With the procedures for displaying output primitives and their attributes, we can create a variety of pictures and graphs. In many applications, there is also a need for altering or manipulating displays. Design applications and facility layouts are created by arranging the orientations and sizes of the component parts of the scene. And animations are produced by moving the "camera" or the objects in a scene along animation paths. Changes in orientation, size, and shape are accomplished with **geometric transformations** that alter the coordinate descriptions of objects. The basic geometric transformations are translation, rotation, and scaling. Other transformations that are often applied to objects include reflection and shear. We first discuss methods for performing geometric transformations and then consider how transformation functions can be incorporated into graphics packages.

5-1 BASIC TRANSFORMATIONS

Here, we first discuss general procedures for applying translation, rotation, and scaling parameters to reposition and resize two-dimensional objects. Then, in Section 5-2, we consider how transformation equations can be expressed in a more convenient matrix formulation that allows efficient combination of object transformations.

Translation

A **translation** is applied to an object by repositioning it along a straight-line path from one coordinate location to another. We translate a two-dimensional point by adding **translation distances**, t_x and t_y, to the original coordinate position (x, y) to move the point to a new position (x', y') (Fig. 5-1).

$$x' = x + t_x, \qquad y' = y + t_y \tag{5-1}$$

The translation distance pair (t_x, t_y) is called a **translation vector** or **shift vector**.

We can express the translation equations 5-1 as a single matrix equation by using column vectors to represent coordinate positions and the translation vector:

$$\mathbf{P} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \qquad \mathbf{P}' = \begin{bmatrix} x'_1 \\ x'_2 \end{bmatrix}, \qquad \mathbf{T} = \begin{bmatrix} t_x \\ t_y \end{bmatrix} \tag{5-2}$$

This allows us to write the two-dimensional translation equations in the matrix form:

$$\mathbf{P}' = \mathbf{P} + \mathbf{T} \tag{5-3}$$

Sometimes matrix-transformation equations are expressed in terms of coordinate row vectors instead of column vectors. In this case, we would write the matrix representations as $\mathbf{P} = [x \ y]$ and $\mathbf{T} = [t_x \ t_y]$. Since the column-vector representation for a point is standard mathematical notation, and since many graphics packages, for example, GKS and PHIGS, also use the column-vector representation, we will follow this convention.

Translation is a *rigid-body transformation* that moves objects without deformation. That is, every point on the object is translated by the same amount. A straight line segment is translated by applying the transformation equation 5-3 to each of the line endpoints and redrawing the line between the new endpoint positions. Polygons are translated by adding the translation vector to the coordinate position of each vertex and regenerating the polygon using the new set of vertex coordinates and the current attribute settings. Figure 5-2 illustrates the application of a specified translation vector to move an object from one position to another.

Similar methods are used to translate curved objects. To change the position of a circle or ellipse, we translate the center coordinates and redraw the figure in the new location. We translate other curves (for example, splines) by displacing the coordinate positions defining the objects, then we reconstruct the curve paths using the translated coordinate points.

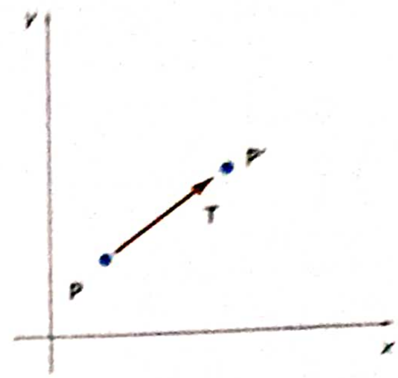


Figure 5-1
Translating a point from position **P** to position **P'** with translation vector **T**.

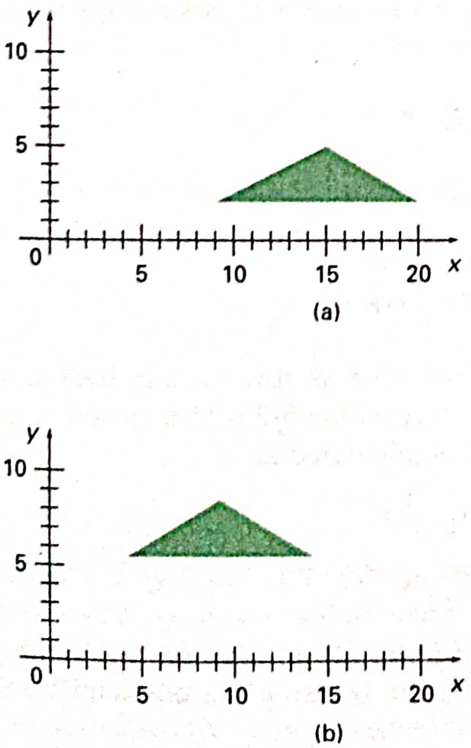


Figure 5-2
Moving a polygon from position (a) to position (b) with the translation vector (-5.50, 3.75).

Rotation

A two-dimensional **rotation** is applied to an object by repositioning it along a circular path in the xy plane. To generate a rotation, we specify a **rotation angle** θ and the position (x_r, y_r) of the **rotation point (or pivot point)** about which the object is to be rotated (Fig. 5-3). Positive values for the rotation angle define counterclockwise rotations about the pivot point, as in Fig. 5-3, and negative values rotate objects in the clockwise direction. This transformation can also be described as a rotation about a **rotation axis** that is perpendicular to the xy plane and passes through the pivot point.

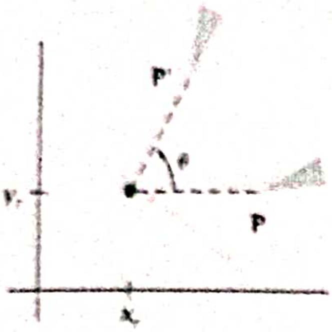
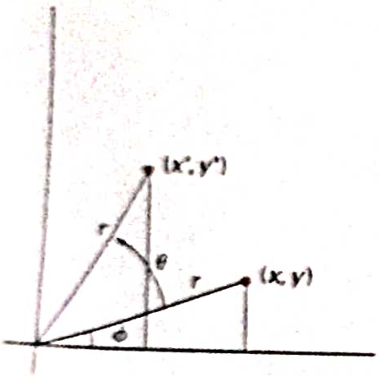


Figure 5-3
Rotation of an object through angle θ about the pivot point (x_r, y_r).

We first determine the transformation equations for rotation of a point position **P** when the pivot point is at the coordinate origin. The angular and coordinate relationships of the original and transformed point positions are shown in Fig. 5-4. In this figure, r is the constant distance of the point from the origin, angle ϕ is the original angular position of the point from the horizontal, and θ is the rotation angle. Using standard trigonometric identities, we can express the transformed coordinates in terms of angles θ and ϕ as

$$
\begin{aligned}
x' &= r\cos(\phi+\theta) = r\cos\phi\cos\theta - r\sin\phi\sin\theta \\
y' &= r\sin(\phi+\theta) = r\cos\phi\sin\theta + r\sin\phi\cos\theta
\end{aligned}
\tag{5-4}
$$



point from
$y)$ to position
rough an angle θ
e to the coordinate
gin. The original angular displacement of the point from the x axis is ϕ.

The original coordinates of the point in polar coordinates are

$$x = r\cos\phi, \qquad y = r\sin\phi \tag{5-5}$$

Substituting expressions 5-5 into 5-4, we obtain the transformation equations for rotating a point at position (x, y) through an angle θ about the origin:

$$
\begin{aligned}
x' &= x\cos\theta - y\sin\theta \\
y' &= x\sin\theta + y\cos\theta
\end{aligned}
\tag{5-6}
$$

With the column-vector representations 5-2 for coordinate positions, we can write the rotation equations in the matrix form:

$$\mathbf{P}' = \mathbf{R}\cdot\mathbf{P} \tag{5-7}$$

where the rotation matrix is

$$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{5-8}$$

When coordinate positions are represented as row vectors instead of column vectors, the matrix product in rotation equation 5-7 is transposed so that the transformed row coordinate vector $[x' \; y']$ is calculated as

$$
\begin{aligned}
\mathbf{P}'^T &= (\mathbf{R}\cdot\mathbf{P})^T \\
&= \mathbf{P}^T\cdot\mathbf{R}^T
\end{aligned}
$$

where $\mathbf{P}^T = [x \; y]$, and the transpose $\mathbf{R}^T$ of matrix $\mathbf{R}$ is obtained by interchanging rows and columns. For a rotation matrix, the transpose is obtained by simply changing the sign of the sine terms.

Rotation of a point about an arbitrary pivot position is illustrated in Fig. 5 Using the trigonometric relationships in this figure, we can generalize Eqs. 5-6 obtain the transformation equations for rotation of a point about any specified r tation position (x_r, y_r):

$$x' = x_r + (x - x_r)\cos\theta - (y - y_r)\sin\theta$$

$$y' = y_r + (x - x_r)\sin\theta + (y - y_r)\cos\theta \qquad (5\text{-}$$

These general rotation equations differ from Eqs. 5-6 by the inclusion of additiv terms, as well as the multiplicative factors on the coordinate values. Thus, th matrix expression 5-7 could be modified to include pivot coordinates by matri addition of a column vector whose elements contain the additive (translationa terms in Eqs. 5-9. There are better ways, however, to formulate such matrix equa tions, and we discuss in Section 5-2 a more consistent scheme for representing th transformation equations.

As with translations, rotations are rigid-body transformations that move objects without deformation. Every point on an object is rotated through the same angle. A straight line segment is rotated by applying the rotation equations 5-9 to each of the line endpoints and redrawing the line between the new endpoint positions. Polygons are rotated by displacing each vertex through the specified rotation angle and regenerating the polygon using the new vertices. Curved lines are rotated by repositioning the defining points and redrawing the curves. A circle or an ellipse, for instance, can be rotated about a noncentral axis by moving the center position through the arc that subtends the specified rotation angle. An ellipse can be rotated about its center coordinates by rotating the major and minor axes.

Scaling

A scaling transformation alters the size of an object. This operation can be carried out for polygons by multiplying the coordinate values (x, y) of each vertex by scaling factors s_x and s_y to produce the transformed coordinates (x', y'):

$$x' = x \cdot s_x, \qquad y' = y \cdot s_y \qquad (5\text{-}10)$$

Scaling factor s_x scales objects in the x direction, while s_y scales in the y direction. The transformation equations 5-10 can also be written in the matrix form:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} \qquad (5\text{-}11)$$

or

$$\mathbf{P}' = \mathbf{S} \cdot \mathbf{P} \qquad (5\text{-}12)$$

where **S** is the 2 by 2 scaling matrix in Eq. 5-11.

Any positive numeric values can be assigned to the scaling factors s_x and s_y. Values less than 1 reduce the size of objects; values greater than 1 produce an enlargement. Specifying a value of 1 for both s_x and s_y leaves the size of objects unchanged. When s_x and s_y are assigned the same value, a **uniform scaling** is pro-

(a)

e (a) into a
h scaling
d $s_y = 1$.

duced that maintains relative object proportions. Unequal values for s_x and s_y result in a **differential scaling** that is often used in design applications, where pictures are constructed from a few basic shapes that can be adjusted by scaling and positioning transformations (Fig. 5-6).

Objects transformed with Eq. 5-11 are both scaled and repositioned. Scaling factors with values less than 1 move objects closer to the coordinate origin, while values greater than 1 move coordinate positions farther from the origin. Figure 5-7 illustrates scaling a line by assigning the value 0.5 to both s_x and s_y in Eq. 5-11. Both the line length and the distance from the origin are reduced by a factor of 1/2.

We can control the location of a scaled object by choosing a position, called the **fixed point**, that is to remain unchanged after the scaling transformation. Coordinates for the fixed point (x_f, y_f) can be chosen as one of the vertices, the object centroid, or any other position (Fig. 5-8). A polygon is then scaled relative to the fixed point by scaling the distance from each vertex to the fixed point. For a vertex with coordinates (x, y), the scaled coordinates (x', y') are calculated as

$$x' = x_f + (x - x_f)s_x, \qquad y' = y_f + (y - y_f)s_y \tag{5-13}$$

We can rewrite these scaling transformations to separate the multiplicative and additive terms:

$$\begin{aligned} x' &= x \cdot s_x + x_f(1 - s_x) \\ y' &= y \cdot s_y + y_f(1 - s_y) \end{aligned} \tag{5-14}$$

where the additive terms $x_f(1 - s_x)$ and $y_f(1 - s_y)$ are constant for all points in the object.

Including coordinates for a fixed point in the scaling equations is similar to including coordinates for a pivot point in the rotation equations. We can set up a column vector whose elements are the constant terms in Eqs. 5-14, then we add this column vector to the product **S** · **P** in Eq. 5-12. In the next section, we discuss a matrix formulation for the transformation equations that involves only matrix multiplication.

q. 5-12
s reduced
loser to
n.

Polygons are scaled by applying transformations 5-14 to each vertex and then regenerating the polygon using the transformed vertices. Other objects are scaled by applying the scaling transformation equations to the parameters defining the objects. An ellipse in standard position is resized by scaling the semimajor and semiminor axes and redrawing the ellipse about the designated center coordinates. Uniform scaling of a circle is done by simply adjusting the radius. Then we redisplay the circle about the center coordinates using the transformed radius.

5-2 MATRIX REPRESENTATIONS AND HOMOGENEOUS COORDINATES

Many graphics applications involve sequences of geometric transformations. An animation, for example, might require an object ... each increment ...