Lecture 4: Affine Transformations

for Satan himself is transformed into an angel of light. 2 Corinthians 11:14

1. Transformations

Transformations are the lifeblood of geometry. Euclidean geometry is based on *rigid motions* -- translation and rotation -- transformations that preserve distances and angles. *Congruent triangles* are triangles where corresponding lengths and angles match.

Transformations generate geometry. The turtle uses translation (FORWARD), rotation (TURN), and uniform scaling (RESIZE) to generate curves by moving about on the plane. We can also apply translation (SHIFT), rotation (SPIN), and uniform scaling (SCALE) to build new shapes from previously defined turtle programs. These three transformations -- translation, rotation and uniform scaling -- are called *conformal transformations*. Conformal transformations preserve angles, but not distances. *Similar triangles* are triangles where corresponding angles agree, but the lengths of corresponding sides are scaled. The ability to scale is what allows the turtle to generate self-similar fractals like the Sierpinski gasket.

In Computer Graphics transformations are employed to position, orient, and scale objects as well as to model shape. Much of elementary Computational Geometry and Computer Graphics is based upon an understanding of the effects of different fundamental transformations.

The transformations that appear most often in 2-dimensional Computer Graphics are the affine transformations. *Affine transformations* are composites of four basic types of transformations: translation, rotation, scaling (uniform and non-uniform), and shear. Affine transformations do not necessarily preserve either distances or angles, but affine transformations map straight lines to straight lines and affine transformations preserve ratios of distances along straight lines (see Figure 1). For example, affine transformations map midpoints to midpoints. In this lecture we are going to develop explicit formulas for various affine transformations; in the next lecture we will use these affine transformations as an alternative to turtle programs to model shapes for Computer Graphics.

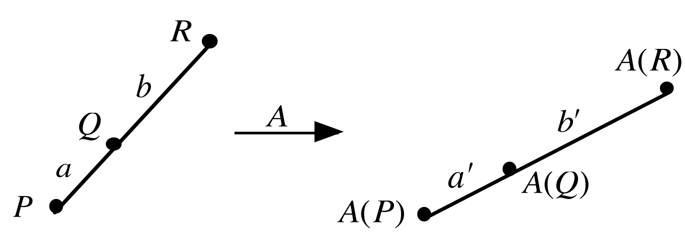

Figure 1: An affine transformations A maps lines to lines and preserve ratios of distances along straight lines. Thus $A(Q)$ lies on the line joining $A(P)$ and $A(R)$, and $a'/b' = a/b$.

A word of warning before we begin. There is a good deal of linear algebra in this chapter. Don't panic. Though there is a lot of algebra, the details are all fairly straightforward. The bulk of this chapter is devoted to deriving matrix representations for the affine transformations which will be used in subsequent chapters to generate shapes for 2-dimensional Computer Graphics. All the matrices that you will need later on are listed at the end of this chapter, so you will not have to slog through these derivations every time you require a specific matrix; you can find these matrices easily whenever you need them. Persevere now because there will be a big payoff shortly.

2. Conformal Transformations

Among the most important affine transformations are the conformal transformations: translation, rotation, and uniform scaling. We shall begin our study of affine transformations by developing explicit matrix representations for these conformal transformations.

To fix our notation, we will use upper case letters $P,Q,R,\ldots$ from the middle of the alphabet to denote points and lower case letters $u,v,w,\ldots$ from the end of the alphabet to denote vectors. Lower case letters with subscripts will denote rectangular coordinates. Thus, for example, we shall write $P=(p_1,p_2)$ to denote that (p_1,p_2) are the rectangular coordinates of the point P. Similarly, we shall write $v=(v_1,v_2)$ to denote that (v_1,v_2) are the rectangular coordinates of the vector v.

2.1 Translation. We have already encountered translation in our study of Turtle Graphics (see Figure 2). To translate a point $P=(p_1,p_2)$ by a vector $w=(w_1,w_2)$, we set

$$P^{new} = P + w \tag{1}$$

or in terms of coordinates

$$p_1^{new} = p_1 + w_1$$

$$p_2^{new} = p_2 + w_2 .$$

Vectors are unaffected by translation, so

$$v^{new} = v .$$

We can rewrite these equations in matrix form. Let I denote the 2×2 identity matrix; then

$$P^{new} = P * I + w$$

$$v^{new} = v * I.$$

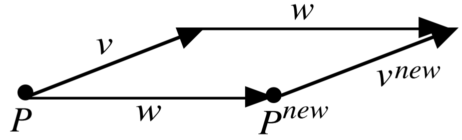

Figure 2: Translation. Points are affected by translation; vectors are not affected by translation.

2.2 Rotation. We also encountered rotation in Turtle Graphics. Recall from Lecture 1 that to rotate a vector $v = (v_1, v_2)$ through an angle θ, we introduce the orthogonal vector $v^{\perp} = (-v_2, v_1)$ and set

$$v^{new} = \cos(\theta)v + \sin(\theta)v^{\perp}$$

or equivalently in terms of coordinates

$$\begin{aligned} v_1^{new} &= v_1\cos(\theta) - v_2\sin(\theta) \\ v_2^{new} &= v_1\sin(\theta) + v_2\cos(\theta). \end{aligned} \tag{2}$$

Introducing the rotation matrix

$$Rot(\theta) = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix},$$

we can rewrite Equation (2) in matrix form as

$$v^{new} = v * Rot(\theta). \tag{3}$$

In our study of affine geometry in the plane, we want to rotate not only vectors about their tails, but also points about other points. Since $P = Q + (P - Q)$, rotating a point P about another point Q through the angle θ is equivalent to rotating the vector $P - Q$ through the angle θ and then adding the resulting vector to Q (see Figure 3). Thus by Equation (3) the formula for rotating a point P about another point Q through the angle θ is

$$P^{new} = Q + (P - Q) * Rot(\theta) = P * Rot(\theta) + Q * \big(I - Rot(\theta)\big), \tag{4}$$

where I is again the 2×2 identity matrix. Notice that if Q is the origin, then this formula reduces to

$$P^{new} = P * Rot(\theta),$$

so $Rot(\theta)$ is also the matrix representing rotation of points about the origin.

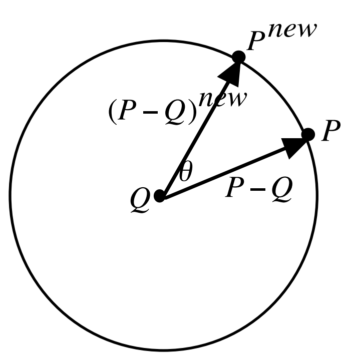

Figure 3: Rotation about the point Q. Since the point Q is fixed and since $P = Q + (P - Q)$, rotating a point P about the point Q through the angle θ is equivalent to rotating the vector $P - Q$ through the angle θ and then adding the resulting vector to Q.

2.3 Uniform Scaling. Uniform scaling also appears in Turtle Graphics. To scale a vector $v = (v_1, v_2)$ by a factor s, we simply set

$$v^{new} = sv \tag{5}$$

or in terms of coordinates

$$v_1^{new} = sv_1$$
$$v_2^{new} = sv_2 .$$

Introducing the scaling matrix

$$Scale(s) = \begin{pmatrix} s & 0 \\ 0 & s \end{pmatrix},$$

we can rewrite Equation (5) in matrix form as

$$v^{new} = v * Scale(s) .$$

Again, in affine geometry we want not only to scale vectors, but also to scale uniformly the distance of points from a fixed point. Since $P = Q + (P - Q)$, scaling the distance of an arbitrary point P from a fixed point Q by the factor s is equivalent to scaling the length of the vector $P - Q$ by the factor s and then adding the resulting vector to Q (see Figure 4). Thus the formula for scaling the distance of an arbitrary point P from a fixed point Q by the factor s is

$$P^{new} = Q + (P - Q) * Scale(s) = P * Scale(s) + Q * \left(I - Scale(s)\right). \tag{6}$$

Notice that if Q is the origin, then this formula reduces to

$$P^{new} = P * Scale(s) ,$$

so $Scale(s)$ is also the matrix that represents uniformly scaling the distance of points from the origin.

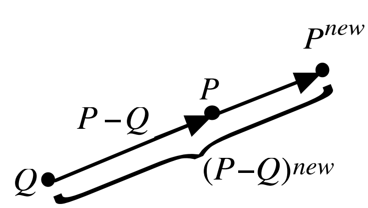

Figure 4: Uniform scaling about a fixed point Q. Since the point Q is fixed and since $P = Q + (P - Q)$, scaling the distance of a point P from the point Q by the factor s is equivalent to scaling the vector $P - Q$ by the factor s and then adding the resulting vector to Q.

3. The Algebra of Affine Transformations

The three conformal transformations -- translation, rotation, and uniform scaling -- all have the following form: there exists a matrix M and a vector w such that

$$\begin{aligned} v^{new} &= v * M \\ P^{new} &= P * M + w. \end{aligned} \tag{7}$$

In fact, this form characterizes all affine transformations. That is, a transformation is said to be *affine* if and only if there is a matrix M and a vector w so that Equation (7) is satisfied.

The matrix M represents a linear transformation on vectors. Recall that a transformation L on vectors is *linear* if

$$\begin{aligned} L(u+v) &= L(u) + L(v) \\ L(cv) &= cL(v). \end{aligned} \tag{8}$$

Matrix multiplication represents a linear transformation because matrix multiplication distributes through vector addition and commutes with scalar multiplication -- that is,

$$\begin{aligned} (u+v) * M &= u * M + v * M \\ (cv) * M &= c(v * M). \end{aligned}$$

The vector w in Equation (7) represents translation on the points. Thus an affine transformation can always be decomposed into a linear transformation followed by a translation. Notice that translation is not a linear transformation, since translation does not satisfy Equation (8) on points; therefore translation cannot be represented by a 2×2 matrix.

Affine transformations can also be characterized abstractly in a manner similar to linear transformations. A transformation A is said to be *affine* if A maps points to points, A maps vectors to vectors, and

$$\begin{aligned} A(u+v) &= A(u) + A(v) \\ A(cv) &= c\,A(v) \\ A(P+v) &= A(P) + A(v). \end{aligned} \tag{9}$$

The first two equalities in Equation (9) say that an affine transformation is a linear transformation on vectors; the third equality asserts that affine transformations are well behaved with respect to the addition of points and vectors. You should check that with this definition, translation is indeed an affine transformation.

In terms of coordinates, linear transformations can be written as

$$\begin{aligned} x^{new} &= ax + by \\ y^{new} &= cx + dy \end{aligned} \quad \Leftrightarrow \quad (x^{new}, y^{new}) = (x, y) * \begin{pmatrix} a & c \\ b & d \end{pmatrix} \tag{10}$$

whereas affine transformations have the form

$$\begin{aligned} x^{new} &= a\,x + b\,y + e \\ y^{new} &= c\,x + d\,y + f \end{aligned} \quad \Leftrightarrow \quad (x^{new}, y^{new}) = (x, y) * \begin{pmatrix} a & c \\ b & d \end{pmatrix} + (e, f). \qquad (11)$$

The constant terms e and f that appear in Equation (11) are what distinguish the affine transformations of Computer Graphics from the linear transformations of classical linear algebra. These constants represent translation, which, as we have seen, is not a linear transformation.

4. The Geometry of Affine Transformations

There is also a geometric way to characterize both linear and affine transformations. Interpreted geometrically Equation (8) says that linear transformations map triangles into triangles and lines into lines (see Figure 5). Moreover, linear transformations preserve ratios of distances along lines, since

$$\frac{|L(c\,v)|}{|L(v)|} = \frac{|c\,L(v)|}{|L(v)|} = \frac{|c|\,|L(v)|}{|L(v)|} = |c| = \frac{|c|\,|v|}{|v|} = \frac{|c\,v|}{|v|}\,.$$

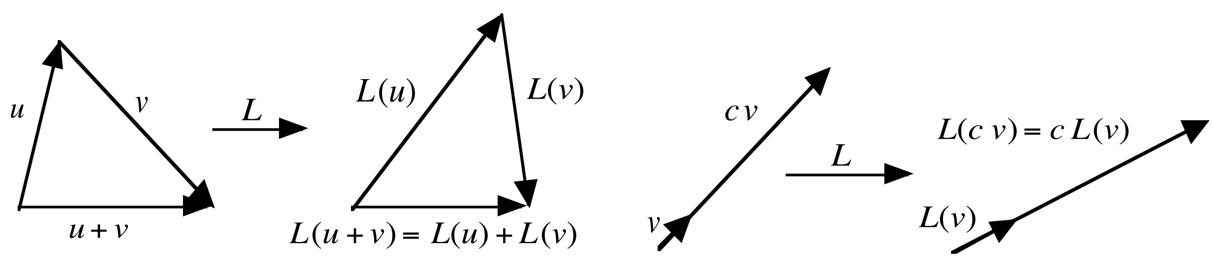

Figure 5: Linear transformations map triangles into triangles (left) and lines into lines (right).

Similarly, interpreted geometrically Equation (9) make precisely the same assertions about affine transformations, extending these results to points as well as to vectors (see Figure 6).

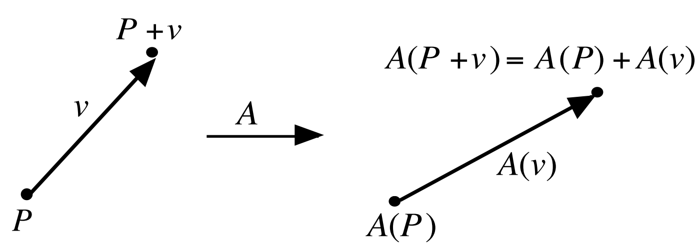

Figure 6: Affine transformations map triangles into triangles and lines into lines (Figure 9), and these results are consistent for points as well as for vectors.

5. Affine Coordinates and Affine Matrices

Linear transformations are typically represented by matrices because composing two linear transformations is equivalent to multiplying the corresponding matrices. We would like to have the same facility with affine transformations -- that is, we would like to be able to compose two affine transformations by multiplying their matrix representations. Unfortunately, our current representation of an affine transformation in terms of a transformation matrix M and a translation vector w does not work so well when we want to compose two affine transformations. In order to overcome this difficulty, we shall now introduce a clever device called *affine coordinates*.

Points and vectors are both represented by pairs of rectangular coordinates, but points and vectors are different types of objects with different behaviors for the same affine transformations. For example, points are affected by translation, but vectors are not. We are now going to introduce a third coordinate -- an affine coordinate -- to distinguish between points and vectors. Affine coordinates will also allow us to represent each affine transformation using a single 3×3 matrix and to compose any two affine transformations by matrix multiplication.

Since points are affected by translation and vectors are not, the affine coordinate for a point is 1; the affine coordinate for a vector is 0. Thus, from now on, we shall write $P=(p_1,p_2,1)$ for points and $v=(v_1,v_2,0)$ for vectors.

Affine transformations can now be represented by 3×3 affine matrices. An *affine matrix* is a matrix where the third column is always $(0 \quad 0 \quad 1)^T$. The affine transformation

$$v^{new}=v*M$$

$$P^{new}=P*M+w$$

represented by the 2×2 transformation matrix M and the translation vector w can be rewritten using affine coordinates in terms of a single 3×3 affine matrix

$$\tilde{M}=\begin{pmatrix} M & 0 \\ w & 1 \end{pmatrix}.$$

Now

$$(v^{new},0)=(v,0)*\tilde{M}=(v,0)*\begin{pmatrix} M & 0 \\ w & 1 \end{pmatrix}=(v*M,0)$$

$$(P^{new},1)=(P,1)*\tilde{M}=(P,1)*\begin{pmatrix} M & 0 \\ w & 1 \end{pmatrix}=(P*M+w,1).$$

Notice how the affine coordinate -- 0 for vectors and 1 for points -- is correctly reproduced by multiplication with the last column of the matrix $\tilde{M}$. Notice too that the 0 in the third coordinate for vectors effectively insures that vectors ignore the translation represented by the vector w.

6. Conformal Transformations -- Revisited

To illustrate some specific examples of affine matrices, below we show how to represent the standard conformal transformations -- translation, rotation, and uniform scaling -- in terms of 3×3 affine matrices. These results follow easily from Equations (1), (4), and (6).

Translation -- by the vector $w = (w_1, w_2)$

$$Trans(w) = \begin{pmatrix} I & 0 \\ w & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ w_1 & w_2 & 1 \end{pmatrix}$$

Rotation -- around the Origin through the angle θ

$$Rot(Origin, \theta) = \begin{pmatrix} Rot(\theta) & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Rotation -- around the point $Q = (q_1, q_2)$ *through the angle* θ

$$Rot(Q, \theta) = \begin{pmatrix} Rot(\theta) & 0 \\ Q * \left(I - Rot(\theta)\right) & 1 \end{pmatrix} = \begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ q_1\left(1 - \cos(\theta)\right) + q_2 \sin(\theta) & -q_1 \sin(\theta) + q_2\left(1 - \cos(\theta)\right) & 1 \end{pmatrix}$$

Uniform Scaling -- around the Origin by the factor s

$$Scale(Origin, s) = \begin{pmatrix} sI & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Uniform Scaling -- around the point $Q = (q_1, q_2)$ *by the factor s*

$$Scale(Q, s) = \begin{pmatrix} sI & 0 \\ Q * (1 - s)I & 1 \end{pmatrix} = \begin{pmatrix} s & 0 & 0 \\ 0 & s & 0 \\ (1 - s)q_1 & (1 - s)q_2 & 1 \end{pmatrix}$$

7. General Affine Transformations

We shall now develop 3×3 matrix representations for arbitrary affine transformations. The two most general ways of specifying an affine transformation of the plane are by specifying either the image of one point and two linearly independent vectors or by specifying the image of three non-collinear points. We shall treat each of these cases in turn.

7.1 Image of One Point and Two Vectors. Fix a point Q and two linearly independent vectors u,v. Since two linearly independent vectors u,v form a basis for vectors in the plane, for any vector w there are scalars σ,τ such that

$$w = \sigma u + \tau v .$$

Hence for any point P, there are scalars s,t such that

$$P - Q = s\,u + t\,v$$

or equivalently

$$P = Q + s\,u + t\,v$$

Therefore if A is an affine transformation, then by Equation (9)

$$\begin{aligned} A(P) &= A(Q) + s\,A(u) + t\,A(v) \\ A(w) &= \sigma\,A(u) + \tau\,A(v) . \end{aligned} \qquad (12)$$

Thus if we know the effect of the affine transformation A on one point Q and on two linearly independent vectors u,v, then we know the effect of A on every point P and every vector w. Hence, in addition to conformal transformations, there is another important way to define affine transformations of the plane: by specifying the image of one point and two linearly independent vectors.

Geometrically, Equation (12) says that an affine transformation A maps the parallelogram determined by the point Q and the vectors u,v into the parallelogram determined by the point $A(Q)$ and the vectors $A(u),A(v)$ (see Figure 7). Thus affine transformations map parallelograms into parallelograms.

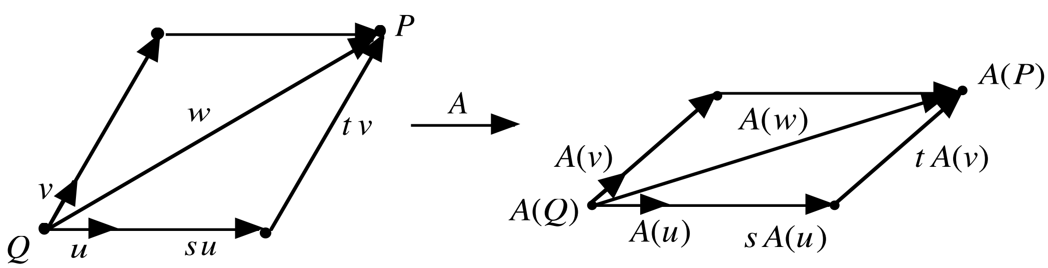

Figure 7: Affine transformations map parallelograms to parallelograms.

To find the 3×3 matrix representation M for such an affine transformation A, let Q^*,u^*,v^* be the images of Q,u,v under the transformation A. Then using affine coordinates, we have

$$(u^*,0) = (u,0) * M$$

$$(v^*,0) = (v,0) * M$$

$$(Q^*,1) = (Q,1) * M$$

where

$$M = \begin{pmatrix} a & c & 0 \\ b & d & 0 \\ e & f & 1 \end{pmatrix}.$$

Thus

$$\underbrace{\begin{pmatrix} u^* & 0 \\ v^* & 0 \\ Q^* & 1 \end{pmatrix}}_{S_{new}} = \underbrace{\begin{pmatrix} u & 0 \\ v & 0 \\ Q & 1 \end{pmatrix}}_{S_{old}} * \underbrace{\begin{pmatrix} a & c & 0 \\ b & d & 0 \\ e & f & 1 \end{pmatrix}}_{M}.$$

Solving for M yields

$$M = S_{old}^{-1} * S_{new}$$

or equivalently

$$M = \begin{pmatrix} u & 0 \\ v & 0 \\ Q & 1 \end{pmatrix}^{-1} * \begin{pmatrix} u^* & 0 \\ v^* & 0 \\ Q^* & 1 \end{pmatrix} = \begin{pmatrix} u_1 & u_2 & 0 \\ v_1 & v_2 & 0 \\ q_1 & q_2 & 1 \end{pmatrix}^{-1} * \begin{pmatrix} u_1^* & u_2^* & 0 \\ v_1^* & v_2^* & 0 \\ q_1^* & q_2^* & 1 \end{pmatrix}. \qquad (13)$$

To compute M explicitly, we need only invert a 3×3 affine matrix. But for a nonsingular 3×3 affine matrix, the inverse can easily be written explicitly; for the formula, see Exercise 13a. An alternative explicit formula for the matrix M without inverses is provided in Exercise 21.

7.2 Non-Uniform Scaling. One application of our preceding approach to general affine transformations is a formula for non-uniform scaling. Uniform scaling scales distances by the same amount in all directions; non-uniform scaling scales distances by different amounts in different directions. We are interested in non-uniform scaling for many reasons. For example, we can apply non-uniform scaling to generate an ellipse by scaling a circle from its center along a fixed direction (see Figure 8). To scale the distance from a fixed point Q along an arbitrary direction w by a scale factor s, we shall now apply our method for generating arbitrary affine transformations by specifying the image of one point and two linearly independent vectors (see Figure 9).

Figure 8: Scaling a circle from its center along a fixed direction generates an ellipse.

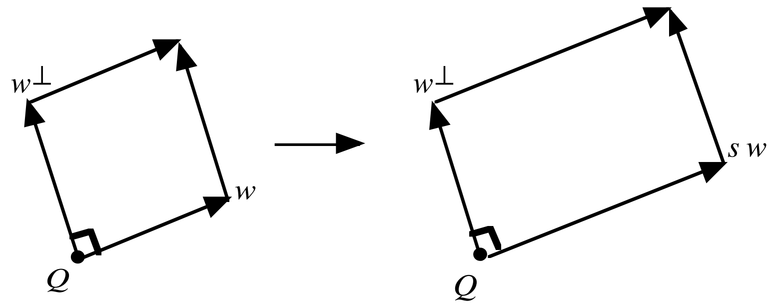

Figure 9: Scaling from the fixed point Q in the direction w by the scale factor s. The point Q is fixed, the vector w is scaled by s, and the orthogonal vector $w^\perp$ remains fixed. Thus non-uniform scaling maps a square into a rectangle.

Let $Scale(Q,w,s)$ denote the matrix that represents the affine transformation which scales the distance from a fixed point Q along an arbitrary direction w by a scale factor s. To find the 3×3 matrix $Scale(Q,w,s)$, we need to know the image of one point and two linearly independent vectors. Consider the point Q and the vectors w and $w^\perp$, where $w^\perp$ is a vector of the same length as w perpendicular to w. It is easy to see how the transformation $Scale(Q,w,s)$ affects Q, w, $w^\perp$:

- $Q \to Q$ because Q is a fixed point of the transformation.
- $w \to s\,w$ because distances are scaled by the factor s along the direction w.
- $w^\perp \to w^\perp$ because distances along the vector $w^\perp$ are not changed, since $w^\perp$ is orthogonal to the scaling direction w.

Therefore by the results of the previous section:

$$Scale(Q,w,s) = \begin{pmatrix} w & 0 \\ w^\perp & 0 \\ Q & 1 \end{pmatrix}^{-1} * \begin{pmatrix} s\,w & 0 \\ w^\perp & 0 \\ Q & 1 \end{pmatrix}.$$

Now recall that if $w = (w_1, w_2)$, then $w^\perp = (-w_2, w_1)$. Therefore in terms of coordinates

$$Scale(Q,w,s) = \begin{pmatrix} w_1 & w_2 & 0 \\ -w_2 & w_1 & 0 \\ q_1 & q_2 & 1 \end{pmatrix}^{-1} * \begin{pmatrix} s\,w_1 & s\,w_2 & 0 \\ -w_2 & w_1 & 0 \\ q_1 & q_2 & 1 \end{pmatrix}.$$

An explicit formula for the matrix $Scale(Q,w,s)$ without inverses is provided in Exercise 20.

If Q is the origin and w is the unit vector along the x-axis, then $w^\perp$ is the unit vector along the y-axis, so

$$\begin{pmatrix} w_1 & w_2 & 0 \\ -w_2 & w_1 & 0 \\ q_1 & q_2 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \mathit{Identity\ Matrix}\,.$$

Therefore scaling along the x-axis is represented by the 3×3 matrix

$$Scale(Origin, i, s) = \begin{pmatrix} s & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Similarly, scaling along the y-axis is represented by the 3×3 matrix

$$Scale(Origin, j, s) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

7.3 Image of Three Non-Collinear Points. If we know the image of three non-collinear points under an affine transformation, then we also know the image one point and two linearly independent vectors. Indeed, suppose that we know the image of the three non-collinear points P_1, P_2, P_3 under the affine transformation A. Then $u = P_2 - P_1$ and $v = P_3 - P_1$ are certainly linearly independent vectors, and

$$A(u) = A(P_2 - P_1) = A(P_2) - A(P_1)$$
$$A(v) = A(P_3 - P_1) = A(P_3) - A(P_1)\ .$$

Hence if we know the image of P_1, P_2, P_3 under the affine transformation A, then we know the image of the point P_1 and the two linearly independent vectors u, v. Therefore the image of three non-collinear points specifies a unique affine transformation.

To find the 3×3 matrix M that represents an affine transformation A defined by the image of three non-collinear points, we can proceed as in Section 7.1. Let P_1^*, P_2^*, P_3^* be the images of the points P_1, P_2, P_3 under the transformation A. Using affine coordinates, we have

$$(P_1^*, 1) = (P_1, 1) * M$$
$$(P_2^*, 1) = (P_2, 1) * M$$
$$(P_3^*, 1) = (P_3, 1) * M$$

where

$$M = \begin{pmatrix} a & c & 0 \\ b & d & 0 \\ e & f & 1 \end{pmatrix}.$$

Thus

$$\underbrace{\begin{pmatrix} P_1^* & 1 \\ P_2^* & 1 \\ P_3^* & 1 \end{pmatrix}}_{P_{new}} = \underbrace{\begin{pmatrix} P_1 & 1 \\ P_2 & 1 \\ P_3 & 1 \end{pmatrix}}_{P_{old}} * \underbrace{\begin{pmatrix} a & c & 0 \\ b & d & 0 \\ e & f & 1 \end{pmatrix}}_{M}.$$

Solving for M yields

$$M = P_{old}^{-1} * P_{new}$$

or equivalently if $P_k = (x_k, y_k)$ and $P_k^* = (x_k^*, y_k^*)$ for $k = 1,2,3$, then

$$M = \begin{pmatrix} P_1 & 1 \\ P_2 & 1 \\ P_3 & 1 \end{pmatrix}^{-1} * \begin{pmatrix} P_1^* & 1 \\ P_2^* & 1 \\ P_3^* & 1 \end{pmatrix} = \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix}^{-1} * \begin{pmatrix} x_1^* & y_1^* & 1 \\ x_2^* & y_2^* & 1 \\ x_3^* & y_3^* & 1 \end{pmatrix}. \qquad (14)$$

To compute M explicitly, we need only invert a 3×3 matrix whose last column is $(1\ 1\ 1)^T$. This inverse can easily be written explicitly; for the formula, see Exercise 13b. An alternative explicit formula for the matrix M without inverses is provided in Exercise 21.

Notice the similarity between the right hand side of Equation (14) and the right hand side of Equation (13) in Section 7.1 for the matrix representing the affine transformation defined by the image of one point and two linearly independent vectors. The only differences are the ones appearing in the first two rows of the third columns, indicating that two vectors have been replaced by two points.

8. Summary

Below is a brief summary of the main high level concepts that you need to remember from this lecture. Also listed below for your convenience are the 3×3 matrix representations for all the affine transformations that we have discussed in this lecture.

8.1 Affine Transformations and Affine Coordinates. Affine transformations are the fundamental transformations of 2-dimensional Computer Graphics. There are several different ways of characterizing affine transformations:

1. A transformation A is affine if there exists a 2×2 matrix M and a vector w such that
   $A(v) = v * M$
   $A(P) = P * M + w$.
2. A transformation A is affine if there is a 3×3 affine matrix M such that
   $\big(A(v), 0\big) = (v, 0) * M$
   $\big(A(P), 1\big) = (P, 1) * M$

3. A transformation A is affine if
   $A(u+v) = A(u) + A(v)$
   $A(cv) = c\,A(v)$
   $A(P+v) = A(P) + A(v)$.

4. A transformation A is affine if A maps triangles to triangles and lines to lines, and A preserves ratios of distances along lines.

Affine transformations include the standard conformal transformations -- translation, rotation, and uniform scaling -- of Turtle Graphics, but affine transformations also incorporate other transformations such as non-uniform scaling. Every affine transformations can be uniquely specified either by the image of one point and two linearly independent vectors or by the image of three non-collinear points.

Affine coordinates are used to distinguish between points and vectors. The affine coordinate of a point is 1; the affine coordinate of a vector is 0. Affine coordinates can be applied to represent all affine transformations by 3×3 matrices of the form,

$$\tilde{M} = \begin{pmatrix} M & 0 \\ w & 1 \end{pmatrix},$$

so that

$$(v^{new}, 0) = (v, 0) * M$$

$$(P^{new}, 1) = (P, 1) * M.$$

These matrix representations allow us to compose affine transformations using matrix multiplication.

The affine transformations of Computer Graphics are closely related to the linear transformations of standard linear algebra. In fact, in terms of matrix representations

$$\mathit{Affine\ Transformation} = \begin{pmatrix} \mathit{Linear} & 0 \\ \mathit{Transformation} & 0 \\ \mathit{Translation} & 1 \end{pmatrix}.$$

In Table 1 we provide a more detailed comparison of the differences between affine and linear transformations.

Affine Transformation	*Linear Transformation*
$A(u+v) = A(u) + A(v)$ $A(cv) = c\,A(v)$ $A(P+v) = A(P) + A(v)$	$L(u+v) = L(u) + L(v)$ $L(cv) = c\,L(v)$
Includes Translation	Excludes Translation
Affine Coordinates	Rectangular Coordinates
Coordinate Formulas $x_{new} = ax_{old} + by_{old} + e$ $y_{new} = cx_{old} + dy_{old} + f$	Coordinate Formulas $x_{new} = ax_{old} + by_{old}$ $y_{new} = cx_{old} + dy_{old}$
3×3 Matrix Representation $(x_{new}, y_{new}, 1) = (x_{old}, y_{old}, 1) * \begin{pmatrix} a & c & 0 \\ b & d & 0 \\ e & f & 1 \end{pmatrix}$	2×2 Matrix Representation $(x_{new}, y_{new}) = (x_{old}, y_{old}) * \begin{pmatrix} a & c \\ b & d \end{pmatrix}$

Table 1: Comparison of affine and linear transformations. For both affine and linear transformations, composition of transformations is equivalent to matrix multiplication.

Finally, a word of caution before we conclude. We use matrices to *represent* transformations, but matrices are not the same things as transformations. Matrix representations for affine transformations are akin to coordinate representations for points and vectors -- that is, a low level computational tool for communicating efficiently with a computer. *Matrices are the assembly language of transformations*. As usual, we do not want to think or write programs in an assembly language; we want to work in a high level language. In the next lecture, we shall introduce a high level language for 2-dimensional Computer Graphics based on affine transformations. You will code the matrices for these transformations only once, but you will use the corresponding high level transformations many, many times.

8.2 Matrices for Affine Transformations in the Plane

Translation -- by the vector $w = (w_1, w_2)$

$$Trans(w) = \begin{pmatrix} I & 0 \\ w & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ w_1 & w_2 & 1 \end{pmatrix}$$

Rotation -- around the Origin through the angle θ

$$Rot(Origin,\theta) = \begin{pmatrix} Rot(\theta) & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Rotation -- around the point $Q = (q_1, q_2)$ *through the angle* θ

$$Rot(Q,\theta) = \begin{pmatrix} Rot(\theta) & 0 \\ Q * \left(I - Rot(\theta)\right) & 1 \end{pmatrix} = \begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ q_1\left(1-\cos(\theta)\right) + q_2 \sin(\theta) & -q_1 \sin(\theta) + q_2\left(1-\cos(\theta)\right) & 1 \end{pmatrix}$$

Uniform Scaling -- around the Origin by the factor s

$$Scale(Origin, s) = \begin{pmatrix} sI & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Uniform Scaling -- around the point $Q = (q_1, q_2)$ *by the factor s*

$$Scale(Q, s) = \begin{pmatrix} sI & 0 \\ Q * (1-s)I & 1 \end{pmatrix} = \begin{pmatrix} s & 0 & 0 \\ 0 & s & 0 \\ (1-s)q_1 & (1-s)q_2 & 1 \end{pmatrix}$$

Non-*Uniform Scaling -- around point* $Q = (q_1, q_2)$ *in direction* $w = (w_1, w_2)$ *by the factor s*

$$Scale(Q, w, s) = \begin{pmatrix} w_1 & w_2 & 0 \\ -w_2 & w_1 & 0 \\ q_1 & q_2 & 1 \end{pmatrix}^{-1} * \begin{pmatrix} sw_1 & sw_2 & 0 \\ -w_2 & w_1 & 0 \\ q_1 & q_2 & 1 \end{pmatrix}$$

Scaling from the Origin along the x-axis by the factor s

$$Scale(Origin, i, s) = \begin{pmatrix} s & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Scaling from the Origin along the y-axis by the factor s

$$Scale(Origin, j, s) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Image of One Point $Q = (x_3, y_3)$ *and Two Vectors* $v_1 = (x_1, y_1)$, $v_2 = (x_2, y_2)$

$$Image(v_1, v_2, Q) = \begin{pmatrix} x_1 & y_1 & 0 \\ x_2 & y_2 & 0 \\ x_3 & y_3 & 1 \end{pmatrix}^{-1} * \begin{pmatrix} x_1^* & y_1^* & 0 \\ x_2^* & y_2^* & 0 \\ x_3^* & y_3^* & 1 \end{pmatrix}$$

Image of Three Non-Collinear Points $P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2)$, $P_3 = (x_3, y_3)$

$$Image(P_1, P_2, P_3) = \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix}^{-1} * \begin{pmatrix} x_1^* & y_1^* & 1 \\ x_2^* & y_2^* & 1 \\ x_3^* & y_3^* & 1 \end{pmatrix}$$

Inverses

$$\begin{pmatrix} x_1 & y_1 & 0 \\ x_2 & y_2 & 0 \\ x_3 & y_3 & 1 \end{pmatrix}^{-1} = \frac{\begin{pmatrix} y_2 & -y_1 & 0 \\ -x_2 & x_1 & 0 \\ x_2y_3 - y_2x_3 & y_1x_3 - x_1y_3 & x_1y_2 - y_1x_2 \end{pmatrix}}{x_1y_2 - y_1x_2}$$

$$\begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix}^{-1} = \frac{\begin{pmatrix} y_2 - y_3 & y_3 - y_1 & y_1 - y_2 \\ x_3 - x_2 & x_1 - x_3 & x_2 - x_1 \\ x_2y_3 - y_2x_3 & y_1x_3 - x_1y_3 & x_1y_2 - y_1x_3 \end{pmatrix}}{\big(x_1(y_2 - y_3) + x_2(y_3 - y_1) + x_3(y_1 - y_2)\big)}$$

Exercises:

1. Show that translation is not a linear transformation on points, but that translation is an affine transformation on points.

2. Show that the formula for scaling a point P around a point Q by a scale factor s is equivalent to

   $P^{new} = (1 - s)Q + sP$.

3. Let w_Q denote the vector from the point Q to the origin.
   a. Without appealing to coordinates and without explicitly multiplying matrices, show that:
      i. $Rot(Q,\theta) = Trans(-w_Q) * Rot(\theta) * Trans(w_Q)$
      ii. $Scale(Q,s) = Trans(-w_Q) * Scale(s) * Trans(w_Q)$
   b. Give a geometric interpretation for the results in part a.

4. Let w_Q denote the vector from the point Q to the origin, let i denote the unit vector along the x-axis, and let α denote the angle between the vectors w and i.

   a. Without appealing to coordinates and without explicitly multiplying matrices, show that:
   $Scale(Q,w,s) = Trans(-w_Q) * Rot(-\alpha) * Scale(Origin, i, s) * Rot(\alpha) * Trans(w_Q)$

   b. Give a geometric interpretation for the result in part a.

5. Show that rotation and scaling about a fixed point commute. That is, show that
   $Rot(Q,\theta) * Scale(Q,s) = Scale(Q,s) * Rot(Q,\theta)$.

6. Verify that if $w = (w_1, w_2)$, then $w^{\perp} = (-w_2, w_1)$ by invoking the formula
   $w^{\perp} = w * Rot(\pi/2)$.

7. What is the geometric effect of the transformation matrix:
$$\begin{pmatrix} w & 0 \\ w^{\perp} & 0 \\ Q & 1 \end{pmatrix}^{-1} * \begin{pmatrix} sw & 0 \\ t\,w^{\perp} & 0 \\ Q & 1 \end{pmatrix}.$$

8. Show that when affine transformations are represented by 3×3 matrices, composition of affine transformations is equivalent to matrix multiplication.

9. Let A be an affine transformation. Using Equation (9) show that for all points P,Q
   $A(Q-P) = A(Q) - A(P)$.

10. Let M be the 3×3 affine matrix that represents the affine transformation A. Show that
$$M = \begin{pmatrix} A(i) & 0 \\ A(j) & 0 \\ A(Origin) & 1 \end{pmatrix}.$$

11. Show that:

    a. the composite of two affine transformations is an affine transformation;

    b. the inverse of a nonsingular affine transformation is an affine transformation.

Conclude that the nonsingular affine transformations form a group.

12. Let M,N be two matrices whose third column is $(0 \quad 0 \quad 1)^T$. Show that:

    a. the third column of $M * N$ is $(0 \quad 0 \quad 1)^T$;

    b. the third column of M^{-1} is $(0 \quad 0 \quad 1)^T$.

Conclude that the nonsingular affine matrices form a group.

13. Verify that

a. $$M = \begin{pmatrix} x_1 & y_1 & 0 \\ x_2 & y_2 & 0 \\ x_3 & y_3 & 1 \end{pmatrix} \Rightarrow M^{-1} = \frac{\begin{pmatrix} y_2 & -y_1 & 0 \\ -x_2 & x_1 & 0 \\ x_2y_3 - y_2x_3 & y_1x_3 - x_1y_3 & x_1y_2 - y_1x_2 \end{pmatrix}}{x_1y_2 - y_1x_2}$$

b. $$M = \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix} \Rightarrow M^{-1} = \frac{\begin{pmatrix} y_2 - y_3 & y_3 - y_1 & y_1 - y_2 \\ x_3 - x_2 & x_1 - x_3 & x_2 - x_1 \\ x_2y_3 - y_2x_3 & y_1x_3 - x_1y_3 & x_1y_2 - y_1x_3 \end{pmatrix}}{\left(x_1(y_2 - y_3) + x_2(y_3 - y_1) + x_3(y_1 - y_2)\right)}$$

14. a. Show that an affine transformation is uniquely determined by the image of two points and one vector that is not parallel to the line determined by the two points.

b. Let P_1^*, P_2^*, v^* be the images of P_1, P_2, v under the affine transformation *A*. Find the 3×3 matrix *M* that represents the affine transformation *A*.

15. Show that every conformal transformation is uniquely determined by:

a. the image of one point and one vector.

b. the image of two points.

16. Show that if *M* is a conformal transformation, then $ScaleFactor(M) = \sqrt{\det(M)}$.

17. Let $Mirror(Q, w)$ denote the matrix representing the transformation that mirrors points and vectors in the line determined by the point *Q* and the direction vector *w*.

a. Show that

i. $$Mirror(Q, w) = \begin{pmatrix} w & 0 \\ w^{\perp} & 0 \\ Q & 1 \end{pmatrix}^{-1} * \begin{pmatrix} w & 0 \\ -w^{\perp} & 0 \\ Q & 1 \end{pmatrix} = Scale(Q, w^{\perp}, -1)$$

b. Conclude that the matrices representing mirroring in the *x* and *y*-axes are given by

ii. $$Mirror(Origin, i) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

iii. $$Mirror(Origin, j) = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

18. *Shear* is the affine transformation defined by mapping a unit square with vertex Q and sides $w, w^{\perp}$ into a parallelogram by tilting the edge $w^{\perp}$ so that $w_{new}^{\perp}$ makes an angle of θ with $w^{\perp}$ (see Figure 10). Show that:

a. $Shear(Q, w, w^{\perp}, \theta) = \begin{pmatrix} w & 0 \\ w^{\perp} & 0 \\ Q & 1 \end{pmatrix}^{-1} * \begin{pmatrix} w & 0 \\ \tan(\theta) w + w^{\perp} & 0 \\ Q & 1 \end{pmatrix}$.

b. $Shear(Origin, i, j, \theta) = \begin{pmatrix} 1 & 0 & 0 \\ \tan(\theta) & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$.

c. $\det\left(Shear(Q, w, w^{\perp}, \theta)\right) = 1$.

d. Shear preserves area.

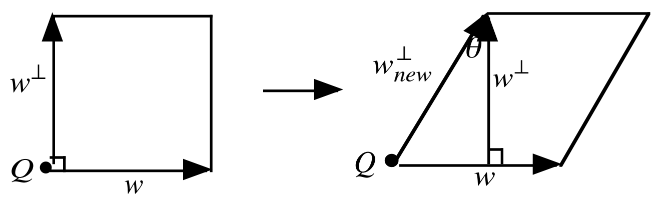

Figure 10: Shear maps the rectangle with vertex Q and sides $w, w^{\perp}$ into a parallelogram by tilting the edge $w^{\perp}$ so that $w_{new}^{\perp}$ makes an angle of θ with $w^{\perp}$.

19. Show that every nonsingular affine transformation is the composition of 1 translation, 1 rotation, 1 shear, and 2 non-uniform scalings.

20. Fix a vector w and a scalar s. Define $A(v) = v + (s-1)\dfrac{v \bullet w}{w \bullet w} w$, where $v \bullet w = v_1 w_1 + v_2 w_2$.

a. Show that

i. $A(w) = sw$

ii. $A(w^{\perp}) = w^{\perp}$

iii. $v = \alpha w + \beta w^{\perp} \Rightarrow A(v) = s\alpha w + \beta w^{\perp}$.

b. Conclude from part *a* that

iv. $v * Scale(Q, w, s) = v + (s-1)\dfrac{v \bullet w}{w \bullet w} w$.

c. Using the result of part *b*, show that for every point P

v. $P * Scale(Q, w, s) = P + (s-1)\dfrac{(P-Q) \bullet w}{w \bullet w} w$.

d. Using the results of parts *b,c,* show that

vi. $$Scale(Q,w,s) = \begin{pmatrix} 1+\dfrac{(s-1)w_1^2}{w_1^2+w_2^2} & \dfrac{(s-1)w_1w_2}{w_1^2+w_2^2} & 0 \\ \dfrac{(s-1)w_1w_2}{w_1^2+w_2^2} & 1+\dfrac{(s-1)w_2^2}{w_1^2+w_2^2} & 0 \\ \dfrac{-(s-1)(q_1w_1^2+q_2w_1w_2)}{w_1^2+w_2^2} & \dfrac{-(s-1)(q_1w_1w_2+q_2w_2^2)}{w_1^2+w_2^2} & 1 \end{pmatrix}.$$

21. Fix *6* points P_1, P_2, P_3 and P_1^*, P_2^*, P_3^* expressed in affine coordinates, and define

$$A(R) = \frac{Det\begin{pmatrix} R^T & P_2^T & P_3^T \end{pmatrix} P_1^* + Det\begin{pmatrix} P_1^T & R^T & P_3^T \end{pmatrix} P_2^* + Det\begin{pmatrix} P_1^T & P_2^T & R^T \end{pmatrix} P_3^*}{Det\begin{pmatrix} P_1^T & P_2^T & P_3^T \end{pmatrix}}.$$

a. Show that

$A(P_i) = P_i^* \quad i = 1,2,3$

b. Using part a, show that the 3×3 matrix representing the affine transformation that maps the points P_1, P_2, P_3 to the points P_1^*, P_2^*, P_3^* is given by

$$Image(P_1,P_2,P_3) = \frac{(P_2\times P_3)^T * P_1^* + (P_3\times P_1)^T * P_2^* + (P_1\times P_2)^T * P_3^*}{Det\begin{pmatrix} P_1^T & P_2^T & P_3^T \end{pmatrix}}.$$

22. In this exercise we are going to prove that for every triangle, up to sign

$$Area(\Delta P_1P_2P_3) = \frac{1}{2}\det\begin{pmatrix} P_1 & 1 \\ P_2 & 1 \\ P_3 & 1 \end{pmatrix} = \frac{1}{2}\det\begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix}. \tag{15}$$

a. Verify that Equation (15) is valid if P_1 is at the origin and P_2 lies on the *x*-axis.

b. Show that for any $\Delta P_1P_2P_3$ there is a rigid motion *R* consisting of a rotation followed by a translation that maps $\Delta P_1P_2P_3$ to $\Delta Q_1Q_2Q_3$, where Q_1 is the origin and Q_2 is a point on the *x*-axis.

c. Show that

i. $Area(\Delta P_1P_2P_3) = Area(\Delta Q_1Q_2Q_3)$

ii. $\det\begin{pmatrix} Q_1 & 1 \\ Q_2 & 1 \\ Q_3 & 1 \end{pmatrix} = \det\begin{pmatrix} P_1 & 1 \\ P_2 & 1 \\ P_3 & 1 \end{pmatrix}$

d. Conclude from parts *a,b,c* that Equation (15) is valid for every triangle.

23. Prove in two ways that nonsingular affine transformations preserve ratios of areas:
   a. by using the fact that nonsingular transformations preserve ratios of distances along lines.
   b. by invoking Equation (15).

24. Let P_1, P_2, P_3 be 3 non-collinear points.
   a. Show that for each point P there exist unique scalars $\beta_1, \beta_2, \beta_3$ such that:
      i. $P = \beta_1 P_1 + \beta_2 P_2 + \beta_3 P_3$
      ii $\beta_1 + \beta_2 + \beta_3 = 1$

   The values $\beta_1, \beta_2, \beta_3$ are called the *barycentric coordinates* of P with respect to $\Delta P_1 P_2 P_3$.
   b. Using Equation (15) and standard properties of determinants, show that up to sign

      iii. $\beta_k = \dfrac{area\left(\Delta P P_i P_j\right)}{area(\Delta P_1 P_2 P_3)} \qquad i,j \neq k$ (see Figure 11).

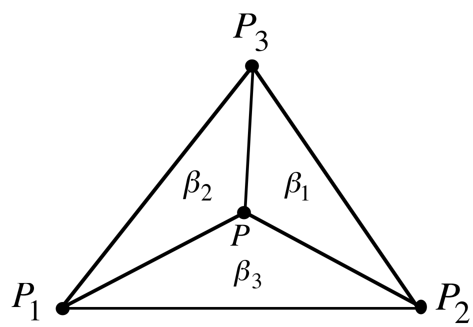

Figure 11: The barycentric coordinates of the point P relative to $\Delta P_1 P_2 P_3$.

25. Let P_1, P_2, P_3 be 3 non-collinear points, and let $\beta_1, \beta_2, \beta_3$ be the barycentric coordinates of P with respect to $\Delta P_1 P_2 P_3$ (see Exercise 24).
   a. Show that if A is an affine transformation, then
      $A(P) = \beta_1 A(P_1) + \beta_2 A(P_2) + \beta_3 A(P_3)$.
   b. Conclude that the barycentric coordinates of $A(P)$ with respect to $\Delta A(P_1)A(P_2)A(P_3)$ are identical to the *barycentric coordinates* of P with respect to $\Delta P_1 P_2 P_3$ (see Figure 12).

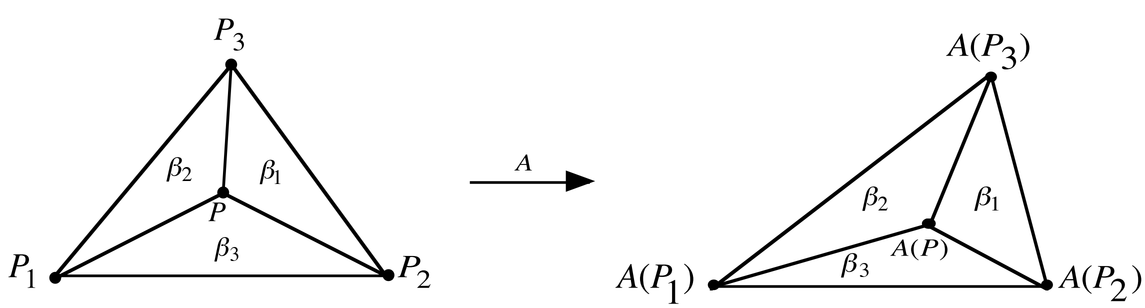

Figure 12: Affine transformations preserve barycentric coordinates.