

Introduction to Computer Graphics (CS602) Lecture 12 **2D Transformations II**

Before starting our next lecture just recall equations of three basic transformations i.e. translation, rotation and scaling:

Translation: $P' = P + T$

Rotation: $P' = R \cdot P$

Scaling: $P' = S \cdot P$

In many cases of computer graphics applications we require sequence of transformations. For example in animation on each next move we may have object to be translated than scaled. Similarly in games an object in a particular moment may have to be rotated as well as translated. That means we have to perform sequence of matrix operations but the matrix we have seen in the previous lecture have order which restrict them to be operated in sequence. However, with slight reformulated we can bring them into the form where they can easily be operated in any sequence thus efficiency can be achieved.

12.1 Homogeneous Coordinates

Again considering our previous lecture all the three basic transformations covered in that lecture can be expressed by following equation:

$$P' = M_1 \cdot P + M_2$$

With coordinate positions P and P' represented as column vectors. Matrix M_1 is a 2 by 2 array containing multiplicative factors, and M_2 is a two-element column matrix containing translation terms. For translation, M_1 is a the identity matrix, For rotation or scaling, M_2 contains the translational terms associated with the pivot point or scaling fixed point. To produce a sequence of transformations with these equations, such as scaling followed by rotation then translation, we must calculate the transformed coordinate's one step at a time. First, coordinate positions are scaled, then these scaled coordinates are rotated, and finally the rotated coordinates are translated.

Now the question is can we find a way to eliminate the matrix addition associated with translation? Yes, we can but for that M_1 will have to be rewritten as a 3x3 matrix and also the coordinate positions will have to be expressed as a homogeneous coordinate triple:

(x, y) as (x_h, y_h, h) where

$$x = \frac{x_h}{h}, \quad y = \frac{y_h}{h}$$

We can choose the h as any non-zero value. However, a convenient choice is 1, thus (x, y) has homogeneous coordinates as $(x, y, 1)$. Expressing positions in homogeneous coordinates allows us to represent all geometric transformation equations as matrix multiplications. Coordinates are represented with three-element column vectors, and transformation operations are written as 3 by 3 matrices.

a) Translation with Homogeneous Coordinates

The translation can now be expressed using homogeneous coordinates as:

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

Abbreviated as:

$$P' = T(t_x, t_y) \cdot P$$

b) Rotation with Homogeneous Coordinates

The rotation can now be expressed using homogeneous coordinates as:

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

Abbreviated as:

$$P' = R(\theta) \cdot P$$

c) Scaling with Homogeneous Coordinates

The scaling can now be expressed using homogeneous coordinates as:

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

Abbreviated as:

$$P' = S(S_x, S_y) \cdot P$$

Matrix representations are standard methods for implementing transformations in

graphics systems. In many systems, rotation and scaling functions produce transformations with respect to the coordinate origin as expressed in the equation above. Rotations and scalings relative to other reference positions are then handled as a succession of transformation operations.

12.2 Composite Transformations

As in the previous section we achieved homogenous matrices for each of the basic transformation, we can find a matrix for any sequence of transformation as a composite transformation matrix by calculating the matrix product of the individual transformations.

a) Translations

If two successive translations vectors (tx_1, ty_1) and (tx_2, ty_2) are applied to a coordinate position P , the final transformed location P is calculated as

$$\begin{aligned} P' &= T(tx_2, ty_2) \cdot \{T(tx_1, ty_1) \cdot P\} \\ &= \{T(tx_2, ty_2) \cdot T(tx_1, ty_1)\} \cdot P \end{aligned}$$

where P and P' are represented as homogeneous-coordinate column vectors. The composite transformation matrix for this sequence of translations is

$$\begin{bmatrix} 1 & 0 & t_{x2} \\ 0 & 1 & t_{y2} \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & t_{x1} \\ 0 & 1 & t_{y1} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_{x1} + t_{x2} \\ 0 & 1 & t_{y1} + t_{y2} \\ 0 & 0 & 1 \end{bmatrix}$$

or

$$T(tx_2, ty_2) \cdot T(tx_1, ty_1) = T(tx_1 + tx_2, ty_1 + ty_2)$$

Which means that two successive translations are additive. Hence,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_{x1} + t_{x2} \\ 0 & 1 & t_{y1} + t_{y2} \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

b) Composite Rotations

Two successive Rotations applied to a point P produce the transformed position

$$\begin{aligned} P' &= R(\theta_2) \cdot \{R(\theta_1) \cdot P\} \\ &= \{R(\theta_2) \cdot R(\theta_1)\} \cdot P \end{aligned}$$

By multiplying the two rotation matrices, we can verify that two successive rotations are additive:

$$R(\theta_2) \cdot R(\theta_1) = R(\theta_1 + \theta_2)$$

so that the final rotated coordinates can be calculated with the composite rotation matrix as

$$P' = R(\theta_1 + \theta_2) \cdot P$$

c) Composite Scalings

Concatenating transformation matrices for two successive scaling operations produces the following composite scaling matrix:

$$\begin{bmatrix} S_{x2} & 0 & 0 \\ 0 & S_{y2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} S_{x1} & 0 & 0 \\ 0 & S_{y1} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} S_{x1} \cdot S_{x2} & 0 & 0 \\ 0 & S_{y1} \cdot S_{y2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

or

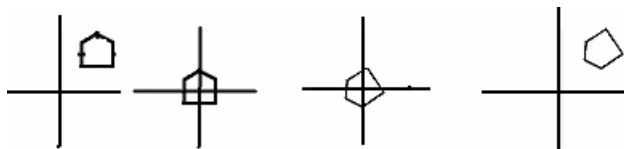
$$S(sx_2, sy_2) \cdot S(sx_1, sy_1) = S(sx_1 \cdot sx_2, sy_1 \cdot sy_2)$$

The resulting matrix in the case indicates that successive scaling operations are multiplicative. That is, if we were to triple the size of an object twice in succession, the final size would be nine times that of the original.

12.3 General Pivot Point Rotation

With a graphics package that only provides a rotate function for revolving object about the coordinate origin, we can generate rotations about any selected pivot point (x_r, y_r) by performing the following sequence of translate-rotate-translate operations:

1. Translate the object so that the pivot-point positions is moved to the coordinate origin
2. Rotate the object about the coordinate origin
3. Translate the object so that the pivot point is returned to its original position



$$\begin{bmatrix} 1 & 0 & x_r \\ 0 & 1 & y_r \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_r \\ 0 & 1 & -y_r \\ 0 & 0 & 1 \end{bmatrix} \\
= \begin{bmatrix} \cos\theta & -\sin\theta & x_r(1-\cos\theta) + y_r \sin\theta \\ \sin\theta & \cos\theta & y_r(1-\cos\theta) - x_r \sin\theta \\ 0 & 0 & 1 \end{bmatrix}$$

which can be expressed in the form

$$T(x_r, y_r) \cdot R(\theta) \cdot T(-x_r, -y_r) = R(x_r, y_r, \theta)$$

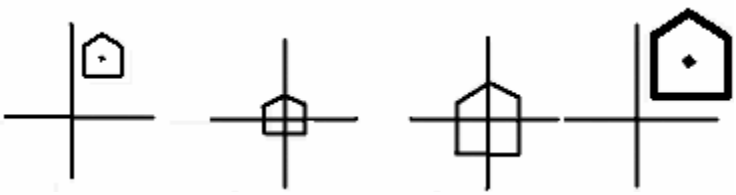
where $T(-x_r, -y_r) = T^{-1}(x_r, y_r)$.

12.4 General Fixed Point Scaling

Following figure is showing a transformation sequence to produce scaling with respect to a selected fixed point (x_f, y_f) using a scaling function that can only scale relative to the coordinate origin.

1. Translate object so that the fixed point coincides with the coordinate origin
2. Scale the object with respect to the coordinate origin
3. Use the inverse translation of step 1 to return the object to its original position

Concatenating the matrices for these three operations produces the required scaling matrix.



$$\begin{bmatrix} 1 & 0 & x_f \\ 0 & 1 & y_f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_f \\ 0 & 1 & -y_f \\ 0 & 0 & 1 \end{bmatrix} \\
= \begin{bmatrix} S_x & 0 & x_f(1-S_x) \\ 0 & S_y & y_f(1-S_y) \\ 0 & 0 & 1 \end{bmatrix}$$

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

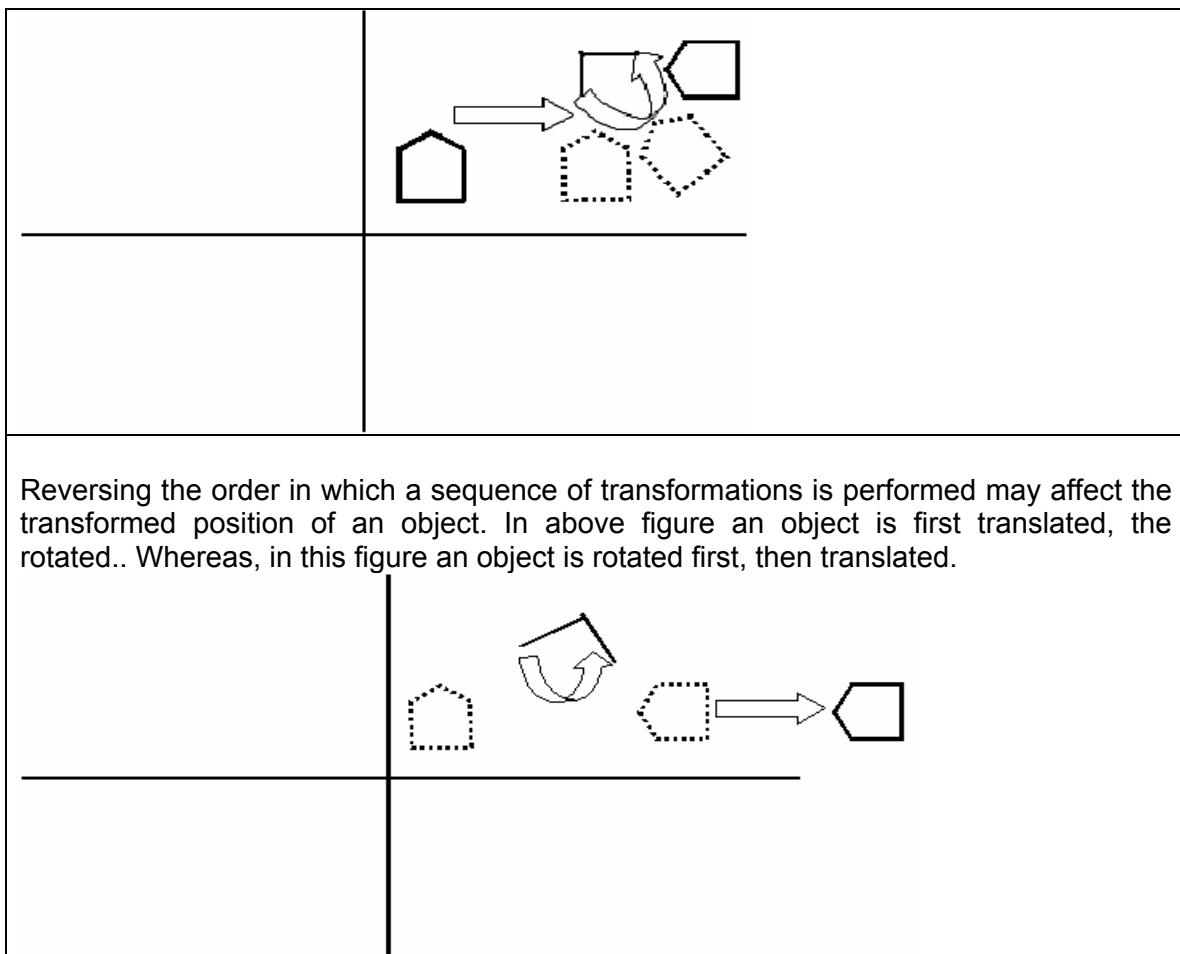
This transformation is automatically generated on systems that provide a scale function that accepts coordinates for the fixed point.

12.5 Concatenation Properties

Matrix multiplication is associative. For any three matrices A, B and C, the matrix product A . B . C can be performed by first multiplying A and B or by first multiplying B and C:

$$A . B . C = (A . B) . C = A . (B . C)$$

Therefore, we can evaluate matrix products using a left-to-right or a right-to-left associative grouping. On the other hand, transformation products may not be commutative. The matrix product A . B is not equal to B . A, in general. This means that if we want to translate and rotate an object, we must be careful about the order in which the composite matrix is evaluated as show in following figure.



For some special cases, such as a sequence of transformations all of same kind, the multiplication of transformation matrices is commutative. As an example, two successive rotations could be performed in either order and the final position would be the same. This commutative property holds also for two successive translations or two successive scalings. Another commutative pair of operation is rotation and uniform scaling ($S_x = S_y$).

12.6 General Composite Transformations and Computational Efficiency

A general two-dimensional transformation, representing a combination of translations, rotations, and scaling, can be expressed as

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} rS_{xx} & rS_{xy} & trs_x \\ rS_{yx} & rS_{yy} & trs_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

The four elements rs_{ij} are the multiplicative rotation-scaling terms in the transformation that involve only rotating angles and scaling factors. Elements trs_x and trs_y are the translational terms containing combinations of translation distances, pivot-point and fixed-point coordinates, and rotation angles and scaling parameters. For example, if an object is to be scaled and rotated about its centroid coordinates (x_c, y_c) and then translated, the values for the elements of the composite transformation matrix are

$$T(t_x, t_y) \cdot R(x_c, y_c, \theta) \cdot S(x_c, y_c, s_x, s_y)$$

$$= \begin{bmatrix} S_x \cos \theta & -S_y \sin \theta & x_c(1 - S_x \cos \theta) + y_c S_y \sin \theta + t_x \\ S_x \sin \theta & S_y \cos \theta & y_c(1 - S_y \cos \theta) - x_c S_x \sin \theta + t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Although matrix given before above matrix requires nine multiplications and six additions, the explicit calculations for the transformed coordinates are

$$\begin{aligned} x' &= x \cdot rs_{xx} + y \cdot rs_{xy} + trs_x \\ y' &= x \cdot rs_{yx} + y \cdot rs_{yy} + trs_y \end{aligned}$$

Thus, we actually only need to perform four multiplications and four additions to transform coordinate positions. This is the maximum number of computations

required for any transformation sequence, once the individual matrices have been concatenated and the elements of the composite matrix evaluated. Without concatenation, the individual transformations would be applied one at a time and the number of calculations could be significantly increased. An efficient implementation for the transformation operations, therefore, is to formulate transformation matrices, concatenate any transformation sequence, and calculate transformed coordinates using above equations.

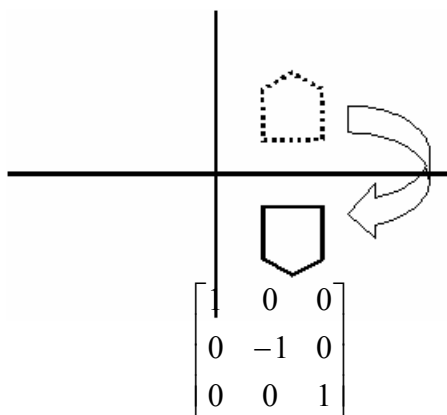
12.7 Other Transformations

Basic transformations such as translation, rotation, and scaling are included in most graphics packages. Some packages provide a few additional transformations that are useful in certain applications. Two such transformations are reflection and shear.

a) Reflection

A reflection is a transformation that produces a mirror image of an object. The mirror image for a two-dimensional reflection is generated relative to an axis of reflection by rotating the object 180° about the reflection axis. We can choose an axis of reflection in the xy plane or perpendicular to the xy plane. When the reflection axis is a line in the xy plane; the rotation path about this axis is in a plane perpendicular to the xy plane. For reflection axes that are perpendicular to the xy plane, the rotation path is in the xy plane. Following are examples of some common reflections.

Reflection about the line $y=0$, the x-axis, relative to axis of reflection can be achieved by rotating the object about axis of reflection by 180° .

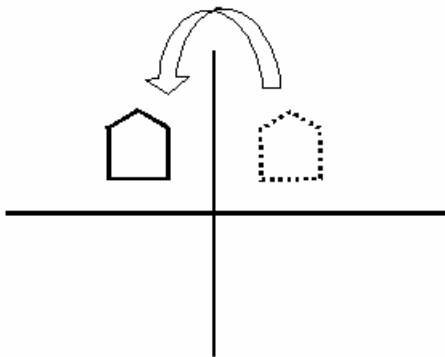


The transformation matrix is

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

Similarly in case of reflection about y-axis the transformation matrix will be, also the reflection is shown in following figure:

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



b) Shear

A transformation that distorts the shape of an object such that the transformed shape appears as if the object were composed of internal layers that had been caused to slide over each other is called a shear. Two common shearing transformations are those that shift coordinate x values and those that shift y values.

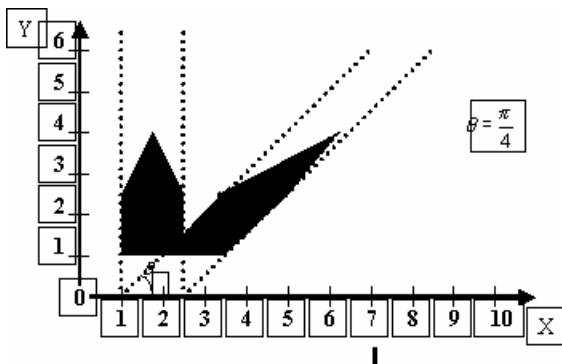
An x direction shear relative to the x axis is produced with the transformation matrix

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

which transforms coordinate position as

- $x' = x + sh_x \cdot y$
- $y' = y$

Any real number can be assigned to the shear parameter sh_x . A coordinate position (x,y) is then shifted horizontally by an amount proportional to its distance (y value) from the x axis ($y=0$). Setting sh_x to 2, for example, changes the square in following figure into a parallelogram. Negative values for sh_x shift coordinate positions to the left.



Similarly y-direction shear relative to the y-axis is produced with the transformation matrix

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

and coordinate positions transformed as

- $x' = x$
- $y' = sh_y \cdot x + y$

Another similar transformation may be in x and y direction shear, where matrix will be

$$\begin{bmatrix} 1 & sh_x & 0 \\ sh_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

and coordinate positions transformed as

- $x' = x + sh_x \cdot y$
- $y' = sh_y \cdot x + y$