by 90° the matrix becomes

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

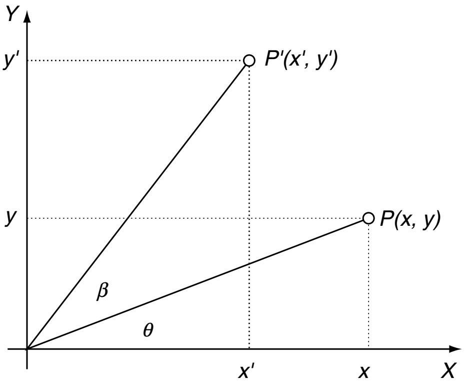


Fig. 7.7. The point $P(x, y)$ is rotated through an angle β to $P'(x', y')$.

Thus the point (1, 0) becomes (0, 1). If we rotate by 360° the matrix becomes

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Such a matrix has a null effect and is called an *identity matrix.*

To rotate a point $(x,\ y)$ about an arbitrary point (p_x, p_y) we first subtract (p_x, p_y) from the coordinates $(x,\ y)$ respectively. This enables us to perform the rotation about the origin. Second, we perform the rotation, and third, we add (p_x, p_y) to compensate for the original subtraction. Here are the steps:

1 Subtract (p_x, p_y):

$$\begin{aligned} x_1 &= (x - p_x) \\ y_1 &= (y - p_y) \end{aligned}$$

2 Rotate β about the origin:

$$\begin{aligned} x_2 &= (x - p_x)\cos(\beta) - (y - p_y)\sin(\beta) \\ y_2 &= (x - p_x)\sin(\beta) + (y - p_y)\cos(\beta) \end{aligned}$$

3 Add (p_x, p_y):

$$\begin{aligned} x' &= (x - p_x)\cos(\beta) - (y - p_y)\sin(\beta) + p_x \\ y' &= (x - p_x)\sin(\beta) + (y - p_y)\cos(\beta) + p_y \end{aligned}$$

Simplifying,

$$\begin{aligned} x' &= x\cos(\beta) - y\sin(\beta) + p_x(1 - \cos(\beta)) + p_y\sin(\beta) \\ y' &= x\sin(\beta) + y\cos(\beta) + p_y(1 - \cos(\beta)) - p_x\sin(\beta) \end{aligned} \tag{7.41}$$

and, in matrix form,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\beta) & -\sin(\beta) & p_x(1 - \cos(\beta)) + p_y\sin(\beta) \\ \sin(\beta) & \cos(\beta) & p_y(1 - \cos(\beta)) - p_x\sin(\beta) \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.42}$$

If we now consider rotating a point 90° about the point (1, 1) the matrix operation becomes

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 2 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

A simple test is to substitute the point (2, 1) for $(x,\ y)$: it is transformed correctly to (1, 2).

The algebraic approach in deriving the above transforms is relatively easy. However, it is also possible to use matrices to derive compound transformations, such as a reflection relative to an arbitrary line and scaling and rotation relative to an arbitrary point. These transformations are called *affine*, as parallel lines remain parallel after being transformed. One cannot always guarantee that angles and lengths are preserved, as the scaling transformation can alter these when different x and y scaling factors are used. For completeness, we will repeat these transformations from a matrix perspective.

7.3.6 2D Scaling

The strategy we used to scale a point $(x,\ y)$ relative to some arbitrary point (p_x, p_y) was to first, translate $(-p_x, -p_y)$; second, perform the scaling; and third, translate (p_x, p_y). These three transforms can be represented in matrix form as follows:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = [\text{translate}(p_x, p_y)] \cdot [\text{scale}(s_x, s_y)] \cdot [\text{translate}(-p_x, -p_y)] \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

which expands to

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & p_x \\ 0 & 1 & p_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -p_x \\ 0 & 1 & -p_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.43}$$

Note the sequence of the transforms, as this often causes confusion. The first transform acting on the point $(x,\ y,\ 1)$ is translate $(-p_x, -p_y)$, followed by scale (s_x, s_y), followed by translate (p_x, p_y). If they are placed in any other sequence, you will discover, like Gauss, that transforms are not commutative!

We can now concatenate these matrices into a single matrix by multiplying them together. This can be done in any sequence, so long as we preserve the original order. Let's start with scale (s_x, s_y) and translate $(-p_x, -p_y)$. This produces

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & p_x \\ 0 & 1 & p_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_x & 0 & -s_x p_x \\ 0 & s_y & -s_y p_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

and finally

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & p_x(1 - s_x) \\ 0 & s_y & p_y(1 - s_y) \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.44}$$

which is the same as the previous transform (7.27).

7.3.7 2D Reflections

A reflection about the y-axis is given by

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.45}$$

Therefore, using matrices, we can reason that a reflection transform about an arbitrary axis $x = a_x$, parallel with the y-axis, is given by

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = [\text{translate}(a_x, 0)] \cdot [\text{reflection}] \cdot [\text{translate}(-a_x, 0)] \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

which expands to

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & a_x \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -a_x \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

We can now concatenate these matrices into a single matrix by multiplying them together. Let's begin by multiplying the reflection and the translate $(-a_x, 0)$ matrices together. This produces

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & a_x \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} -1 & 0 & a_x \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

and finally

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 2a_x \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.46}$$

which is the same as the previous transform (7.33).

7.3.8 2D Rotation about an Arbitrary Point

A rotation about the origin is given by

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\beta) & -\sin(\beta) & 0 \\ \sin(\beta) & \cos(\beta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.47}$$

Therefore, using matrices, we can develop a rotation about an arbitrary point (p_x, p_y) as follows:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = [\text{translate}(p_x,\ p_y)] \cdot [\text{rotate } \beta] \cdot [\text{translate}(-p_x, -p_y)] \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

which expands to

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & p_x \\ 0 & 1 & p_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos(\beta) & -\sin(\beta) & 0 \\ \sin(\beta) & \cos(\beta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -p_x \\ 0 & 1 & -p_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

We can now concatenate these matrices into a single matrix by multiplying them together. Let's begin by multiplying the rotate β and the translate $(-p_x, -p_y)$ matrices together. This produces

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & p_x \\ 0 & 1 & p_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos(\beta) & -\sin(\beta) & -p_x\cos(\beta) + p_y\sin(\beta) \\ \sin(\beta) & \cos(\beta) & -p_x\sin(\beta) - p_y\cos(\beta) \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

and finally

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\beta) & -\sin(\beta) & p_x(1-\cos(\beta)) + p_y\sin(\beta) \\ \sin(\beta) & \cos(\beta) & p_y(1-\cos(\beta)) - p_x\sin(\beta) \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.48}$$

which is the same as the previous transform (7.42).

I hope it is now is obvious to the reader that one can derive all sorts of transforms either algebraically, or by using matrices – it is just a question of convenience.

7.4 3D Transformations

Now we come to transformations in three dimensions, where we apply the same reasoning as in two dimensions. Scaling and translation are basically the same, but where in 2D we rotated a shape about a point, in 3D we rotate an object about an axis.

7.4.1 3D Translation

The algebra is so simple for 3D translation that we can write the homogeneous matrix directly:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.49}$$

7.4.2 3D Scaling

The algebra for 3D scaling is

$$\begin{aligned} x' &= s_x x \\ y' &= s_y y \\ z' &= s_z z \end{aligned} \tag{7.50}$$

which in matrix form is

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.51}$$

The scaling is relative to the origin, but we can arrange for it to be relative to an arbitrary point (p_x, p_y, p_z) with the following algebra:

$$\begin{aligned} x' &= s_x(x - p_x) + p_x \\ y' &= s_y(y - p_y) + p_y \\ z' &= s_z(z - p_z) + p_z \end{aligned} \tag{7.52}$$

which in matrix form is

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 & p_x(1 - s_x) \\ 0 & s_y & 0 & p_y(1 - s_y) \\ 0 & 0 & s_z & p_z(1 - s_z) \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.53}$$

7.4.3 3D Rotations

In two dimensions a shape is rotated about a point, whether it be the origin or some arbitrary position. In three dimensions an object is rotated about an axis, whether it be the x-, y- or z-axis, or some arbitrary axis. To begin with, let's look at rotating a vertex about one of the three orthogonal axes; such rotations are called *Euler rotations* after the Swiss mathematician Leonhard Euler (1707–1783).

Recall that a general 2D-rotation transform is given by

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\beta) & -\sin(\beta) & 0 \\ \sin(\beta) & \cos(\beta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.54}$$

which in 3D can be visualized as rotating a point $P(x, y, z)$ on a plane parallel with the xy-plane as shown in Figure 7.8. In algebraic terms this can be written as

$$\begin{aligned} x' &= x\cos(\beta) - y\sin(\beta) \\ y' &= x\sin(\beta) + y\cos(\beta) \\ z' &= z \end{aligned} \tag{7.55}$$

Therefore, the 3D transform can be written as

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\beta) & -\sin(\beta) & 0 & 0 \\ \sin(\beta) & \cos(\beta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.56}$$

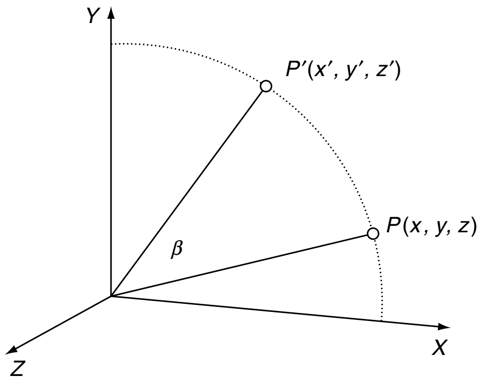


Fig. 7.8. Rotating P about the z-axis.

which basically rotates a point about the z-axis.

When rotating about the x-axis, the x-coordinate remains constant while the y- and z-coordinates are changed. Algebraically, this is

$$\begin{aligned} x' &= x \\ y' &= y\cos(\beta) - z\sin(\beta) \\ z' &= y\sin(\beta) + z\cos(\beta) \end{aligned} \tag{7.57}$$

or, in matrix form,

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\beta) & -\sin(\beta) & 0 \\ 0 & \sin(\beta) & \cos(\beta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.58}$$

When rotating about the y-axis, the y-coordinate remains constant while the x- and z-coordinates are changed. Algebraically, this is

$$\begin{aligned} x' &= z\sin(\beta) + x\cos(\beta) \\ y' &= y \\ z' &= z\cos(\beta) - x\sin(\beta) \end{aligned} \tag{7.59}$$

or, in matrix form,

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.60}$$

Note that the matrix terms do not appear to share the symmetry seen in the previous two matrices. Nothing has really gone wrong, it is just the way the axes are paired together to rotate the coordinates.

The above rotations are also known as *yaw*, *pitch* and *roll*. Great care should be taken with these terms when referring to other books and technical papers. Sometimes a left-handed system of axes is used rather than a right-handed set, and the vertical axis may be the y-axis or the z-axis.

Consequently, the matrices representing the rotations can vary greatly. In this text all Cartesian coordinate systems are right-handed, and the vertical axis is always the y-axis.

The roll, pitch and yaw angles can be defined as follows:

- *roll* is the angle of rotation about the z-axis
- *pitch* is the angle of rotation about the x-axis
- *yaw* is the angle of rotation about the y-axis.

Figure 7.9 illustrates these rotations and the sign convention. The homogeneous matrices representing these rotations are as follows:

- rotate *roll* about the z-axis:

$$\begin{bmatrix} \cos(roll) & -\sin(roll) & 0 & 0 \\ \sin(roll) & \cos(roll) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7.61}$$

- rotate *pitch* about the x-axis:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(pitch) & -\sin(pitch) & 0 \\ 0 & \sin(pitch) & \cos(pitch) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7.62}$$

- rotate *yaw* about the y-axis:

$$\begin{bmatrix} \cos(yaw) & 0 & \sin(yaw) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(yaw) & 0 & \cos(yaw) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7.63}$$

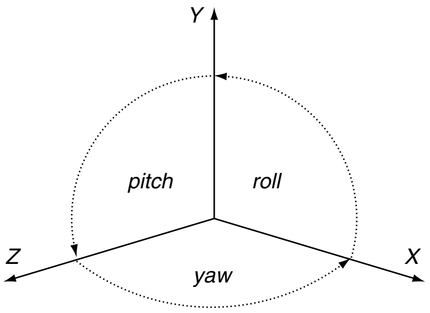


Fig. 7.9. The convention for *roll*, *pitch* and *yaw* angles.

A common sequence for applying these rotations is *roll, pitch, yaw*, as seen in the following transform:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = [yaw] \cdot [pitch] \cdot [roll] \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.64}$$

and if a translation is involved,

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = [translate] \cdot [yaw] \cdot [pitch] \cdot [roll] \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.65}$$

When these rotation transforms are applied, the vertex is first rotated about the z-axis (roll), followed by a rotation about the x-axis (pitch), followed by a rotation about the y-axis (yaw). Euler rotations are relative to the fixed frame of reference. This is not always easy to visualize, as one's attention is normally with the rotating frame of reference. Let's consider a simple example where an axial system is subjected to a pitch rotation followed by a yaw rotation relative to fixed frame of reference.

We begin with two frames of reference XYZ and $X'Y'Z'$ mutually aligned. Figure 7.10 shows the orientation of $X'Y'Z'$ after it is subjected to a pitch of 90° about the x-axis. Figure 7.11 shows the the final orientation after $X'Y'Z'$ is subjected to a yaw of 90° about the y-axis.

7.4.4 Gimbal Lock

Let's take another example starting from the point where the two axial systems are mutually aligned. Figure 7.12 shows the orientation of $X'Y'Z'$ after it is subjected to a roll of 45° about the z-axis, and Figure 7.13 shows the orientation of $X'Y'Z'$ after it is subjected to a pitch of 90° about the x-axis. Now the interesting thing about this orientation is that if we now performed

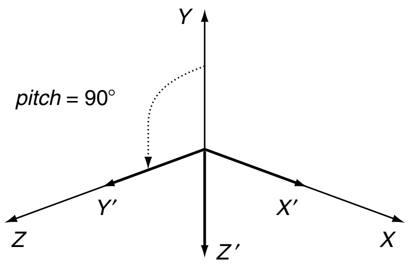


Fig. 7.10. The $X'Y'Z'$ axial system after a *pitch* of 90°.

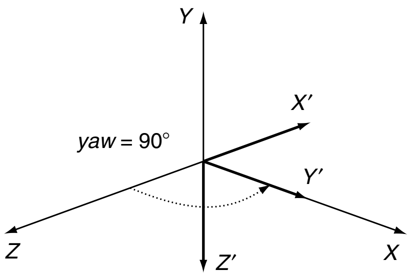


Fig. 7.11. The $X'Y'Z'$ axial system after a *yaw* of 90°.

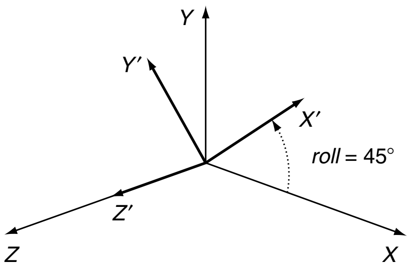


Fig. 7.12. The $X'Y'Z'$ axial system after a *roll* of 45°.

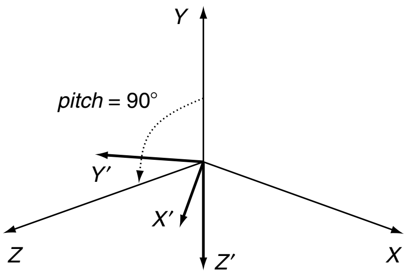


Fig. 7.13. The $X'Y'Z'$ axial system after a *pitch* of 90°.

a yaw of 45° about the z-axis, it would rotate the x′-axis towards the x-axis, counteracting the effect of the original roll. yaw has become a negative roll rotation, caused by the 90° pitch. This situation is known as *gimbal lock*, because one degree of rotational freedom has been lost. Quite innocently, we have stumbled across one of the major weaknesses of Euler angles: under certain conditions it is only possible to rotate an object about two axes. One way of preventing this is to create a secondary set of axes constructed from three orthogonal vectors that are also rotated alongside an object or virtual camera. But instead of making the rotations relative to the fixed frame of reference, the roll, pitch and yaw rotations are relative to the rotating frame of reference. Another method is to use quaternions, which will be investigated later in this chapter.

7.4.5 Rotating about an Axis

The above rotations were relative to the x-, y- and z-axes. Now let's consider rotations about an axis parallel to one of these axes. To begin with, we will rotate about an axis parallel with the z-axis, as shown in Figure 7.14. The scenario is very reminiscent of the 2D case for rotating a point about an arbitrary point, and the general transform is given by

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = [translate(p_x, p_y, 0)].[rotate\beta].[translate(-p_x, -p_y, 0)].\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.66}$$

and the matrix is

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\beta) & -\sin(\beta) & 0 & p_x(1-\cos(\beta)) + p_y\sin(\beta) \\ \sin(\beta) & \cos(\beta) & 0 & p_y(1-\cos(\beta)) - p_x\sin(\beta) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.67}$$

I hope you can see the similarity between rotating in 3D and 2D: the x- and y-coordinates are updated while the z-coordinate is held constant. We can

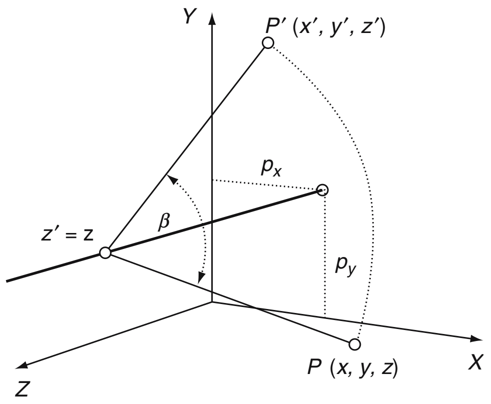


Fig. 7.14. Rotating a point about an axis parallel with the z-axis.

now state the other two matrices for rotating about an axis parallel with the x-axis and parallel with the y-axis:

- rotating about an axis parallel with the x-axis:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\beta) & -\sin(\beta) & p_y(1-\cos(\beta)) + p_z\sin(\beta) \\ 0 & \sin(\beta) & \cos(\beta) & p_z(1-\cos(\beta)) - p_y\sin(\beta) \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.68}$$

- rotating about an axis parallel with the y-axis:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) & p_x(1-\cos(\beta)) - p_z\sin(\beta) \\ 0 & 1 & 0 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) & p_z(1-\cos(\beta)) + p_x\sin(\beta) \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.69}$$

7.4.6 3D Reflections

Reflections in 3D occur with respect to a plane, rather than an axis. The matrix giving the reflection relative to the yz-plane is

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.70}$$

and the reflection relative to a plane parallel to, and a_x units from, the yz-plane is

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 2a_x \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.71}$$

It is left to the reader to develop similar matrices for the other major axial planes.

7.5 Change of Axes

Points in one coordinate system often have to be referenced in another one. For example, to view a 3D scene from an arbitrary position, a virtual camera is positioned in the world space using a series of transformations. An object's coordinates, which are relative to the world frame of reference, are computed relative to the camera's axial system, and then used to develop a perspective projection. Before explaining how this is achieved in 3D, let's examine the simple case of changing axial systems in two dimensions.

7.5.1 2D Change of Axes

Figure 7.15 shows a point $P(x,\ y)$ relative to the XY-axes, but we require to know the coordinates relative to the $X'Y'$-axes. To do this, we need to know the relationship between the two coordinate systems, and ideally we want to apply a technique that works in 2D and 3D. If the second coordinate system is a simple translation (t_x, t_y) relative to the reference system, as shown in Figure 7.15, the point $P(x,\ y)$ has coordinates relative to the translated system $(x - t_x, y - t_y)$:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.72}$$

If the $X'Y'$-axes are rotated β relative to the XY-axes, as shown in Figure 7.16, a point $P(x,\ y)$ relative to the XY-axes has coordinates (x', y') relative to the rotated axes given by

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(-\beta) & -\sin(-\beta) & 0 \\ \sin(-\beta) & \cos(-\beta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

which simplifies to

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\beta) & \sin(\beta) & 0 \\ -\sin(\beta) & \cos(\beta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.73}$$

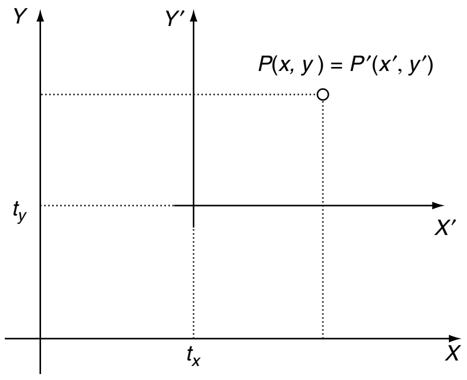


Fig. 7.15. The $X'Y'$-axes are translated by (t_x, t_y).

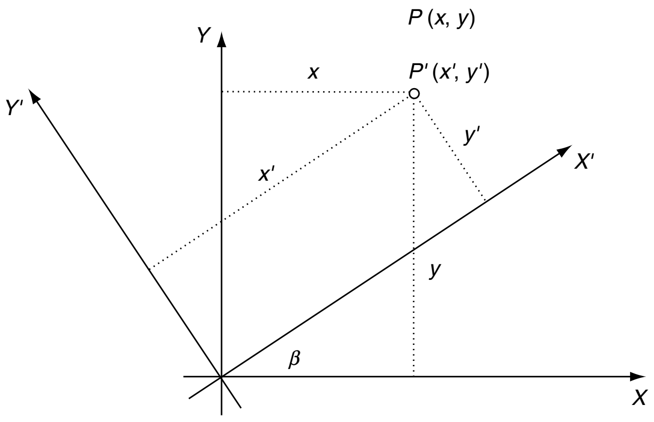


Fig. 7.16. The secondary set of axes are rotated by β.

When a coordinate system is rotated and translated relative to the reference system, a point $P(x,\ y)$ has coordinates (x', y') relative to the new axes given by

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\beta) & \sin(\beta) & 0 \\ -\sin(\beta) & \cos(\beta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

which simplifies to

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\beta) & \sin(\beta) & -t_x\cos(\beta) - t_y\sin(\beta) \\ -\sin(\beta) & \cos(\beta) & t_x\sin(\beta) - t_y\cos(\beta) \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{7.74}$$

7.6 Direction Cosines

Direction cosines are the cosines of the angles between a vector and the axes, and for unit vectors they are the vector's components. Figure 7.17 shows two unit vectors X' and Y', and by inspection the direction cosines for X' are $\cos(\beta)$ and $\cos(90° - \beta)$, which can be rewritten as $\cos(\beta)$ and $\sin(\beta)$, and the direction cosines for Y' $\cos(90° + \beta)$ and $\cos(\beta)$, which can be rewritten as $-\sin(\beta)$ and $\cos(\beta)$. But these direction cosines $\cos(\beta), \sin(\beta), -\sin(\beta)$ and $\cos(\beta)$ are the four elements of the rotation matrix used above:

$$\begin{bmatrix} \cos(\beta) & \sin(\beta) \\ -\sin(\beta) & \cos(\beta) \end{bmatrix} \tag{7.75}$$

The top row contains the direction cosines for the X'-axis and the bottom row contains the direction cosines for the Y'-axis. This relationship also holds in 3D.

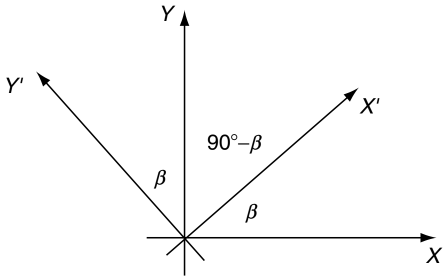


Fig. 7.17. If the X'- and Y'-axes are assumed to be unit vectors their direction cosines form the elements of the rotation matrix.

Before exploring changes of axes in 3D let's evaluate a simple example in 2D where a set of axes is rotated 45° as shown in Figure 7.18. The appropriate transform is

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(45^\circ) & \sin(45^\circ) & 0 \\ -\sin(45^\circ) & \cos(45^\circ) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$
$$= \begin{bmatrix} 0.707 & 0.707 & 0 \\ -0.707 & 0.707 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

The four vertices on a unit square become

$$\begin{aligned} (0,0) &\rightarrow (0,0) \\ (1,0) &\rightarrow (0.707,-0.707) \\ (1,1) &\rightarrow (1.414,0) \\ (0,1) &\rightarrow (0.707,0.707) \end{aligned}$$

which inspection of Figure 7.18 shows to be correct.

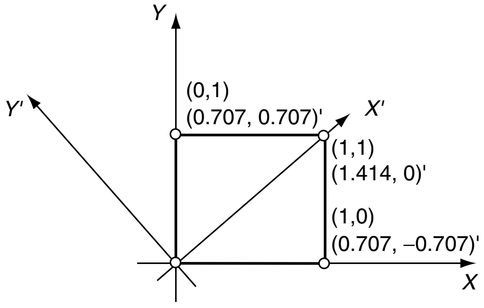


Fig. 7.18. The vertices of a unit square relative to the two axial systems.

7.6.1 Positioning the Virtual Camera

Four coordinate systems are used in the computer graphics pipeline: *object space*, *world space*, *camera space* and *image space*.

- The *object space* is a domain where objects are modelled and assembled.
- The *world space* is where objects are positioned and animated through appropriate transforms. The world space also hosts a virtual camera or observer.
- The *camera space* is a transform of the world space to the camera's point of view.
- Finally, *the image space* is a projection – normally perspective – of the camera space onto an image plane.

The transforms considered so far are used to manipulate and position objects within the world space. What we will consider next is how a virtual camera or observer is positioned in world space, and the process of converting world coordinates to camera coordinates. The procedure used generally depends on the method employed to define the camera's frame of reference within the world space, which may involve the use of direction cosines, Euler angles or quaternions. We will examine how each of these techniques could be implemented.

7.6.2 Direction Cosines

A 3D unit vector has three components $[x\ y\ z]^{\mathrm{T}}$, which are equal to the cosines of the angles formed between the vector and the three orthogonal axes. These angles are known as *direction cosines* and can be computed taking the dot product of the vector and the Cartesian unit vectors. Figure 7.19 shows the direction cosines and the angles. These direction cosines enable any point $P(x, y, z)$ in one frame of reference to be transformed into $P'(x', y', z')$ in another frame of reference as follows:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.76}$$

where: r_{11}, r_{12}, r_{13} are the direction cosines of the secondary x-axis
r_{21}, r_{22}, r_{23} are the direction cosines of the secondary y-axis
r_{31}, r_{32}, r_{33} are the direction cosines of the secondary z-axis.

To illustrate this operation, consider the situation shown in Figure 7.20 which shows two axial systems mutually aligned. Evaluating the direction cosines results in the following matrix transformation:

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

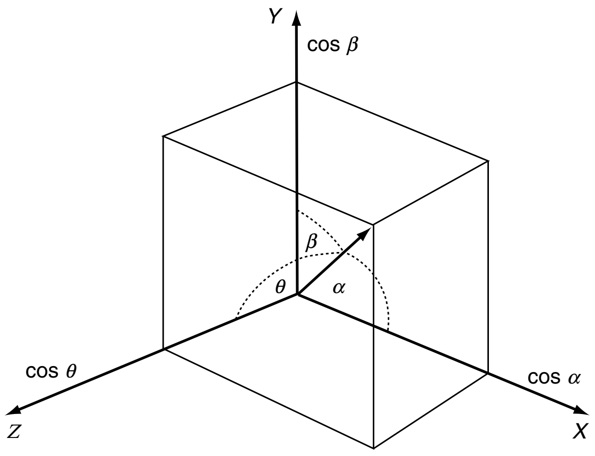


Fig. 7.19. The components of a unit vector are equal to the cosines of the angles between the vector and the axes.

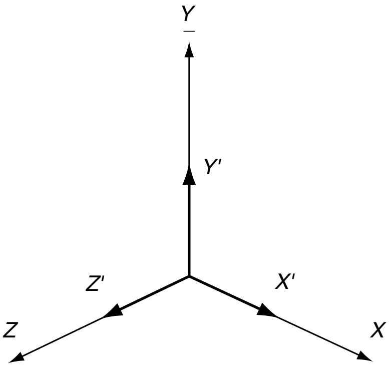


Fig. 7.20. Two axial systems mutually aligned.

which is the identity matrix and implies that $(x', y', z') = (x, y, z)$.

Figure 7.21 shows another situation, and the associated transform is

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

Substituting the $(1, 1, 0)$ for (x, y, z) produces values of $(1, -1, 0)$ for (x', y', z') in the new frame of reference, which by inspection is correct.

If the virtual camera is offset by (t_x, t_y, t_z) the transform relating points in world space to camera space can be expressed as a compound operation

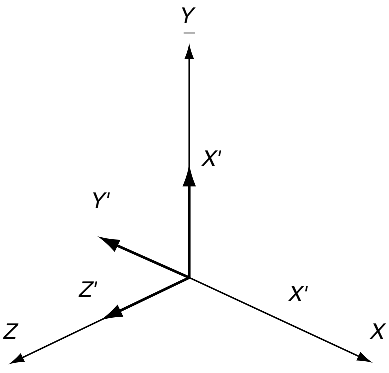


Fig. 7.21. The $X'Y'Z'$ axial system after a *roll* of 90°.

consisting of a translation back to the origin, followed by a change of axial systems. This can be expressed as

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & -t_x \\ 0 & 1 & 0 & -t_y \\ 0 & 0 & 1 & -t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.77}$$

As an example, consider the situation shown in Figure 7.22. The values of (t_x, t_y, t_z) are (10, 1, 1), and the direction cosines are as shown in the following matrix operation:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & -10 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

which concatenates to

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 10 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Substituting (0, 0, 0) for (x, y, z) in the above transform produces $(10, -1, 1)$ for (x', y', z'), which can be confirmed from Figure 7.22. Similarly, substituting (0, 1, 1) for (x, y, z) produces (10,0,0) for (x', y', z'), which is also correct.

7.6.3 Euler Angles

Another approach for locating the virtual camera involves Euler angles, but we must remember that they suffer from gimbal lock (see page 70). However, if

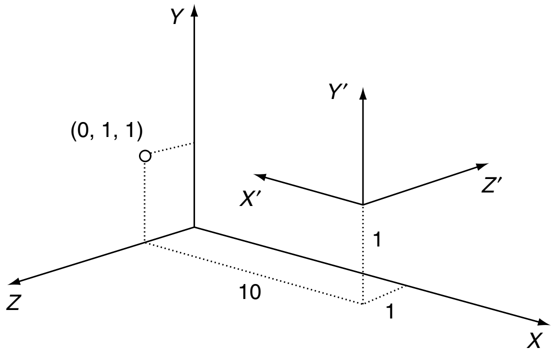


Fig. 7.22. The secondary axial system is subject to a *yaw* of 180° and an offset of (10, 1, 1).

the virtual camera is located in world space using Euler angles, the transform relating world coordinates to camera coordinates can be derived from the inverse operations. The *yaw*, *pitch*, *roll* matrices described above are called *orthogonal matrices*, as the inverse matrix is the transpose of the original rows and columns. Consequently, to rotate through angles –*roll*, –*pitch* and –*yaw*, we use

- rotate –*roll* about the z-axis:

$$\begin{bmatrix} \cos(roll) & \sin(roll) & 0 & 0 \\ -\sin(roll) & \cos(roll) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7.78}$$

- rotate –*pitch* about the x-axis:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(pitch) & \sin(pitch) & 0 \\ 0 & -\sin(pitch) & \cos(pitch) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7.79}$$

- rotate –*yaw* about the y-axis:

$$\begin{bmatrix} \cos(yaw) & 0 & -\sin(yaw) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(yaw) & 0 & \cos(yaw) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7.80}$$

The same result is obtained simply by substituting *–roll*, *–pitch*, *–yaw* in the original matrices. As described above, the virtual camera will normally be translated from the origin by (t_x, t_y, t_z), which implies that the transform from the world space to the camera space must be evaluated as follows:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = [-roll]\cdot[-pitch]\cdot[-yaw]\cdot[-translate(-t_x, -t_y, -t_z)\cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.81}$$

which can be represented by a single homogeneous matrix:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} & T_{13} & T_{14} \\ T_{21} & T_{22} & T_{23} & T_{24} \\ T_{31} & T_{32} & T_{33} & T_{34} \\ T_{41} & T_{42} & T_{43} & T_{44} \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.82}$$

where

$$\begin{aligned}
T_{11} &= \cos(yaw)\cos(roll) + \sin(yaw)\sin(pitch)\sin(roll) \\
T_{12} &= \cos(pitch)\sin(roll) \\
T_{13} &= -\sin(yaw)\cos(roll) + \cos(yaw)\sin(pitch)\sin(roll) \\
T_{14} &= -(t_x T_{11} + t_y T_{12} + t_z T_{13}) \\
T_{21} &= -\cos(yaw)\sin(roll) + \sin(yaw)\sin(pitch)\cos(roll) \\
T_{22} &= \cos(pitch)\cos(roll) \\
T_{23} &= \sin(yaw)\sin(roll) + \cos(yaw)\sin(pitch)\cos(roll) \\
T_{24} &= -(t_x T_{21} + t_y T_{22} + t_z T_{23}) \\
T_{31} &= \sin(yaw)\cos(pitch) \\
T_{32} &= -\sin(pitch) \\
T_{33} &= \cos(yaw)\cos(pitch) \\
T_{34} &= -(t_x T_{31} + t_y T_{32} + t_z T_{33}) \\
T_{41} &= 0 \\
T_{42} &= 0 \\
T_{43} &= 0 \\
T_{44} &= 1
\end{aligned} \tag{7.83}$$

This, too, can be verified by a simple example. For instance, consider the situation shown in Figure 7.22 where the following conditions prevail:

$$\begin{aligned}
roll &= 0^\circ \\
pitch &= 0^\circ \\
yaw &= 180^\circ \\
t_x &= 10
\end{aligned}$$

$$t_y = 1$$
$$t_z = 1$$

The transform is

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 10 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

which is identical to the equation used for direction cosines. Another example is shown in Figure 7.23, where the following conditions exist:

$$\begin{aligned} roll &= 90^\circ \\ pitch &= 180^\circ \\ yaw &= 0^\circ \\ t_x &= 0.5 \\ t_y &= 0.5 \\ t_z &= 11 \end{aligned}$$

The transform is

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 & 0.5 \\ -1 & 0 & 0 & 0.5 \\ 0 & 0 & -1 & 11 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Substituting $(1, 1, 1)$ for (x, y, z) produces $(-0.5, -0.5, 10)$ for (x', y', z'). Similarly, substituting $(0, 0, 1)$ for (x, y, z) produces $(0.5, 0.5, 10)$ for (x', y', z'), which can be visually verified from Figure 7.23.

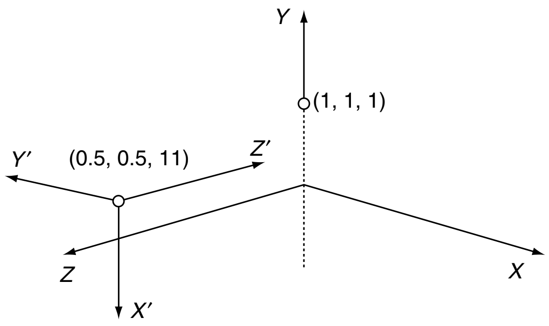


Fig. 7.23. The secondary axial system is subjected to a *roll* of 90°, a *pitch* of 180°, and a translation of (0.5, 0.5, 11).

7.7 Rotating a Point about an Arbitrary Axis

Let us now consider two ways of developing a matrix for rotating a point about an arbitrary axis. The first approach employs vector analysis and is quite succinct. The second technique, however, is less analytical and relies on matrices and trigonometric evaluation and is rather laborious. Fortunately, they both arrive at the same result!

Figure 7.24 shows three views of the geometry associated with the task at hand. The left-hand image illustrates the overall scenario; the middle image illustrates a side elevation; whilst the right-hand image illustrates a plan elevation.

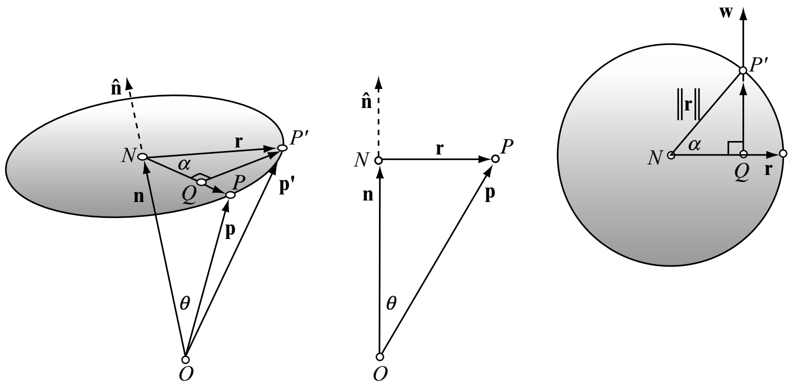


Fig. 7.24. Three views of the geometry associated with rotating a point about an arbitrary axis.

The axis of rotation is given by the unit vector $\hat{\mathbf{v}} = a\mathbf{i} + b\mathbf{j} + c\mathbf{k}$

$P(x_p, y_p, z_p)$ is the point to be rotated by angle α and $P'(x'_p, y'_p, z'_p)$ is the rotated point.

O is the origin, whilst $\mathbf{p}$ and $\mathbf{p}'$ are position vectors for P and P' respectively.

From Figure 7.24

$$\mathbf{p}' = \overrightarrow{ON} + \overrightarrow{NQ} + \overrightarrow{QP'}$$

To find $\overrightarrow{ON}$

$$\|\mathbf{n}\| = \|\mathbf{p}\| \cos\theta = \hat{\mathbf{n}}\cdot\mathbf{p}$$

therefore

$$\overrightarrow{ON} = \mathbf{n} = \hat{\mathbf{n}}(\hat{\mathbf{n}}\cdot\mathbf{p})$$

To find $\overrightarrow{NQ}$

$$\overrightarrow{NQ} = \frac{NQ}{NP}\mathbf{r} = \frac{NQ}{NP'}\mathbf{r} = \cos\alpha \cdot \mathbf{r}$$

but

$$\mathbf{p} = \mathbf{n} + \mathbf{r} = \hat{\mathbf{n}}(\hat{\mathbf{n}}\bullet\mathbf{p}) + \mathbf{r}$$

therefore

$$\mathbf{r} = \mathbf{p} - \hat{\mathbf{n}}(\hat{\mathbf{n}}\bullet\mathbf{p})$$

and

$$\overrightarrow{NQ} = [\mathbf{p} - \hat{\mathbf{n}}(\hat{\mathbf{n}}\bullet\mathbf{p})]\cos\alpha$$

To find $\overrightarrow{QP'}$

Let

$$\hat{\mathbf{n}} \times \mathbf{p} = \mathbf{w}$$

where

$$\|\mathbf{w}\| = \|\hat{\mathbf{n}}\| \cdot \|\mathbf{p}\|\sin\theta = \|\mathbf{p}\|\sin\theta$$

but

$$\|\mathbf{r}\| = \|\mathbf{p}\|\sin\theta$$

therefore

$$\|\mathbf{w}\| = \|\mathbf{r}\|$$

Now

$$\frac{QP'}{NP'} = \frac{QP'}{\|\mathbf{r}\|} = \frac{QP'}{\|\mathbf{w}\|} = \sin\alpha$$

therefore

$$\overrightarrow{QP'} = \mathbf{w}\sin\alpha = (\hat{\mathbf{n}} \times \mathbf{p})\sin\alpha$$

then

$$\mathbf{p}' = \hat{\mathbf{n}}(\hat{\mathbf{n}}\bullet\mathbf{p}) + [\mathbf{p} - \hat{\mathbf{n}}(\hat{\mathbf{n}}\bullet\mathbf{p})]\cos\alpha + (\hat{\mathbf{n}} \times \mathbf{p})\sin\alpha$$

and

$$\mathbf{p}' = \mathbf{p}\cos\alpha + \hat{\mathbf{n}}(\hat{\mathbf{n}}\bullet\mathbf{p})(1 - \cos\alpha) + (\hat{\mathbf{n}} \times \mathbf{p})\sin\alpha$$

Let

$$K = 1 - \cos\alpha$$

then

$$\mathbf{p}' = \mathbf{p}\cos\alpha + \hat{\mathbf{n}}(\hat{\mathbf{n}}\bullet\mathbf{p})K + (\hat{\mathbf{n}} \times \mathbf{p})\sin\alpha$$

and

$$\begin{aligned}
\mathbf{p}' &= (x_p\mathbf{i} + y_p\mathbf{j} + z_p\mathbf{k})\cos\alpha + (a\mathbf{i} + b\mathbf{j} + c\mathbf{k})(ax_p + by_p + cz_p)K \\
&\quad + [(bz_p - cy_p)\,\mathbf{i} + (cx_p - az_p)\,\mathbf{j} + (ay_p - bx_p)\,\mathbf{k}]\sin\alpha \\
\mathbf{p}' &= [x_p\cos\alpha + a(ax_p + by_p + cz_p)K + (bz_p - cy_p)\sin\alpha]\,\mathbf{i} \\
&\quad + [y_p\cos\alpha + b(ax_p + by_p + cz_p)K + (cx_p - az_p)\sin\alpha]\,\mathbf{j} \\
&\quad + [z_p\cos\alpha + c(ax_p + by_p + cz_p)K + (ay_p - bx_p)\sin\alpha]\,\mathbf{k} \\
\mathbf{p}' &= \left[x_p\left(a^2K + \cos\alpha\right) + y_p(abK - c\sin\alpha) + z_p(acK + b\sin\alpha)\right]\mathbf{i} \\
&\quad + \left[x_p(abK + c\sin\alpha) + y_p\left(b^2K + \cos\alpha\right) + z_p(bcK - a\sin\alpha)\right]\mathbf{j} \\
&\quad + \left[x_p(acK - b\sin\alpha) + y_p(bcK + a\sin\alpha) + z_p\left(c^2K + \cos\alpha\right)\right]\mathbf{k}
\end{aligned}$$

and the transformation becomes

$$\begin{bmatrix} x'_p \\ y'_p \\ z'_p \end{bmatrix} = \begin{bmatrix} a^2K + \cos\alpha & abK - c\sin\alpha & acK + b\sin\alpha \\ abK + c\sin\alpha & b^2K + \cos\alpha & bcK - a\sin\alpha \\ acK - b\sin\alpha & bcK + a\sin\alpha & c^2K + \cos\alpha \end{bmatrix} \cdot \begin{bmatrix} x_p \\ y_p \\ z_p \end{bmatrix}$$

where $K = 1 - \cos\alpha$.

Now let's approach the problem using transforms and trigonometric identities.

Figure 7.25 shows a point $P\ (x,\ y,\ z)$ to be rotated through an angle α to $P'(x', y', z')$ about an axis defined by $\mathbf{v} = a\mathbf{i} + b\mathbf{j} + c\mathbf{k}$ where $\|\mathbf{v}\| = 1$.

The transforms to achieve this operation can be expressed as follows

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = T_5 \times T_4 \times T_3 \times T_2 \times T_1 \times \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

which aligns the axis of rotation with the x-axis, performs the rotation of $P\alpha$ about the x-axis, and returns the axis of rotation to its original position.

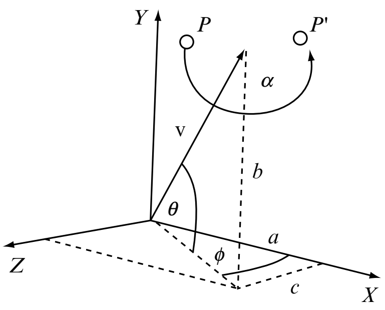


Fig. 7.25. The geometry associated with rotating a point about an arbitrary axis.

Therefore

$$\begin{aligned}
&T_1 \text{ rotates } \phi \text{ about the } y\text{-axis}\\
&T_2 \text{ rotates } -\theta \text{ about the } z\text{-axis}\\
&T_3 \text{ rotates } \alpha \text{ about the } x\text{-axis}\\
&T_4 \text{ rotates } \theta \text{ about the } z\text{-axis}\\
&T_5 \text{ rotates } -\phi \text{ about the } y\text{-axis}
\end{aligned}$$

where

$$T_1 = \begin{bmatrix} \cos\phi & 0 & \sin\phi \\ 0 & 1 & 0 \\ -\sin\phi & 0 & \cos\phi \end{bmatrix} \quad T_2 = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$T_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \quad T_4 = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$T_5 = \begin{bmatrix} \cos\phi & 0 & -\sin\phi \\ 0 & 1 & 0 \\ \sin\phi & 0 & \cos\phi \end{bmatrix}$$

Let

$$T_1 \times T_2 \times T_3 \times T_4 \times T_5 = \begin{bmatrix} E_{1,1} & E_{1,2} & E_{1,3} \\ E_{2,1} & E_{2,2} & E_{2,3} \\ E_{3,1} & E_{3,2} & E_{3,3} \end{bmatrix}$$

From Figure 7.25

$$\begin{aligned}
\cos\theta &= \sqrt{1-b^2} \Rightarrow \cos^2\theta = 1-b^2\\
\sin\theta &= b \Rightarrow \sin^2\theta = b^2\\
\cos\phi &= \frac{a}{\sqrt{1-b^2}} \Rightarrow \cos^2\phi = \frac{a^2}{1-b^2}\\
\sin\phi &= \frac{c}{\sqrt{1-b^2}} \Rightarrow \sin^2\phi = \frac{c^2}{1-b^2}
\end{aligned}$$

To find $E_{1,1}$

$$\begin{aligned}
E_{1,1} &= \cos^2\phi\cos^2\theta + \cos^2\phi\sin^2\theta\cos\alpha + \sin^2\phi\cos\alpha\\
E_{1,1} &= \frac{a^2}{1-b^2}\cdot\left(1-b^2\right) + \frac{a^2}{1-b^2}\cdot b^2\cos\alpha + \frac{c^2}{1-b^2}\cos\alpha\\
E_{1,1} &= a^2 + \frac{a^2b^2}{1-b^2}\cos\alpha + \frac{c^2}{1-b^2}\cos\alpha\\
E_{1,1} &= a^2 + \left(\frac{c^2+a^2b^2}{1-b^2}\right)\cos\alpha
\end{aligned}$$

but

$$a^2 + b^2 + c^2 = 1 \;\Rightarrow c^2 = 1 - a^2 - b^2$$

substituting c^2 in $E_{1,1}$

$$\begin{aligned}
E_{1,1} &= a^2 + \left(\frac{1 - a^2 - b^2 + a^2b^2}{1 - b^2}\right)\cos\alpha \\
E_{1,1} &= a^2 + \frac{\left(1 - a^2\right)\left(1 - b^2\right)}{1 - b^2}\cos\alpha \\
E_{1,1} &= a^2 + \left(1 - a^2\right)\cos\alpha \\
E_{1,1} &= a^2\left(1 - \cos\alpha\right) + \cos\alpha
\end{aligned}$$

Let

$$K = 1 - \cos\alpha$$

then

$$E_{1,1} = a^2K + \cos\alpha$$

To find $E_{1,2}$

$$\begin{aligned}
E_{1,2} &= \cos\phi\cos\theta\sin\theta - \cos\phi\sin\theta\cos\theta\cos\alpha - \sin\phi\cos\theta\sin\alpha \\
E_{1,2} &= \frac{a}{\sqrt{1-b^2}}\cdot\sqrt{1-b^2}\cdot b - \frac{a}{\sqrt{1-b^2}}\cdot b\cdot\sqrt{1-b^2}\cos\alpha \\
&\quad -\frac{c}{\sqrt{1-b^2}}\cdot\sqrt{1-b^2}\sin\alpha \\
E_{1,2} &= ab - ab\cos\alpha - c\sin\alpha \\
E_{1,2} &= ab\left(1 - \cos\alpha\right) - c\sin\alpha \\
E_{1,2} &= abK - c\sin\alpha
\end{aligned}$$

To find $E_{1,3}$

$$\begin{aligned}
E_{1,3} &= \cos\phi\sin\phi\cos^2\theta + \cos\phi\sin\phi\sin^2\theta\cos\alpha + \sin^2\phi\sin\theta\sin\alpha \\
&\quad + \cos^2\phi\sin\theta\sin\alpha - \cos\phi\sin\phi\cos\alpha \\
E_{1,3} &= \cos\phi\sin\phi\cos^2\theta + \cos\phi\sin\phi\sin^2\theta\cos\alpha + \sin\theta\sin\alpha\left(\sin^2\phi + \cos^2\phi\right) \\
&\quad - \cos\phi\sin\phi\cos\alpha \\
E_{1,3} &= \cos\phi\sin\phi\cos^2\theta + \cos\phi\sin\phi\sin^2\theta\cos\alpha + \sin\theta\sin\alpha - \cos\phi\sin\phi\cos\alpha \\
E_{1,3} &= \frac{a}{\sqrt{1-b^2}}\cdot\frac{c}{\sqrt{1-b^2}}\cdot\left(1 - b^2\right) + \frac{a}{\sqrt{1-b^2}}\cdot\frac{c}{\sqrt{1-b^2}}\cdot b^2\cos\alpha + b\sin\alpha \\
&\quad -\frac{a}{\sqrt{1-b^2}}\cdot\frac{c}{\sqrt{1-b^2}}\cos\alpha
\end{aligned}$$

$$E_{1,3} = ac + ac \cdot \frac{b^2}{(1-b^2)} \cos\alpha + b\sin\alpha - \frac{ac}{(1-b^2)} \cos\alpha$$
$$E_{1,3} = ac + ac \cdot \frac{\left(b^2-1\right)}{(1-b^2)} \cos\alpha + b\sin\alpha$$
$$E_{1,3} = ac\,(1-\cos\alpha) + b\sin\alpha$$
$$E_{1,3} = acK + b\sin\alpha$$

To find $E_{2,1}$

$$E_{2,1} = \sin\theta\cos\theta\cos\phi - \cos\theta\sin\theta\cos\phi\cos\alpha + \cos\theta\sin\phi\sin\alpha$$
$$E_{2,1} = b\sqrt{1-b} \cdot \frac{a}{\sqrt{1-b^2}} - \sqrt{1-b^2} \cdot b \cdot \frac{a}{\sqrt{1-b^2}} \cos\alpha + \sqrt{1-b^2} \cdot \frac{c}{\sqrt{1-b^2}} \sin\alpha$$
$$E_{2,1} = ab - ab\cos\alpha + c\sin\alpha$$
$$E_{2,1} = ab(1-\cos\alpha) + c\sin\alpha$$
$$E_{2,1} = abK + c\sin\alpha$$

To find $E_{2,2}$

$$E_{2,2} = \sin^2\theta + \cos^2\theta\cos\alpha$$
$$E_{2,2} = b^2 + \left(1-b^2\right)\cos\alpha$$
$$E_{2,2} = b^2 + \cos\alpha - b^2\cos\alpha$$
$$E_{2,2} = b^2(1-\cos\alpha) + \cos\alpha$$
$$E_{2,2} = b^2K + \cos\alpha$$

To find $E_{2,3}$

$$E_{2,3} = \sin\theta\cos\theta\sin\phi - \cos\theta\sin\theta\sin\phi\cos\alpha - \cos\theta\cos\phi\sin\alpha$$
$$E_{2,3} = b \cdot \sqrt{1-b^2} \cdot \frac{c}{\sqrt{1-b^2}} - \sqrt{1-b^2} \cdot b \cdot \frac{c}{\sqrt{1-b^2}} \cos\alpha - \sqrt{1-b^2} \cdot \frac{a}{\sqrt{1-b^2}} \sin\alpha$$
$$E_{2,3} = bc - bc\cos\alpha - a\sin\alpha$$
$$E_{2,3} = bc(1-\cos\alpha) - a\sin\alpha$$
$$E_{2,3} = bcK - a\sin\alpha$$

To find $E_{3,1}$

$$E_{3,1} = \cos\phi\sin\phi\cos^2\theta + \cos\phi\sin\phi\sin^2\theta\cos\alpha - \cos^2\phi\sin\theta\sin\alpha - \cos\phi\sin\phi\cos\alpha$$
$$E_{3,1} = \frac{a}{\sqrt{1-b^2}} \cdot \frac{c}{\sqrt{1-b^2}} \cdot \left(1-b^2\right) + \frac{a}{\sqrt{1-b^2}} \cdot \frac{c}{\sqrt{1-b^2}} \cdot b^2\cos\alpha - \sin\theta\sin\alpha\left(\cos^2\phi + \sin^2\phi\right) - \frac{a}{\sqrt{1-b^2}} \cdot \frac{c}{\sqrt{1-b^2}} \cos\alpha$$

but

$$\begin{aligned}
&\sin^2\phi + \cos^2\phi = 1\\
&\qquad E_{3,1} = ac - \frac{ac}{1-b^2}\left(1-b^2\right)\cos\alpha - b\sin\alpha\\
&\qquad E_{3,1} = ac - ac\cos\alpha - b\sin\alpha\\
&\qquad E_{3,1} = ac\left(1-\cos\alpha\right) - b\sin\alpha\\
&\qquad E_{3,1} = acK - b\sin\alpha
\end{aligned}$$

To find $E_{3,2}$

$$\begin{aligned}
E_{3,2} &= \sin\phi\cos\theta\sin\theta - \sin\phi\sin\theta\cos\theta\cos\alpha + \cos\phi\cos\theta\sin\alpha\\
E_{3,2} &= \frac{c}{\sqrt{1-b^2}}\cdot\sqrt{1-b^2}\cdot b - \frac{c}{\sqrt{1-b^2}}\cdot b\cdot\sqrt{1-b^2}\cdot\cos\alpha\\
&\quad + \frac{a}{\sqrt{1-b^2}}\cdot\sqrt{1-b^2}\cdot\sin\alpha\\
E_{3,2} &= bc - bc\cos\alpha + a\sin\alpha\\
E_{3,2} &= bc(1-\cos\alpha) + a\sin\alpha\\
E_{3,2} &= bcK + a\sin\alpha
\end{aligned}$$

To find $E_{3,3}$

$$\begin{aligned}
E_{3,3} &= \sin^2\phi\cos^2\theta + \sin^2\phi\sin^2\theta\cos\alpha - \cos\phi\sin\phi\sin\theta\sin\alpha\\
&\quad + \cos\phi\sin\phi\sin\theta\sin\alpha + \cos^2\phi\cos\alpha\\
E_{3,3} &= \frac{c^2}{1-b^2}\cdot\left(1-b^2\right) + \frac{c^2}{1-b^2}\cdot b^2\cos\alpha + \frac{a^2}{1-b^2}\cos\alpha\\
E_{3,3} &= c^2 + \frac{b^2c^2}{1-b^2}\cos\alpha + \frac{a^2}{1-b^2}\cos\alpha\\
E_{3,3} &= c^2 + \left(\frac{b^2c^2+a^2}{1-b^2}\right)\cos\alpha
\end{aligned}$$

but

$$\begin{aligned}
a^2 &= 1-b^2-c^2\\
E_{3,3} &= c^2 + \left(\frac{b^2c^2+1-b^2-c^2}{1-b^2}\right)\cos\alpha\\
E_{3,3} &= c^2 + \frac{\left(1-b^2\right)\left(1-c^2\right)}{1-b^2}\cos\alpha\\
E_{3,3} &= c^2 + \left(1-c^2\right)\cos\alpha\\
E_{3,3} &= c^2\left(1-\cos\alpha\right) + \cos\alpha\\
E_{3,3} &= c^2K + \cos\alpha
\end{aligned}$$

Therefore the transform is

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} a^2K+\cos\alpha & abK-c\sin\alpha & acK+b\sin\alpha \\ abK+c\sin\alpha & b^2K+\cos\alpha & bcK-a\sin\alpha \\ acK-b\sin\alpha & bcK+a\sin\alpha & c^2K+\cos\alpha \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

where

$$K = 1-\cos\alpha$$

Which is identical to the transformation derived from the first approach.

Now let's test the matrix with a simple example that can be easily verified. If we rotate the point P (10,5,0), 360° about an axis defined by $\mathbf{v} = \mathbf{i}+\mathbf{j}+\mathbf{k}$, it should return to itself producing P' (10,5,0).

Therefore

$$\alpha = 360^\circ \quad \cos\alpha = 1 \quad \sin\alpha = 0 \quad K = 0$$

and

$$a = 1 \quad b = 1 \quad c = 1$$

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 10 \\ 5 \\ 0 \end{bmatrix}$$

As the matrix is an identity matrix $P' = P$.

7.7.1 Quaternions

As mentioned earlier, quaternions were invented by Sir William Rowan Hamilton in the mid 19th century. Sir William was looking for a way to represent complex numbers in higher dimensions, and it took 15 years of toil before he stumbled upon the idea of using a 4D notation -hence the name 'quaternion'.

Since this discovery, mathematicians have shown that quaternions can be used to rotate points about an arbitrary axis, and hence the orientation of objects and the virtual camera. In order to develop the equation that performs this transformation we will have to understand the action of quaternions in the context of rotations.

A quaternion $\mathbf{q}$ is a quadruple of real numbers and is defined as

$$\mathbf{q} = [s, \mathbf{v}] \tag{7.84}$$

where s is a scalar and is a 3D vector. If we express the vector in terms of its components, we have in an algebraic form

$$\mathbf{q} = [s + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}] \tag{7.85}$$

where s, x, y and z are real numbers.

7.7.2 Adding and Subtracting Quaternions

Given two quaternions $\mathbf{q}_1$ and $\mathbf{q}_2$,

$$\begin{aligned}\mathbf{q}_1 &= [s_1, \mathbf{v}_1] = [s_1 + x_1\mathbf{i} + y_1\mathbf{j} + z_1\mathbf{k}]\\ \mathbf{q}_2 &= [s_2, \mathbf{v}_2] = [s_2 + x_2\mathbf{i} + y_2\mathbf{j} + z_2\mathbf{k}]\end{aligned} \tag{7.86}$$

they are equal if, and only if, their corresponding terms are equal. Furthermore, like vectors, they can be added and subtracted as follows:

$$\mathbf{q}_1 \pm \mathbf{q}_2 = [(s_1 \pm s_2) + (x_1 \pm x_2)\mathbf{i} + (y_1 \pm y_2)\mathbf{j} + (z_1 \pm z_2)\mathbf{k}] \tag{7.87}$$

7.7.3 Multiplying Quaternions

Hamilton discovered that special rules must be used when multiplying quaternions:

$$\begin{aligned}&\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1\\ &\mathbf{ij} = \mathbf{k},\ \mathbf{jk} = \mathbf{i},\ \mathbf{ki} = \mathbf{j}\\ &\mathbf{ji} = -\mathbf{k},\ \mathbf{kj} = -\mathbf{i},\ \mathbf{ik} = -\mathbf{j}\end{aligned} \tag{7.88}$$

Note that although quaternion addition is commutative, the rules make quaternion multiplication non-commutative.

Given two quaternions $\mathbf{q}_1$ and $\mathbf{q}_2$,

$$\begin{aligned}\mathbf{q}_1 &= [s_1, \mathbf{v}_1] = [s_1 + x_1\mathbf{i} + y_1\mathbf{j} + z_1\mathbf{k}]\\ \mathbf{q}_2 &= [s_2,\ \mathbf{v}_2] = [s_2 + x_2\mathbf{i} + y_2\mathbf{j} + z_2\mathbf{k}]\end{aligned} \tag{7.89}$$

the product $\mathbf{q}_1\mathbf{q}_2$, is given by

$$\begin{aligned}\mathbf{q}_1\mathbf{q}_2 = [&(s_1s_2 - x_1x_2 - y_1y_2 - z_1z_2) + (s_1x_2 + s_2x_1 + y_1z_2 - y_2z_1)\mathbf{i}\\ &+(s_1y_2 + s_2y_1 + z_1x_2 - z_2x_1)\mathbf{j} + (s_1z_2 + s_2z_1 + x_1y_2 - x_2y_1)\mathbf{k}\end{aligned} \tag{7.90}$$

which can be rewritten using the dot and cross product notation as

$$\mathbf{q}_1\mathbf{q}_2 = [(s_1s_2 - \mathbf{v}_1 \cdot \mathbf{v}_2),\ s_1\mathbf{v}_2 + s_2\mathbf{v}_1 + \mathbf{v}_1 \times \mathbf{v}_2] \tag{7.91}$$

7.7.4 The Inverse Quaternion

Given the quaternion

$$\mathbf{q} = [s + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}] \tag{7.92}$$

the inverse quaternion $\mathbf{q}^{-1}$ is

$$\mathbf{q}^{-1} = \frac{[s - x\mathbf{i} - y\mathbf{j} - z\mathbf{k}]}{|\mathbf{q}|^2} \tag{7.93}$$

where $|\mathbf{q}|$ is the magnitude, or modulus, of $\mathbf{q}$, and is equal to

$$\|\mathbf{q}\| = \sqrt{s^2 + x^2 + y^2 + z^2} \tag{7.94}$$

It can also be shown that

$$\mathbf{q}\mathbf{q}^{-1} = \mathbf{q}^{-1}\mathbf{q} = 1 \tag{7.95}$$

7.7.5 Rotating Points about an Axis

Basically, quaternions are associated with vectors rather than individual points. Therefore, in order to manipulate a single vertex, we must first turn it into a position vector, which has its tail vertex at the origin. A vertex can then be represented in quaternion form by its equivalent position vector and a zero scalar term. For example, a point $P(x,\ y,\ z)$ is represented in quaternion form by

$$\mathbf{p} = [0 + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}] \tag{7.96}$$

which can then be transformed into another position vector using the process described below. The coordinates of the rotated point are the components of the rotated position vector. This may seem an indirect process, but in reality it turns out to be rather simple. Let's now consider how this is achieved.

It can be shown that a position vector $\mathbf{p}$ can be rotated about an axis by some angle using the following operation:

$$\mathbf{p}' = \mathbf{q}\mathbf{p}\mathbf{q}^{-1} \tag{7.97}$$

where the axis and angle of rotation are encoded within the unit quaternion $\mathbf{q}$, whose modulus is 1, and $\mathbf{p}'$ is the rotated vector. For example, to rotate a point $P(x,\ y,\ z)$ through an angle θ about an axis, we use the following steps:

1 Convert the point $P(x,\ y,\ z)$ to a quaternion $\mathbf{p}$:

$$\mathbf{p} = [0 + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}]$$

2 Define the axis of rotation as a unit vector $\mathbf{u}$:

$$\mathbf{u} = [x_u\mathbf{i} + y_u\mathbf{j} + z_u\mathbf{k}]$$

3 *Define the transforming quaternion* $\mathbf{q}$:

$$\mathbf{q} = [\cos(\theta/2),\ \sin(\theta/2)\mathbf{u}]$$

4 Define the inverse of the transforming quaternion $\mathbf{q}^{-1}$:

$$\mathbf{q}^{-1} = [\cos(\theta/2),\ -\sin(\theta/2)\mathbf{u}]$$

5 Compute $\mathbf{p}'$:

$$\mathbf{p}' = \mathbf{q}\mathbf{p}\mathbf{q}^{-1}$$

6 Unpack (x', y', z')

$$(x', y', z') \quad \mathbf{p}' = [0 + x'\mathbf{i} + y'\mathbf{j} + z'\mathbf{k}]$$

We can verify the action of the above transform with a simple example. Consider the point $P(0,\ 1,\ 1)$ in Figure 7.26 which is to be rotated 90° about the y-axis. We can see that the rotated point P' has the coordinates (1, 1, 0), which we will confirm algebraically. The point P is represented by a quaternion $\mathbf{P}$, and is rotated by evaluating the quaternion $\mathbf{P}'$:

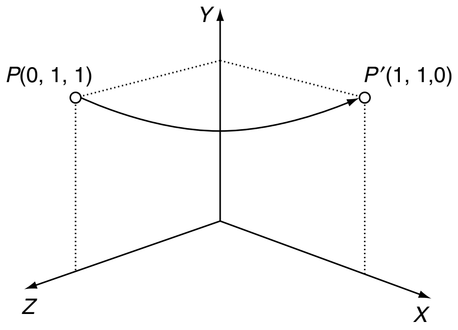


Fig. 7.26. The point $P(0,1,1)$ is rotated to $P'(1,1,0)$ using a quaternion coincident with the y-axis.

$$\mathbf{P}' = \mathbf{qPq}^{-1}$$

which will store the rotated coordinates. The axis of rotation is $[\mathbf{j}]$, therefore the unit quaternion $\mathbf{q}$ is given by

$$\begin{aligned}\mathbf{q} &= [\cos(90°/2),\ \sin(90°/2)[0,\ 1,\ 0]] \\ &= [\cos(45°),\ [0,\ \sin(45°),\ 0]]\end{aligned}$$

The inverse quaternion $\mathbf{q}^{-1}$ is given by

$$\mathbf{q}^{-1} = \frac{[\cos(90°/2), -\sin(90°/2)[0,\ 1,\ 0]]}{|\mathbf{q}|^2}$$

but as $\mathbf{q}$ is a unit quaternion, the denominator $|\mathbf{q}|^2$ equals unity and can be ignored. Therefore

$$\mathbf{q}^{-1} = [\cos(45°),\ [0, -\sin(45°)0]]$$

Let's evaluate $\mathbf{qPq}^{-1}$ in two stages: $(\mathbf{qP})\mathbf{q}^{-1}$.

1

$$\begin{aligned}\mathbf{qP} &= [\cos(45°),\ [0,\ \sin(45°),\ 0]\,] \cdot [0,\ [0,\ 1,\ 1]] \\ &= [-\sin(45°),\ [\sin(45°),\ \cos(45°),\ \cos(45°)]]\end{aligned}$$

2

$$\begin{aligned}(\mathbf{qP})\mathbf{q}^{-1} &= [-\sin(45^\circ),\ [\sin(45^\circ),\ \cos(45^\circ),\ \cos(45^\circ)]\,] \\ &\quad \cdot[\cos(45^\circ),\ [0,\ -\sin(45^\circ),\ 0]\,] \\ &= [0,\ [2\sin(45^\circ)\cos(45^\circ), 1, \cos(45^\circ)\cos(45^\circ) - \sin(45^\circ)\sin(45^\circ]\,] \\ P' &= [0, [1,\ 1,\ 0]]\end{aligned}$$

and the vector component of P' confirms that P is indeed rotated to (1, 1, 0).

We will evaluate one more example before continuing. Consider a rotation about the z-axis as illustrated in Figure 7.27. The original point has coordinates (0, 1, 1) and is rotated $-90°$. From the figure we see that this should finish at (1, 0, 1). This time the quaternion $\mathbf{q}$ is defined by

$$\begin{aligned}\mathbf{q} &= [\cos(-90^\circ/2),\ \sin(-90^\circ/2)[0,\ 0,\ 1]] \\ &= [\cos(45^\circ),\ [0, 0, -\sin(45^\circ)]]\end{aligned}$$

with its inverse

$$\mathbf{q}^{-1} = [\cos(45^\circ),\ [0,\ 0,\ \sin(45^\circ]]$$

and the point to be rotated in quaternion form is $\mathbf{P} = [0, [0, 1, 1]]$. Evaluating this in two stages we have

1

$$\begin{aligned}\mathbf{qP} &= [\cos(45^\circ),\ [0,\ 0, -\sin(45^\circ)]] \cdot [0,\ [0,\ 1,\ 1]] \\ &= [\sin(45^\circ), [\sin(45^\circ),\ \cos(45^\circ),\ \cos(45^\circ)]]\end{aligned}$$

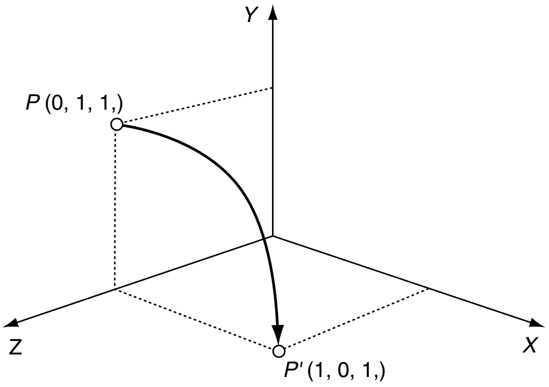


Fig. 7.27. The point $P(0, 1, 1)$ is rotated $-90°$ to $P'(1, 0, 1)$ using a quaternion coincident with the z-axis.

2

$$\begin{aligned}(\mathbf{pP})\mathbf{q}^{-1} &= [\sin(45^\circ),\ [\sin(45^\circ),\ \cos(45^\circ),\ \cos(45^\circ)]]\\ &\quad\cdot[\cos(45^\circ),\ [0, 0, \sin(45^\circ)]]\\ &= [0,\ [\sin(90^\circ),\ \cos(90^\circ),\ 1]]\end{aligned}$$

The vector component of P' confirms that P is rotated to (1, 0, 1).

7.7.6 Roll, Pitch and Yaw Quaternions

Having already looked at roll, pitch and yaw rotations, we can now define them as quaternions:

$$\begin{aligned}\mathbf{q}_{roll} &= [\cos(\theta/2),\ \sin(\theta/2)[0,\ 0,\ 1]]\\ \mathbf{q}_{pitch} &= [\cos(\theta/2),\ \sin(\theta/2)[1,\ 0,\ 0]]\\ \mathbf{q}_{yaw} &= [\cos(\theta/2),\ \sin(\theta/2)[0,\ 1,\ 0]]\end{aligned} \tag{7.98}$$

where θ is the angle of rotation.

These quaternions can be multiplied together to create a single quaternion representing a compound rotation. For example, if the quaternions are defined as

$$\begin{aligned}\mathbf{q}_{roll} &= [\cos(roll/2),\ \sin(roll/2)[0,\ 0,\ 1]]\\ \mathbf{q}_{pitch} &= [\cos(pitch/2),\ \sin(pitch/2)[1,\ 0,\ 0]]\\ \mathbf{q}_{yaw} &= [\cos(yaw/2),\ \sin(yaw/2)[0,\ 1,\ 0]]\end{aligned} \tag{7.99}$$

they can be concatenated to a single quaternion $\mathbf{q}$:

$$\mathbf{q} = \mathbf{q}_{yaw}\mathbf{q}_{pitch}\mathbf{q}_{roll} = [s + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}] \tag{7.100}$$

where

$$\begin{aligned}s &= \cos\left(\frac{yaw}{2}\right)\cos\left(\frac{pitch}{2}\right)\cos\left(\frac{roll}{2}\right) + \sin\left(\frac{yaw}{2}\right)\sin\left(\frac{pitch}{2}\right)\sin\left(\frac{roll}{2}\right)\\ x &= \cos\left(\frac{yaw}{2}\right)\sin\left(\frac{pitch}{2}\right)\cos\left(\frac{roll}{2}\right) + \sin\left(\frac{yaw}{2}\right)\cos\left(\frac{pitch}{2}\right)\sin\left(\frac{roll}{2}\right)\\ y &= \sin\left(\frac{yaw}{2}\right)\cos\left(\frac{pitch}{2}\right)\cos\left(\frac{roll}{2}\right) - \cos\left(\frac{yaw}{2}\right)\sin\left(\frac{pitch}{2}\right)\sin\left(\frac{roll}{2}\right)\\ z &= \cos\left(\frac{yaw}{2}\right)\cos\left(\frac{pitch}{2}\right)\sin\left(\frac{roll}{2}\right) - \sin\left(\frac{yaw}{2}\right)\sin\left(\frac{pitch}{2}\right)\cos\left(\frac{roll}{2}\right)\end{aligned} \tag{7.101}$$

Let's examine this compound quaternion with an example. For instance, given the following conditions let's derive a single quaternion **q** to represent the compound rotation:

$$\begin{aligned} roll &= 90^\circ \\ pitch &= 180^\circ \\ yaw &= 0^\circ \end{aligned}$$

The values of s, x, y, z are

$$\begin{aligned} s &= 0 \\ x &= \cos(45^\circ) \\ y &= -\sin(45^\circ) \\ z &= 0 \end{aligned}$$

and the quaternion **q** is

$$\mathbf{q} = [0,\ [\cos(45^\circ), -\sin(45^\circ),\ 0]]$$

If the point $P(1,\ 1,\ 1)$ is subjected to this compound rotation, the rotated point is computed using the standard quaternion transform:

$$\mathbf{P}' = \mathbf{qPq}^{-1}$$

Let's evaluate $\mathbf{qPq}^{-1}$ in two stages:

1

$$\begin{aligned} \mathbf{qP} &= [0,\ [\cos(45^\circ), -\sin(45^\circ),\ 0]\,] \cdot [0,\ [1,\ 1,\ 1]] \\ &= [0,\ [-\sin(45^\circ), -\cos(45^\circ),\ \sin(45^\circ) + \cos(45^\circ)]] \end{aligned}$$

2

$$\begin{aligned} (\mathbf{qP})\mathbf{q}^{-1} &= [0,\ [-\sin(45^\circ), -\cos(45^\circ),\ \sin(45^\circ) + \cos(45^\circ)]] \\ &\quad \cdot [0,\ [-\cos(45^\circ),\ \sin(45^\circ),\ 0]] \\ \mathrm{P}' &= [0,\ [-1,\ -1,\ -1]] \end{aligned}$$

Therefore, the coordinates of the rotated point are $(-1,\ -1,\ -1)$, which can be confirmed from Figure 7.28.

7.7.7 Quaternions in Matrix Form

There is a direct relationship between quaternions and matrices. For example, given the quaternion $[s + x\mathbf{i} + y\mathbf{j} + z\mathbf{k})$ the equivalent matrix is

$$\begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix}$$

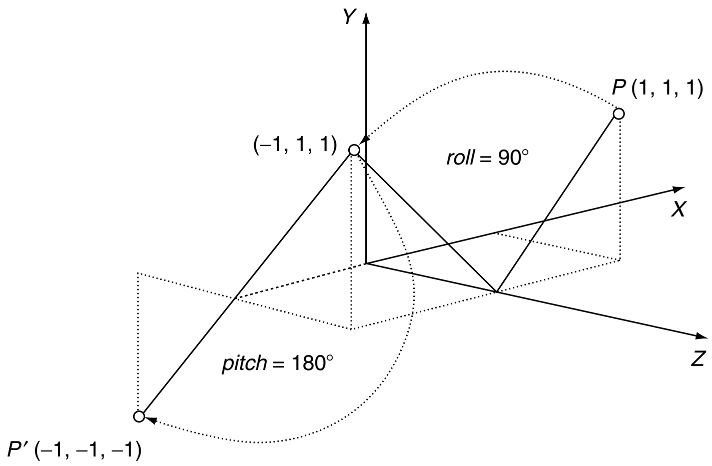


Fig. 7.28. The point P is subject to a compound roll of 90° and a pitch of 180°. This diagram shows the transform in two stages.

where

$$\begin{aligned}
M_{11} &= 1 - 2(y^2 + z^2) \\
M_{12} &= 2(xy - sz) \\
M_{13} &= 2(xz + sy) \\
M_{21} &= 2(xy + sz) \\
M_{22} &= 1 - 2(x^2 + z^2) \\
M_{23} &= 2(yz - sx) \\
M_{31} &= 2(xz - sy) \\
M_{32} &= 2(yz + sx) \\
M_{33} &= 1 - 2(x^2 + y^2)
\end{aligned} \tag{7.102}$$

Substituting the following values of s, x, y, z:

$$\begin{aligned}
s &= 0 \\
x &= \cos(45^\circ) \\
y &= -\sin(45^\circ) \\
z &= 0
\end{aligned}$$

the matrix transformation is

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

Substituting (1, 1, 1) for (x, y, z) the rotated point becomes $(-1, -1, -1)$, as shown in Figure 7.28.

7.7.8 Frames of Reference

A quaternion, or its equivalent matrix, can be used to rotate a vertex or position a virtual camera. If unit quaternions are used, the associated matrix is orthogonal, which means that its transpose is equivalent to rotating the frame of reference in the opposite direction. For example, if the virtual camera is oriented with a yaw rotation of 180°, i.e. looking along the negative z-axis, the orientation quaternion is [0, [0, 1, 0]]. Therefore $s = 0, x = 0, y = 1, z = 0$. The equivalent matrix is

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

which is equal to its transpose. Therefore, a vertex (x, y, z) in world space has coordinates (x', y', z') in camera space and the transform is defined by

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

If the vertex (x, y, z) is $(1, 1, 0)$, (x', y', z') becomes $(-1, 1, 0)$, which is correct. However, it is unlikely that the virtual camera will only be subjected to a simple rotation, as it will normally be translated from the origin. Consequently, a translation matrix will have to be introduced as described above.

7.8 Transforming Vectors

The transforms described in this chapter have been used to transform single points. However, a geometric database will contain not only pure vertices, but also vectors, which must also be subject to any prevailing transform. A generic transform Q of a 3D point can be represented by

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = [Q] \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{7.103}$$

and as a vector is defined by two points we can write

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = [Q] \cdot \begin{bmatrix} x_2 - x_1 \\ y_2 - y_1 \\ z_2 - z_1 \\ 1 - 1 \end{bmatrix} \tag{7.104}$$

where we see the homogeneous scaling term collapse to zero. This implies that any vector $[x \quad y \quad z]^{\mathrm{T}}$ can be transformed using

$$\begin{bmatrix} x' \\ y' \\ z' \\ 0 \end{bmatrix} = [Q] \cdot \begin{bmatrix} x \\ y \\ z \\ 0 \end{bmatrix} \tag{7.105}$$

Let's put this to the test by using a transform from an earlier example. The problem concerned a change of axial system where a virtual camera was subject to the following:

$$\begin{aligned} \mathit{roll} &= 180^\circ \\ \mathit{pitch} &= 90^\circ \\ \mathit{yaw} &= 90^\circ \\ t_x &= 2 \\ t_y &= 2 \\ t_z &= 0 \end{aligned}$$

and the transform is

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 & 2 \\ 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

When the point (1, 1, 0) is transformed it becomes (1, 0, 1), as shown in Figure 7.29. But if we transform the vector $\begin{bmatrix}1 \; 1 \; 0\end{bmatrix}^{\mathrm{T}}$ it becomes $\begin{bmatrix}-1 \; 0 \; -1\end{bmatrix}^{\mathrm{T}}$, using the following transform

$$\begin{bmatrix} -1 \\ 0 \\ -1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 & 2 \\ 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

which is correct with reference to Figure 7.29.

7.9 Determinants

Before concluding this chapter I would like to expand upon the role of the determinant in transforms. Normally, determinants arise in the solution of

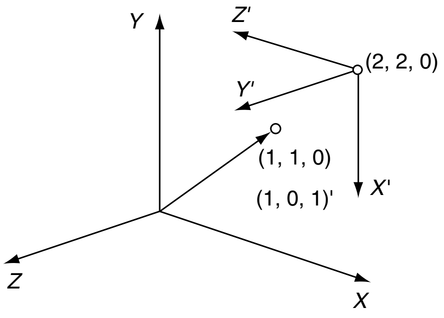


Fig. 7.29. Vector $[1\,1\,0]^T$ is transformed to $[-1\ 0\ -1]^T$.

linear equations such as

$$\begin{aligned} c_1 &= a_1x + b_1y \\ c_2 &= a_2x + b_2y \end{aligned} \tag{7.106}$$

where values of x and y are defined in terms of the other constants. Without showing the solution, the values of x and y are given by

$$x = \frac{c_1b_2 - c_2b_1}{a_1b_2 - a_2b_1}$$

$$y = \frac{a_1c_2 - a_2c_1}{a_1b_2 - a_2b_1} \tag{7.107}$$

provided that the denominator $a_1b_2 - a_2b_1 \neq 0$.

It is also possible to write the linear equations in matrix form as

$$\begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} \tag{7.108}$$

and we notice that the denominator comes from the matrix terms $a_1b_2 - a_2b_1$. This is called the *determinant*, and is valid only for square matrices. A determinant is defined as follows:

$$\begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix} = a_1b_2 - a_2b_1 \tag{7.109}$$

With this notation it is possible to rewrite the original linear equations as

$$\frac{x}{\begin{vmatrix} c_1 & b_1 \\ c_2 & b_2 \end{vmatrix}} = \frac{y}{\begin{vmatrix} a_1 & c_1 \\ a_2 & c_2 \end{vmatrix}} = \frac{1}{\begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix}} \tag{7.110}$$

With a set of three linear equations:

$$\begin{aligned} d_1 &= a_1x + b_1y + c_1z \\ d_2 &= a_2x + b_2y + c_2z \\ d_3 &= a_3x + b_3y + c_3z \end{aligned} \tag{7.111}$$

the value of x is defined as

$$x = \frac{d_1b_2c_3 - d_1b_3c_2 + d_2b_3c_1 - d_2b_1c_3 + d_3b_1c_2 - d_3b_2c_1}{a_1b_2c_3 - a_1b_3c_2 + a_2b_3c_1 - a_2b_1c_3 + a_3b_1c_2 - a_3b_2c_1} \tag{7.112}$$

with similar expressions for y and z. Once more, the denominator comes from the determinant of the matrix associated with the matrix formulation of the linear equations:

$$\begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{7.113}$$

where

$$\begin{vmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{vmatrix} = a_1b_2c_3 - a_1b_3c_2 + a_2b_3c_1 - a_2b_1c_3 + a_3b_1c_2 - a_3b_2c_1$$

which can be written as

$$a_1 \begin{vmatrix} b_2 & c_2 \\ b_3 & c_3 \end{vmatrix} - a_2 \begin{vmatrix} b_1 & c_1 \\ b_3 & c_3 \end{vmatrix} + a_3 \begin{vmatrix} b_1 & c_1 \\ b_2 & c_2 \end{vmatrix} \tag{7.114}$$

Let's now see what creates a zero determinant. If we write, for example

$$10 = 2x + y \tag{7.115}$$

there are an infinite number of solutions for x and y, and it is impossible to solve the equation. However, if we introduce a second equation relating x and y:

$$4 = 5x - y \tag{7.116}$$

we can solve for x and y using (7.107):

$$\begin{aligned} x &= \frac{10 \times (-1) - 4 \times 1}{2 \times (-1) - 5 \times 1} = \frac{-14}{-7} = 2 \\ y &= \frac{2 \times 4 - 5 \times 10}{2 \times (-1) - 5 \times 1} = \frac{-42}{-7} = 6 \end{aligned} \tag{7.117}$$

therefore $x = 2$ and $y = 6$, which is correct.

But say the second equation had been

$$20 = 4x + 2y \tag{7.118}$$

which would have created the pair of simultaneous equations

$$10 = 2x + y \tag{7.119}$$

$$20 = 4x + 2y \tag{7.120}$$

If we now solve for x and y we get

$$\begin{aligned} x &= \frac{10 \times 2 - 20 \times 1}{2 \times 2 - 4 \times 1} = \frac{0}{0} = \text{undefined} \\ y &= \frac{2 \times 20 - 4 \times 10}{2 \times 2 - 4 \times 1} = \frac{0}{0} = \text{undefined} \end{aligned}$$

which yields undefined results. The reason for this is that (7.119) is the same as (7.120) – the second equation is nothing more than twice the first equation, and therefore brings nothing new to the relationship. When this occurs, the equations are said to be *linearly dependent.*

Having shown the algebraic origins of the determinant, we can now go on to investigate its graphical significance. Consider the transform

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} \tag{7.121}$$

The determinant of the transform is $ad - cb$. If we subject the vertices of a unit-square to this transform, we create the situation shown in Figure 7.30. The vertices of the unit-square are moved as follows:

$$\begin{array}{ll} (0,0) & (0,0) \\ (1,0) & (a,c) \\ (1,1) & (a+b, c+d) \\ (0,1) & (b,d) \end{array} \tag{7.122}$$

From Figure 7.30 it can be seen that the area of the transformed unit-square A' is given by

$$\begin{aligned} area &= (a+b)(c+d) - \text{B} - \text{C} - \text{D} - \text{E} - \text{F} - \text{G} \\ &= ac + ad + cb + bd - \frac{1}{2}bd - cb - \frac{1}{2}ac - \frac{1}{2}bd - cb - \frac{1}{2}ac \\ &= ad - cb \end{aligned} \tag{7.123}$$

which is the determinant of the transform. But as the area of the original unit-square was 1, the determinant of the transform controls the scaling factor applied to the transformed shape.

Let's examine the determinants of two transforms. The first 2D transform encodes a scaling of 2, and results in an overall area scaling of 4:

$$\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$

and the determinant is

$$\begin{vmatrix} 2 & 0 \\ 0 & 2 \end{vmatrix} = 4$$

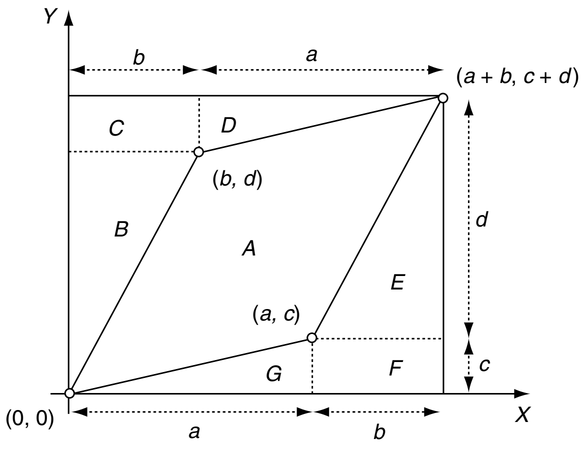


Fig. 7.30. The inner parallelogram is the transformed unit square.

The second 2D transform encodes a scaling of 3 and a translation of $(3, 3)$, and results in an overall area scaling of 9:

$$\begin{bmatrix} 3 & 0 & 3 \\ 0 & 3 & 3 \\ 0 & 0 & 1 \end{bmatrix}$$

and the determinant is

$$3\begin{vmatrix} 3 & 3 \\ 0 & 1 \end{vmatrix} - 0\begin{vmatrix} 0 & 3 \\ 0 & 1 \end{vmatrix} + 0\begin{vmatrix} 0 & 3 \\ 3 & 3 \end{vmatrix} = 9$$

These two examples demonstrate the extra role played by the elements of a matrix.

7.10 Perspective Projection

Of all the projections employed in computer graphics, the *perspective projection* is the one most widely used. There are two stages to its computation: the first stage involves converting world coordinates to the camera's frame of reference, and the second stage transforms camera coordinates to the projection plane coordinates. We have already looked at the transforms for locating a camera in world space, and the inverse transform for converting world coordinates to the camera's frame of reference. Let's now investigate how these camera coordinates are transformed into a perspective projection.

We begin by assuming that the camera is directed along the z-axis as shown in Figure 7.31. Positioned d units along the axis is a projection screen,

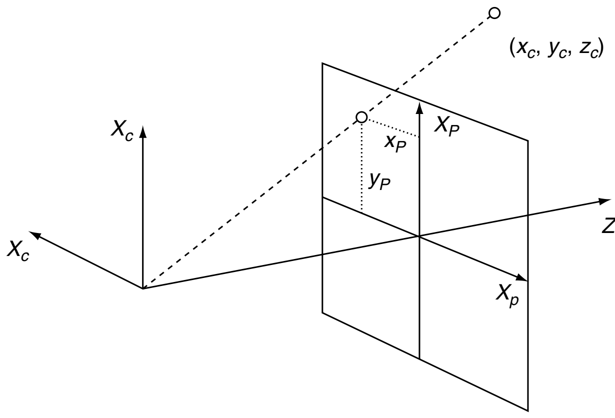


Fig. 7.31. The axial systems used to produce a perspective projection.

which will be used to capture a perspective projection of an object. Figure 7.31 shows that any point $(x_\mathrm{c}, y_\mathrm{c}, z_\mathrm{c})$ becomes transformed to $(x_\mathrm{s}, y_\mathrm{s}, d)$. It also shows that the screen's x-axis is pointing in the opposite direction to the camera's x-axis, which can be compensated for by reversing the sign of x_s when it is computed.

Figure 7.32 shows plan and side views of the scenario depicted in Figure 7.31, which enables us to inspect the geometry and make the following observations:

$$\begin{aligned} \frac{x}{z} &= \frac{-x_p}{d} \quad x_p = -d\frac{x}{z} \quad x_p = \frac{-y}{z/d} \\ \frac{y}{z} &= \frac{y_p}{d} \quad y_p = d\frac{y}{z} \quad y_p = \frac{y}{z/d} \end{aligned} \tag{7.124}$$

This can be expressed in matrix as

$$\begin{bmatrix} x_s \\ y_s \\ z_s \\ W \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

At first this may seem strange, but if we multiply it out we get

$$[x_p \ y_p \ z_p \ W]^T = [-x \ y \ z \ z/d]^T$$

and if we remember the idea behind homogeneous coordinates, we must divide the terms $x_\mathrm{p}, y_\mathrm{p}, z_\mathrm{p}$ by W to get the scaled terms, which produces

$$x_p = \frac{-x}{z/d}, \quad y_p = \frac{y}{z/d}, \quad z_p = \frac{z}{z/d} = d$$

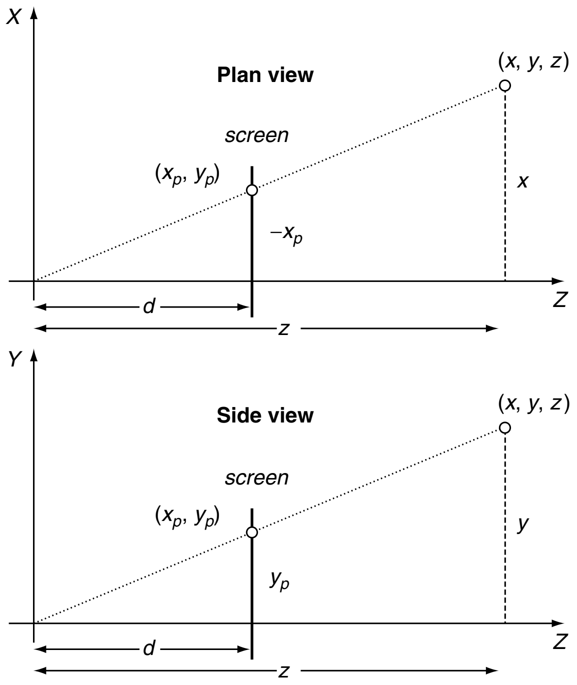


Fig. 7.32. The plan and side views for computing the perspective projection transform.

which, after all, is rather elegant. Notice that this transform takes into account the sign change that coours with the x-coordinate. Some books will leave this sign reversal until the mapping is made to screen coordinates

7.11 Summary

The purpose of this chapter was to introduce transforms and matrices – I hope this has been achieved. This end of the chapter is not really the end of the subject, as one can do so much with matrices and quaternions. For example, it would be interesting to see how a matrix behaves when some of its elements are changed dynamically, and what happens when we interpolate between a pair of quaternions. Such topics are addressed in later chapters.