Computer Graphics
and Visualisation

Geometry for Computer Graphics

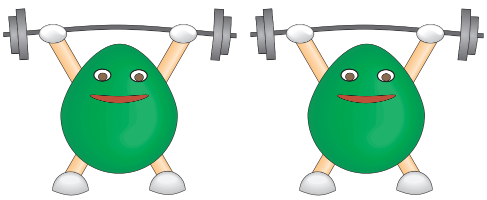
Student Notes

Developed by

F Lin
K Wyrwas
J Irwin
C Lilley
W T Hewitt
T L J Howard

Computer Graphics Unit
Manchester Computing Centre
University of Manchester

Department of Computer Science
University of Manchester



THE UNIVERSITY
of MANCHESTER

Produced by the ITTI Gravigs Project
Computer Graphics Unit
Manchester Computing Centre
MANCHESTER M13 9PL

First published by UCoSDA in May 1995

ISBN 1 85889 059 4

For further information on this and other modules please contact:
The ITTI Gravigs Project, Computer Graphics Unit, Manchester Computing Centre.
Tel: 0161 275 6095 Email: *gravigs@mcc.ac.uk*

To order further copies of this or other modules please contact:
Mrs Jean Burgan, UCoSDA.
Tel: 0114 272 5248 Email: *j.burgan@sheffield.ac.uk*

These materials have been produced as part of the Information Technology Training Initiative, funded by the Information Systems Committee of the Higher Education Funding Councils.

The authors would like to thank Janet Edwards for her assistance in the preparation of these documents.

Printed by the Reprographics Department, Manchester Computing Centre from PostScript source supplied by the authors.

Table of Contents

1 2D Transformations

1.1 Introduction

In computer graphics many applications need to alter or manipulate a picture, for example, by changing its size, position or orientation. This can be done by applying a geometric transformation to the coordinate points defining the picture. These notes cover the basic theory of two-dimensional (2D) geometric transformations.

1.2 Types of Transformation

1.2.1 Translation

A common requirement is to move a picture to a new position, as in Figure 1. (The original object is drawn using dotted lines, and the transformed object using solid lines. This convention will be used throughout.)

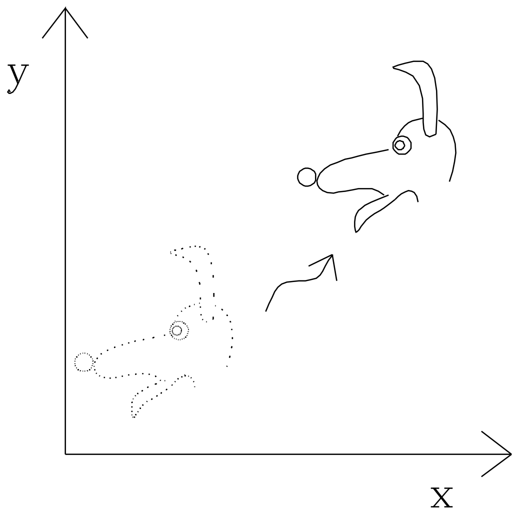


Figure 1: moving a picture

This is achieved by means of a translation or shift transformation. In order to translate a point, constant values are added to the *x*- and *y*-coordinates of the point, as shown in Figure 2.

The new coordinates of the point (x', y') are given by

$$x' = x + t_x$$
$$y' = y + t_y$$

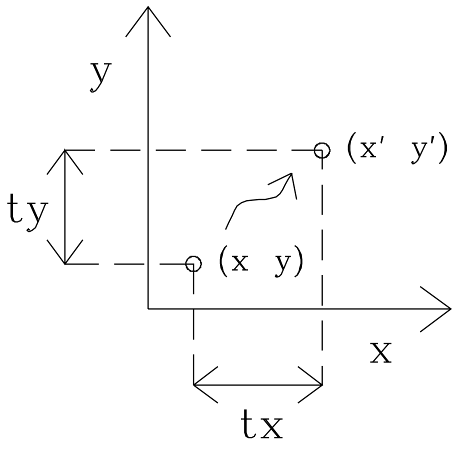


Figure 2: translating a point

1.2.2 Scaling

A scaling transformation is used to change the size of an object. Scaling about the origin (0, 0) is achieved by multiplying the coordinates of a point by *x*- and *y*-scale factors:

$$x' = x \cdot s_x$$
$$y' = y \cdot s_y$$

If $|s_x|$ and $|s_y|$ are both >1, the effect is of increasing the size of an object. In order to reduce the size, $|s_x|$ and $|s_y|$ must be <1.

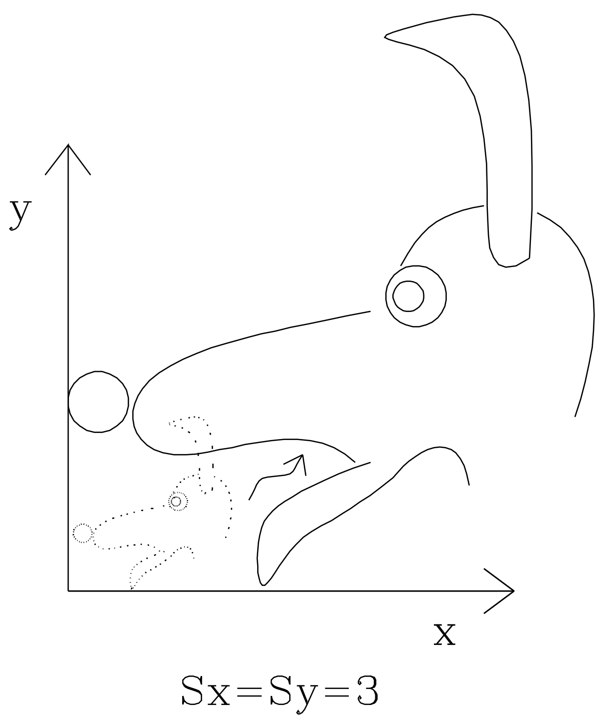


Figure 3: symmetric scaling

Figure 3 illustrates a symmetric or uniform scaling transformation in which the *x*- and *y*-scale factors are the same ($s_x = s_y$) so that the object is expanded by the same amount in each axis direction.

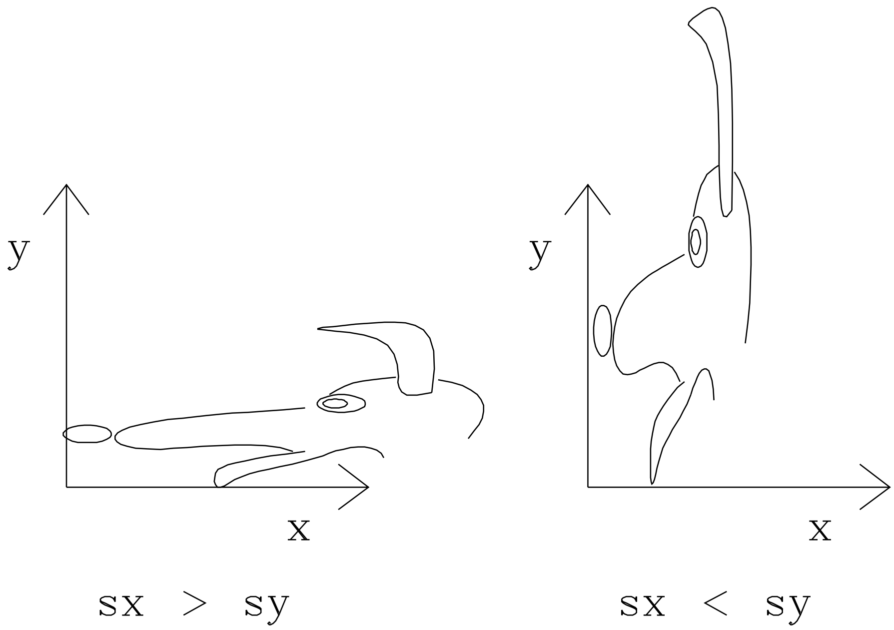


Figure 4: asymmetric scaling

Figure 4 illustrates two asymmetric or non-uniform scaling transformations in which the *x*- and *y*-scale factors are not equal ($s_x \neq s_y$). Here the object changes its size by different amounts in the *x*- and *y*-axis directions.

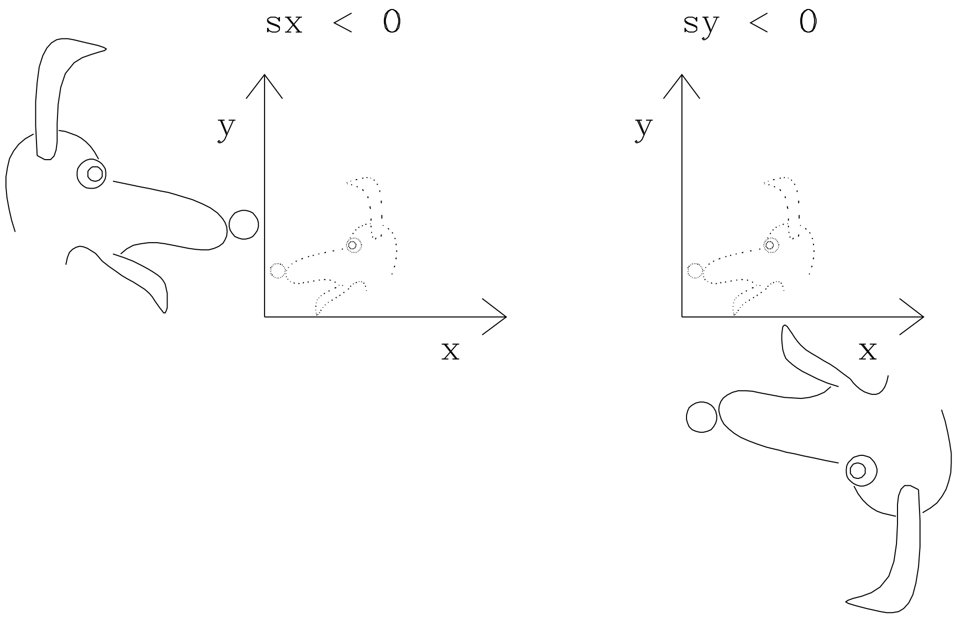


Figure 5: the effect of negative scale factors

If the scale factor in *x* is negative ($s_x < 0$) then the object is reflected in the *y*-axis. Similarly, if the scale factor in *y* is negative ($s_y < 0$) then the object is reflected in the *x*-axis. These two cases are shown in Figure 5 .

1.2.3 Rotation

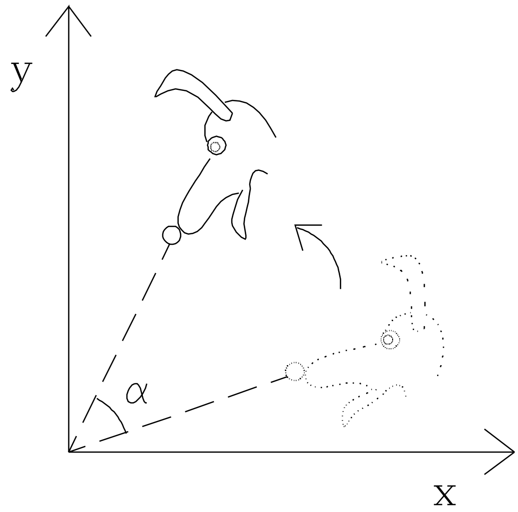


Figure 6: rotating an object about the origin

Another common type of transformation is rotation. This is used to orientate objects. Figure 6 shows an object rotated by an angle α about the origin.

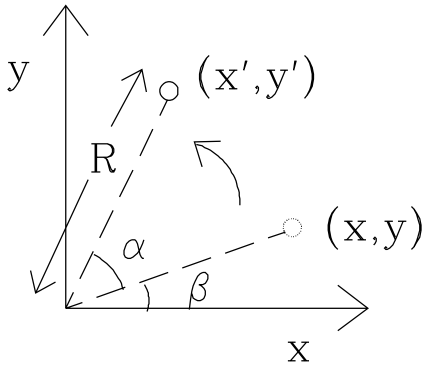


Figure 7: rotating a point about the origin

The rotation of one point in the object is illustrated in Figure 7. A line joining the point with the origin makes an angle β with the *x*-axis and has length *R*, hence

$$x = R \cdot \cos\beta$$
$$y = R \cdot \sin\beta$$

After rotation the point has coordinates *x*' and *y*' with values

$$x' = R \cdot \cos(\alpha + \beta)$$
$$y' = R \cdot \sin(\alpha + \beta)$$

Expanding these formulae for cos(α+β) and sin(α+β) and rearranging gives

$$x' = R \cdot \cos\alpha \cdot \cos\beta - R \cdot \sin\alpha \cdot \sin\beta$$
$$y' = R \cdot \sin\alpha \cdot \cos\beta + R \cdot \sin\beta \cdot \cos\alpha$$

Finally, substituting for $R \cdot \cos\beta$ and $R \cdot \sin\beta$ gives

$$x' = x \cdot \cos\alpha - y \cdot \sin\alpha$$
$$y' = x \cdot \sin\alpha + y \cdot \cos\alpha$$

1.2.4 Shearing

A shear transformation has the effect of distorting the shape of an object.

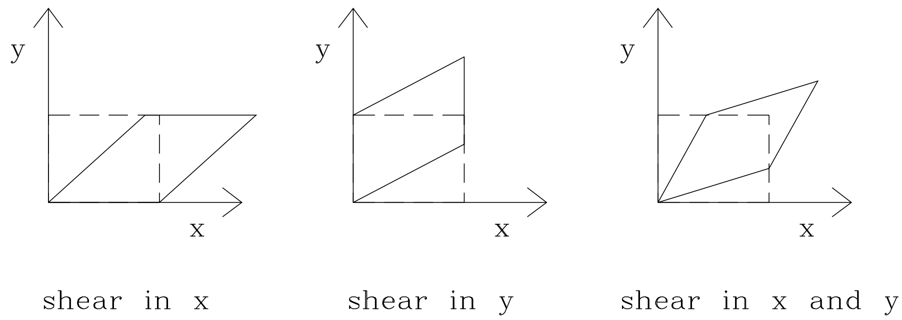


Figure 8: shear transformations

Figure 8 illustrates several different kinds of shear transformation applied to a rectangular object. The first diagram shows a shear in x in which the x-coordinates of points are displaced as a function of their height. The middle diagram shows a shear in y, where the y-coordinates are displaced according to their x-coordinate. Finally, a shear in both x and y is shown. The new x- and y-coordinates of a point after shearing are given by

$$x' = x + y \cdot a$$
$$y' = y + x \cdot b$$

If $a \neq 0$ then a shear in x is obtained. Similarly, if $b \neq 0$ then a shear in y is obtained.

1.3 Matrix Representation of Transformations

In the last section we looked at the basic types of transformation and for each derived an expression for the new coordinates of a point after transformation. We can now write down a general formula for the transformation of points

$$x' = a \cdot x + b \cdot y + c$$
$$y' = d \cdot x + e \cdot y + f$$

where a, b, c, d, e and f are all constants. The expressions for x' and y' are linear functions of x and y. This can be re-expressed using matrices as

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ d & e \end{bmatrix} \bullet \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} c \\ f \end{bmatrix}$$

Now include all of the constants in one matrix

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

A square matrix is much easier to deal with so the matrix is extended to a 3×3 matrix

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

The column vectors representing points now have an extra entry. If the bottom row of the matrix is [0 0 1] then w' will be 1 and we can ignore it. The effect of setting the bottom row of the matrix to values other than [0 0 1] is dealt with later (see section 6 - Homogeneous Coordinates).

The formulae for each of the different types of transformation can now be re-written using this matrix notation:

- Translate $\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$
- Scale $\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$
- Rotate $\begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$
- Shear $\begin{bmatrix} 1 & a & 0 \\ b & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

There is a special matrix which leaves the coordinates x' and y' equal to x and y. This is known as the unit or identity matrix:

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

$$x' = x$$
$$y' = y$$
$$w' = 1$$

1.4 Concatenation of Transformations

We have examined the basic types of transformation and derived the corresponding matrices. In this section we will see how these transformations can be combined to perform more complex operations such as rotation or scaling about an arbitrary point.

Consider the transformation to rotate an object about its centre point (x_c, y_c). This can be broken down into a series of basic transformations as follows:

- Translate the object by ($-x_c$, $-y_c$) so that the centre coincides with the origin.
- Perform a rotation about the origin.
- Translate the object by (x_c, y_c) to return it to its original position.

This series of transformations is illustrated in Figure 9.

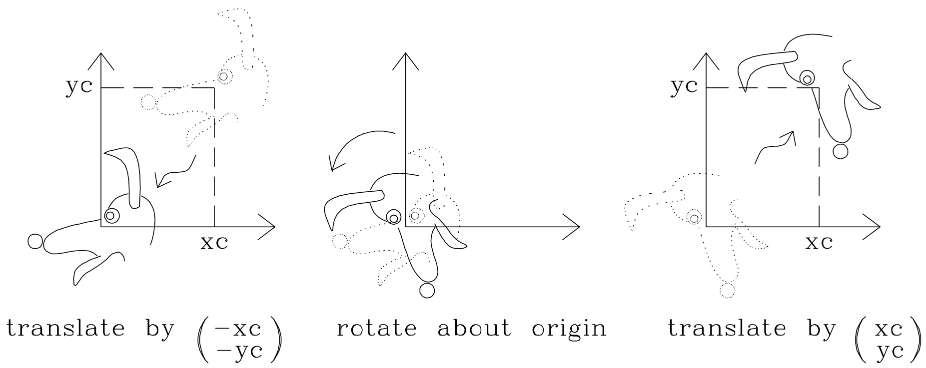


Figure 9: rotating an object about its centre

How can this composite transformation be expressed in terms of matrices? If we apply each of the component transformations separately:

- Translate by ($-x_c$, $-y_c$):

$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -x_c \\ 0 & 1 & -y_c \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Rotate by an angle about the origin:

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} 1 & 0 & -x_c \\ 0 & 1 & -y_c \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Translate by (x_c, y_c):

$$\begin{bmatrix} x_3 \\ y_3 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_c \\ 0 & 1 & y_c \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & x_c \\ 0 & 1 & y_c \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} 1 & 0 & -x_c \\ 0 & 1 & -y_c \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\alpha & -\sin\alpha & (x_c - \cos\alpha \cdot x_c + \sin\alpha \cdot y_c) \\ \sin\alpha & \cos\alpha & (y_c - \sin\alpha \cdot x_c - \cos\alpha \cdot y_c) \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

The net effect of the transformation is to map the point (x, y) onto the point (x_3, y_3). This mapping can be expressed as the matrix multiplication of the three basic transformation matrices used. The value of using square matrices to represent transformations can now be seen. Square matrices can be multiplied together to produce another square matrix of the same dimensions. Hence composite transformations can be expressed as a single transformation matrix by multiplying them together. Each point to be transformed is multiplied by this matrix which performs all the component transformations in one step. When there are many points to be transformed, this is considerably more efficient than multiplying the points by each component transformation matrix in turn.

1.5 Ordering Transformations

We have already seen how more than one transformation can be combined by multiplying together the corresponding transformation matrices. Matrix multiplication is not a commutative operation $\mathbf{M}_1 \bullet \mathbf{M}_2 \neq \mathbf{M}_2 \bullet \mathbf{M}_1$. In the same way, the application of transformations is not, in general, commutative and therefore the order in which transformations are combined is important. For example, consider the two transformations illustrated in Figure 10 .

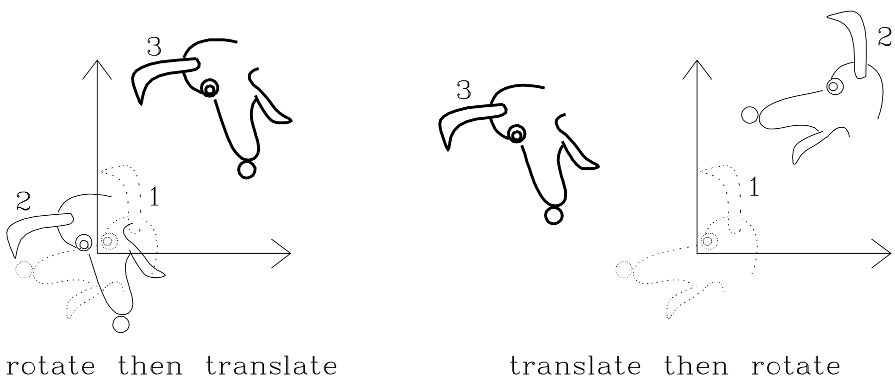


Figure 10: different orders of transformations

The first example shows the effect of rotating and then translating the object. The second example does the same translation and rotation but in a different order, first translating and then rotating the object. The effect in both cases is clearly not the same.

The transformation matrix $\mathbf{M}_1$ maps the point $\mathbf{p}$ onto the point $\mathbf{p}'$:

$$\mathbf{p}' = \mathbf{M}_1 \bullet \mathbf{p}$$

If a second transformation $\mathbf{M}_2$ is to be combined with $\mathbf{M}_1$ such that $\mathbf{M}_1$ is applied first followed by $\mathbf{M}_2$, then $\mathbf{M}_2$ is postconcatenated with $\mathbf{M}_1$ so that

$$\mathbf{p}' = \mathbf{M}_2 \bullet \mathbf{M}_1 \bullet \mathbf{p}$$

Alternatively, $\mathbf{M}_2$ may be preconcatenated with $\mathbf{M}_1$. This will cause $\mathbf{M}_2$ to be applied first:

$$\mathbf{p}' = \mathbf{M}_1 \bullet \mathbf{M}_2 \bullet \mathbf{p}$$

Note that the order in which transformations are applied can be seen by reading outwards from the vector being transformed. In other words, the transformation which is applied first appears closest to the vector.

Two other terms used for combining transformation matrices are premultiply and postmultiply. Premultiply corresponds to postconcatenate and postmultiply corresponds to preconcatenate. The *pre* and *post* terms in the example transformation program refer to *pre*multiply and *post*multiply.

1.6 Homogeneous Coordinates

To obtain square matrices an additional row was added to the matrix and an additional coordinate, the *w*-coordinate, was added to the vector for a point. In this way a point in 2D space is expressed in three-dimensional homogeneous coordinates. This technique of representing a point in a space whose dimension is one greater than that of the point is called homogeneous representation. It provides a consistent, uniform way of handling affine transformations.

On converting a 2D point (x, y) to homogeneous coordinates the *w*-coordinate is set to 1, giving the corresponding homogeneous coordinate point $(x, y, 1)$. This may then be transformed by the 3×3 homogeneous transformation matrix as shown below.

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

This gives the transformed point (x', y', w'). All of the transformation matrices examined up to now have had [0 0 1] in the bottom row and therefore w' has always been 1. In this case the transformed 2D point is (x', y'). In general, the elements in the bottom row of the matrix, g, h and i, may be set to any value resulting in $w' \neq 1$. The effect of this general transformation matrix is to transform a point $(x, y, 1)$ in the $w = 1$ plane onto the point (x', y', w') in the $w = w'$ plane. The *real-world* coordinate space is the plane $w = 1$ and therefore the transformed point must be mapped back onto the $w = 1$ plane. This is done by projecting the point

(x', y', w') onto the plane $w = 1$, as shown in Figure 11. This process is known as homogeneous division.

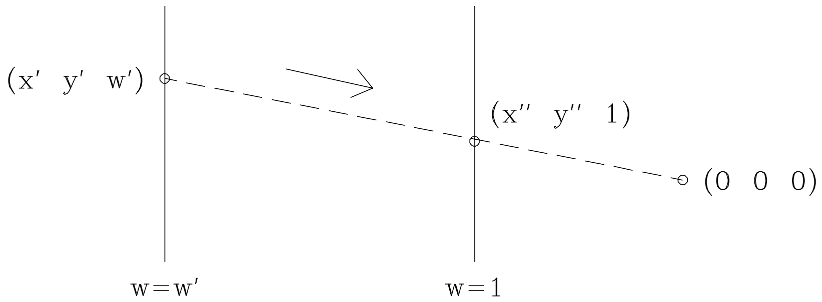


Figure 11: homogeneous division

The 2D *real-world* point is (x'', y'') where x'' and y'' are the x- and y-coordinates of the projected point. The mathematical effect of the projection is that of dividing the x- and y-components by the w-component. Hence

$$x'' = x'/w'$$
$$y'' = y'/w'$$

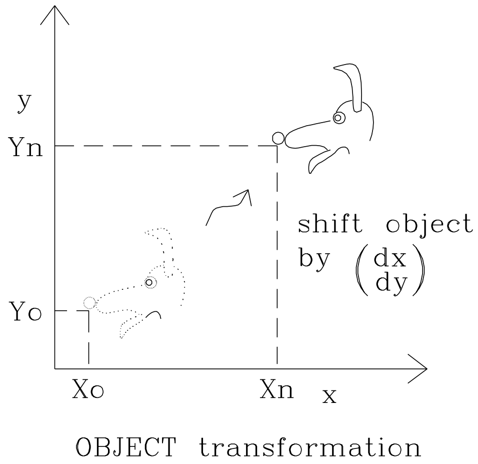
Now let's look at an example of a transformation matrix with values other than [0 0 1] on the bottom row. Consider the following transformation:

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 4 \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

The resulting homogeneous point is $(x, y, 4)$. We obtain the corresponding 2D point by performing the homogeneous division. This gives us the point $(x/4, y/4)$. From this we can see that the bottom right element in the matrix performs overall or uniform scaling.

1.7 Object and Axis Transformations

The types of transformation we have examined up to now are known as object transformations. We think of the object being transformed, while the axes remain fixed. There is another way of looking at transformations - as axis transformations. Here, the object remains fixed while the axes are changed. Figure 12 illustrates the difference between these two types of transformation. In the first example, the sequence of points making up the object are shifted by (d_x, d_y), the transformed points are plotted relative to the same set of axes. This is an object transformation.



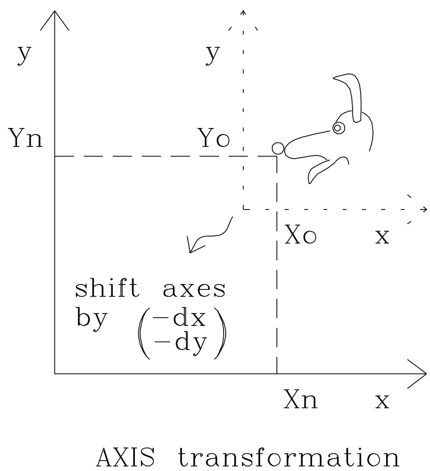


Figure 12: equivalent object and axis transformations

The second example shows an axis transformation in which the axes are shifted by ($-d_x$, $-d_y$). This time the points are plotted with respect to the new axes, although they remain fixed in space. After the object transformation which shifts the object by (d_x, d_y), the new coordinates of the point (x_0, y_0) are given by

$$x_n = x_0 + d_x$$
$$y_n = y_0 + d_y$$

This is the same as the new coordinates of the same point after the axis transformation shifting the axes by ($-d_x$, $-d_y$). We can deduce from this that an axis translation is equivalent to an equal and opposite object translation. This rule also applies to the other types of transformation. Hence an object transformation which scales an object by (s_x, s_y) is equivalent to an axis transformation which scales the axes by ($1/s_x$, $1/s_y$). Similarly the rotation of an object by α is equivalent to an axis rotation of $-\alpha$. The transformation matrices corresponding to some common object and axis transformations are given below:

- Translate OBJECT by (t_x, t_y) $\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$

- Translate AXES by (t_x, t_y) $\begin{bmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{bmatrix}$

- Scale OBJECT by (s_x, s_y) $\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$

- Scale AXES by (s_x, s_y) $\begin{bmatrix} 1/s_x & 0 & 0 \\ 0 & 1/s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$

- Rotate OBJECT by α $\begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$

- Rotate AXES by $-\alpha$ $\begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$

As a general rule, the inverse of an object transformation is the corresponding axis transformation.

1.8 The Normalization Transformation in GKS

The normalization transformation in GKS maps the contents of a window in world coordinate space into a viewport specified in normalized device coordinate space. This is shown in Figure 13.

This type of window to viewport transformation is very common in graphics systems. The window and viewport are aligned with their respective axes and are therefore defined by their bottom left and top right hand corners. The two corners of the window are given by the points (w_{xmin}, w_{ymin}) and (w_{xmax}, w_{ymax}). The corresponding corners of the viewport are (v_{xmin}, v_{ymin}) and (v_{xmax}, v_{ymax}). The stages in the transformation are as follows:

- Apply a translation to map the bottom left hand corner of the window to the origin:

$$\mathbf{P}' = \begin{bmatrix} 1 & 0 & -w_{xmin} \\ 0 & 1 & -w_{ymin} \\ 0 & 0 & 0 \end{bmatrix} \bullet \mathbf{P}$$

- Next apply a scaling to make the size of the window the same as the size of the viewport. If the x- and y-scale factors, s_x and s_y are

$$s_x = \frac{v_{xmax} - v_{xmin}}{w_{xmax} - w_{xmin}}$$

$$s_y = \frac{v_{ymax} - v_{ymin}}{w_{ymax} - w_{ymin}}$$

the transformation becomes

$$\mathbf{P}' = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} 1 & 0 & -w_{xmin} \\ 0 & 1 & -w_{ymin} \\ 0 & 0 & 0 \end{bmatrix} \bullet \mathbf{P}$$

- Finally, apply a translation so that a point at the origin is mapped onto the bottom left hand corner of the viewport:

$$\mathbf{P}' = \begin{bmatrix} 1 & 0 & v_{xmin} \\ 0 & 1 & v_{ymin} \\ 0 & 0 & 0 \end{bmatrix} \bullet \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} 1 & 0 & -w_{xmin} \\ 0 & 1 & -w_{ymin} \\ 0 & 0 & 0 \end{bmatrix} \bullet \mathbf{P}$$

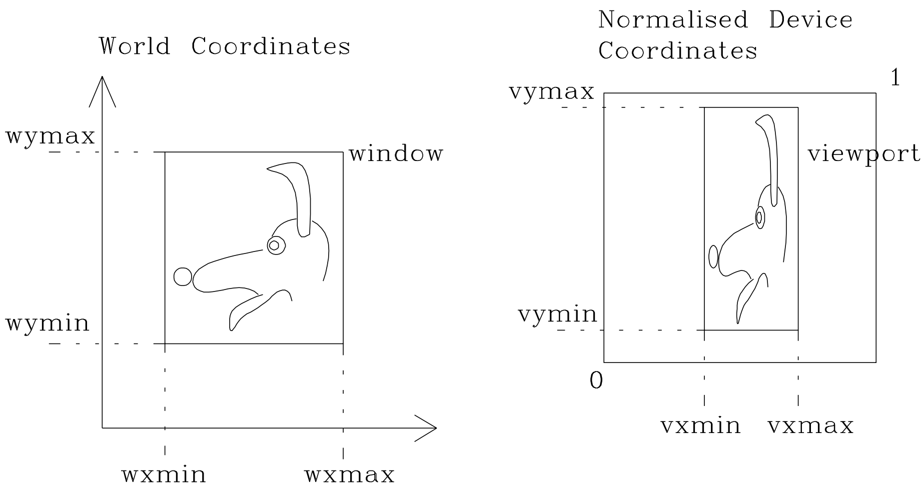


Figure 13: the normalization transformation in GKS

1.9 Summary

We have examined the basic types of transformation: translation, scale, rotation and shear. A transformation may be either an object transformation in which the points of the object are transformed, or an axis transformation in which the coordinate axes are transformed and the object points re-expressed relative to the new axes. All of these transformations can be expressed in a 3×3 matrix which is multiplied with the vector for a point to obtain the coordinates of the transformed point. A 3×3 matrix is used to enable different transformations to be combined by multiplying the matrices together. This means that a 2D point to be transformed must be represented as a three-dimensional homogeneous point (x, y, 1). After transformation we have the point (x', y', w'). The *real-world* 2D coordinates are obtained by dividing the x- and y-components by the w-component.

2 3D Transformations

2.1 Introduction

We have already looked at the two-dimensional transformations which are used to manipulate pictures. Equivalent transformations are needed to manipulate three-dimensional pictures in three-dimensional space. However, not only are transformations useful as a tool for creating and subsequently altering a picture, they can also help us to visualise the three-dimensional shape of the resulting picture. We would examine an unfamiliar object by picking it up and turning it round to look at it from above, below and from the side, or by holding it at arm's length or standing back from it. In the same way, transformations can be used to rotate, translate or scale a picture to obtain an understanding of its shape. This is particularly important in computer graphics in which the medium for displaying pictures is the two-dimensional display screen on which depth information may not be obvious.

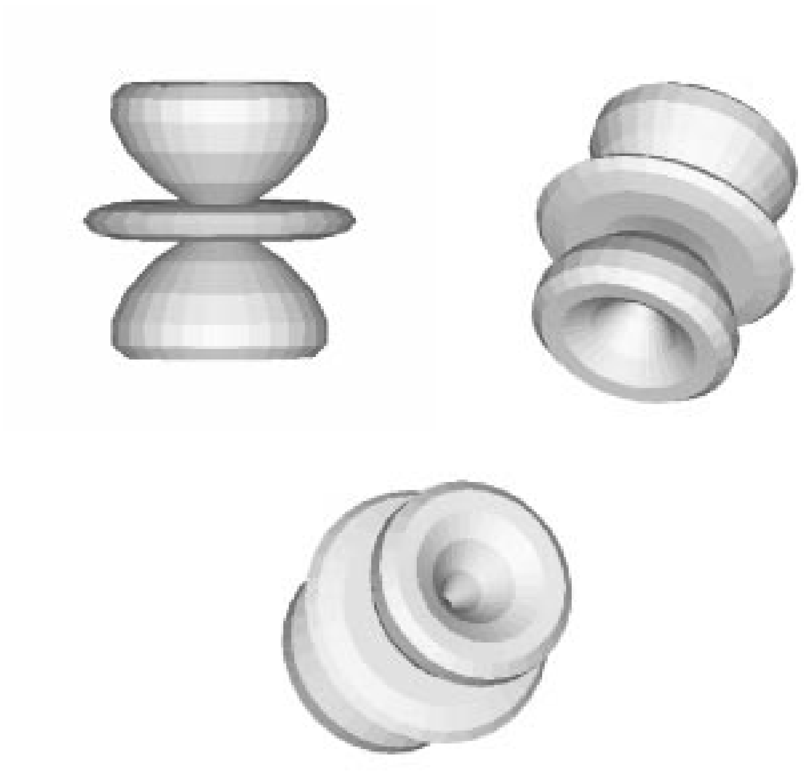


Figure 14: investigating the three-dimensional shape of a surface

In Figure 14 a complex surface is displayed showing the same surface rotated in different ways. Only by looking at a number of these different views can we begin to understand the 3D shape of the surface. These notes will develop techniques for expressing three-dimensional transformations by extending the two-dimensional techniques already presented. Right-handed coordinate systems will be used, as shown in Figure 15.

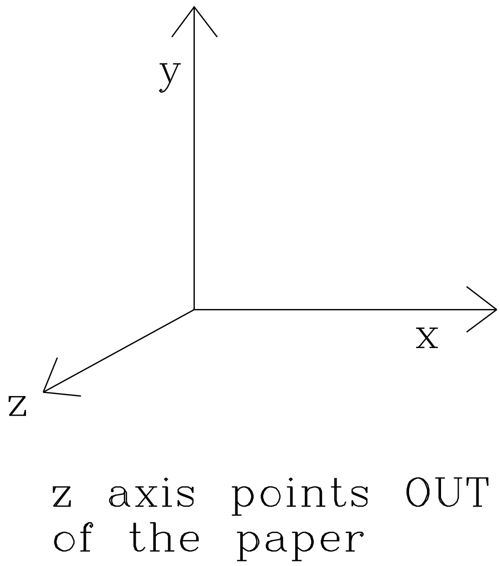


Figure 15: a right-handed coordinate system

2.2 Homogeneous Coordinates

Based on our experience in 2D, we immediately introduce homogeneous coordinates so that a point in 3D space (x, y, z) is represented by a four-dimensional position vector (x, y, z, w). This point may then be transformed by the following matrix operation:

$$\begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}$$

In order to obtain the 3D coordinates from the transformed homogeneous point, we divide the x-, y- and z-components by the w-component:

$$x'' = x'/w'$$
$$y'' = y'/w'$$
$$z'' = z'/w'$$

2.3 Types of 3D Transformation

2.3.1 Translation

The matrix to perform 3D translation is shown below

$$\begin{bmatrix} 1 & 0 & 0 & d \\ 0 & 1 & 0 & h \\ 0 & 0 & 1 & l \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Matrix element d is the displacement added to the x-coordinate, h is the displacement added to the y-coordinate, and l is added to the z-coordinate.

2.3.2 Scaling

3D scaling is performed by the elements on the main diagonal of the matrix:

$$\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & k & 0 \\ 0 & 0 & 0 & p \end{bmatrix}$$

The x-, y- and z-scale factors are given by a, f and k respectively. The element p provides overall scaling by a factor of $1/p$.

2.3.3 Rotation

The terms in the upper-left 3×3 component matrix control 3D rotation:

$$\begin{bmatrix} a & b & c & 0 \\ e & f & g & 0 \\ i & j & k & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The basic 2D rotation was a rotation about the origin in the xy-plane. There are three basic 3D rotations: rotation about the x-axis, rotation about the y-axis and rotation about the z-axis. These are illustrated in Figure 16.

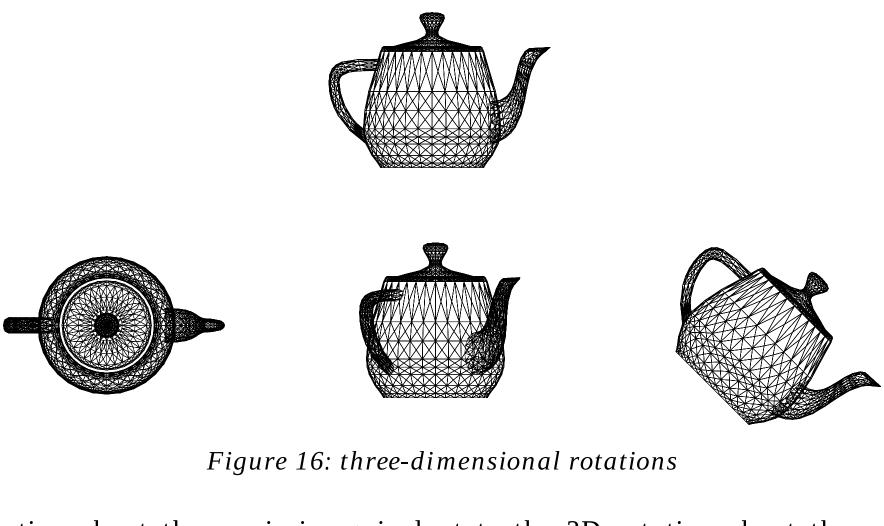


Figure 16: three-dimensional rotations

A rotation about the z-axis is equivalent to the 2D rotation about the origin. Hence we can write down the x and y terms of the matrix straight away. Since we are rotating about the z-axis, the z-coordinate should not be changed, and so the z-row and column should both be [0 0 1 0] (as in the identity matrix). Similarly for rotation about x and y, the row and column corresponding the axis of rotation are taken from the identity matrix. The cosine and sine terms are then used to fill the remaining elements of the 3×3 component matrix. The matrices for rotation about the three axes are:

- Rotation about the x-axis: $$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
- Rotation about the y-axis: $$\begin{bmatrix} \cos\alpha & 0 & \sin\alpha & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\alpha & 0 & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
- Rotation about the z-axis: $$\begin{bmatrix} \cos\alpha & -\sin\alpha & 0 & 0 \\ \sin\alpha & \cos\alpha & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2.3.4 Shearing

The off-diagonal elements in the upper-left 3×3 component matrix produce 3D shearing effects:

$$\begin{bmatrix} 1 & b & c & 0 \\ e & 1 & g & 0 \\ i & j & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

In 3D a shear in x may be obtained as a function of the y- and z-coordinates. This is controlled by matrix elements b and c respectively. Similarly elements e and g control shearing in y as a function of x and z and elements i and j control shearing in z as a function of x and y.

2.3.5 Rotation about an arbitrary axis

Having looked at the basic 3D transformations we will now look at a more complex example involving a combination of these transformations. A common requirement is to rotate an object about an arbitrary axis rather than one of the coordinate axes. In the notes on 2D transformations the transformation for rotating about an arbitrary point was derived. This involved shifting the object and point so that the point coincides with the origin, a rotation about the origin and finally a second shift which is the inverse of the first to place the object back in its original position. Similarly a 3D rotation about an arbitrary axis involves transforming the object and axis of rotation so that the axis coincides with one of the coordinate axes, followed by a rotation about the coordinate axis and finishing with a transformation which is the inverse of the first. The individual steps are as follows:

- Translate so that axis of rotation passes through the origin.
- Rotate object so that axis of rotation coincides with one of the coordinate axes.
- Perform the specified rotation about appropriate coordinate axis.
- Apply inverse rotations to bring axis of rotation back to original orientation.
- Apply inverse translation to bring rotation axis back to original position.

For more details of this transformation see Hearn and Baker[1]. This derives the rotations required to orientate the axis of rotation so that it is parallel to one of the coordinate axes.

2.4 Perspective Transformations

All of the 3D transformation matrices examined so far have been of the form

$$\begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ 0 & 0 & 0 & p \end{bmatrix}$$

i.e. elements m, n and o are equal to zero. This section looks at the effect achieved when one or more of these values is non-zero. Consider the following transformation

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

After homogeneous division the real 3D coordinates of the transformed point are (x'', y'', z'') where

$$\begin{aligned} x'' &= x/(\gamma \cdot z + 1) \\ y'' &= y/(\gamma \cdot z + 1) \\ z'' &= z/(\gamma \cdot z + 1) \end{aligned}$$

As the original z-coordinate tends to infinity

$$\begin{aligned} x'' &\to 0 \\ y'' &\to 0 \\ z'' &\to 1/\gamma \end{aligned}$$

Hence, after transformation lines originally parallel to the z-axis will appear to pass through the point (0, 0, $1/\gamma$), known as the *vanishing point*. This kind of transformation is known as a *perspective* transformation and is illustrated in Figure 17. A house aligned with the x-, y- and z-axes is shown before and after a perspective transformation.

A perspective transformation has a distorting effect which gives the transformed object a natural appearance, similar to that which would be seen by the eye from the point (0, 0, $-1/\gamma$). The eye point is often referred to as the *centre of projection*.

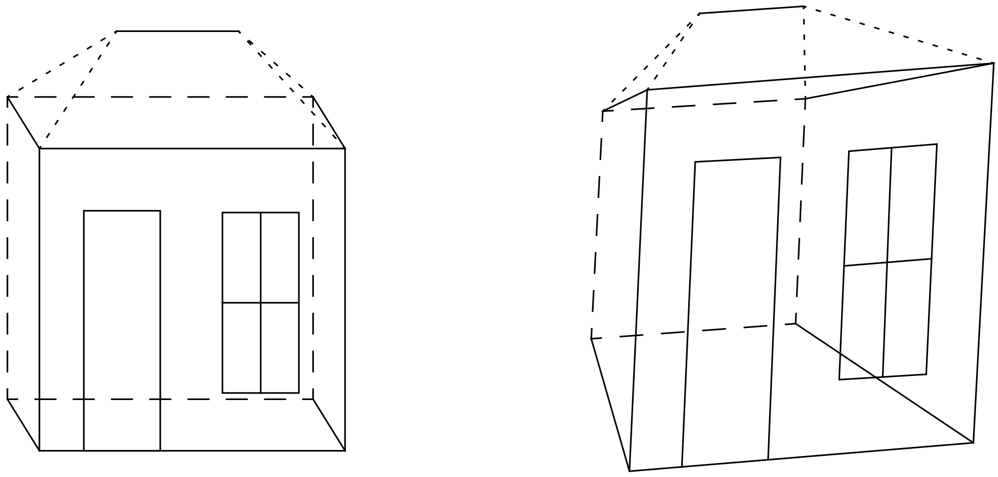


Figure 17: a perspective transformation

Different types of perspective transformation are obtained if the other two elements on the bottom row are set. For example, the matrix

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

would create a perspective transformation with a vanishing point for lines originally parallel to the x-axis at ($1/\alpha$, 0, 0) and a centre of projection at ($-1/\alpha$, 0, 0). Similarly the matrix

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

would create a perspective transformation with a vanishing point for lines originally parallel to the y-axis at (0, $1/\beta$, 0) and a centre of projection at (0, $-1/\beta$, 0). These two cases are illustrated in Figure 18.

Perspective transformations with only one vanishing point are known as one point perspective transformations. If two or three of the matrix elements are non-zero together, a two or three point perspective transformation is obtained, as shown in Figure 19.

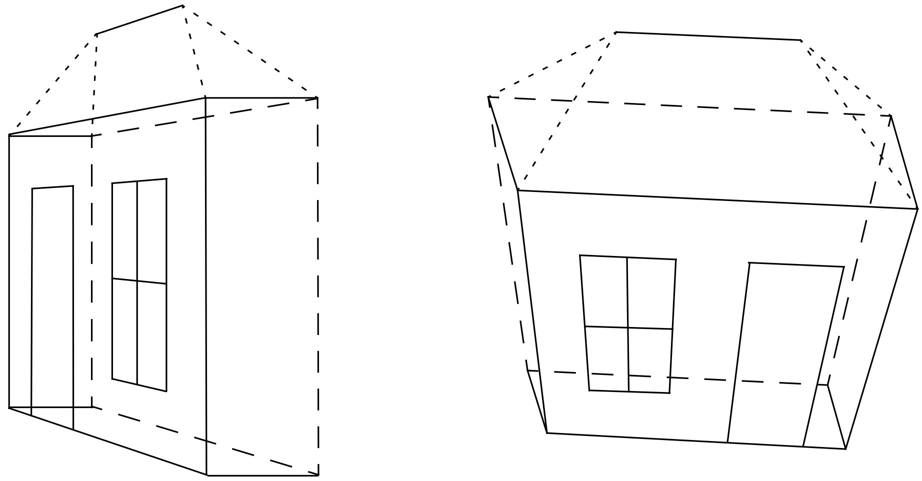


Figure 18: one point perspective transformations

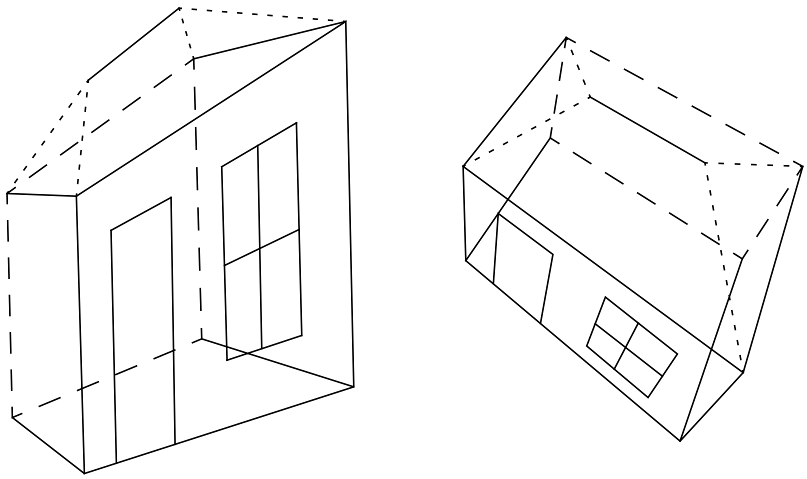


Figure 19: two and three point perspective transformations

2.4.1 Points behind the eye point

The following shows a perspective transformation applied to the point (x, y, z) with the eye point (centre of projection) at (0, 0, c):

$$\begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1/c & 1 \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ (c-z)/c \end{bmatrix}$$

The value of w' varies depending on the value of the original z-coordinate as shown below

$$z < c \rightarrow w' > 0$$
$$z = c \rightarrow w' = 0$$
$$z > c \rightarrow w' < 0$$

This is illustrated in Figure 20.

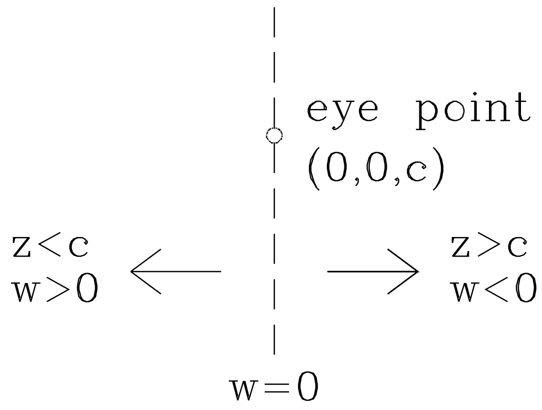


Figure 20: variation of w' with the original z-coordinate in a perspective transformation

Now consider a line which joins the points $\mathbf{p}_1$ and $\mathbf{p}_2$ which are positioned on either side of the eye point ($\mathbf{p}_1$ has $z < c$ and $\mathbf{p}_2$ has $z > c$). After transformation $\mathbf{p}_1$ will have a positive w value and $\mathbf{p}_2$ will have a negative w value. Figure 21 shows the transformed line in homogeneous coordinate space (for simplicity, only the xw-plane is drawn). Homogeneous division then projects the transformed points $\mathbf{p}_1'$ and $\mathbf{p}_2'$ onto the $w = 1$ plane. The resulting line however is not the line joining the projected points $\mathbf{p}_1''$ and $\mathbf{p}_2''$. In Figure 22 other points along the line $\mathbf{p}_1'\mathbf{p}_2'$ are projected onto the $w = 1$ plane. It can be seen from this that the projected line is actually in two parts: $\mathbf{p}_1''$ to positive infinity and negative infinity to $\mathbf{p}_2''$. These are known as *external line segments* and is the correct interpretation of the application of a perspective transformation to a line joining points on either side of the eye point. More details can be found in Blinn and Newell[2].

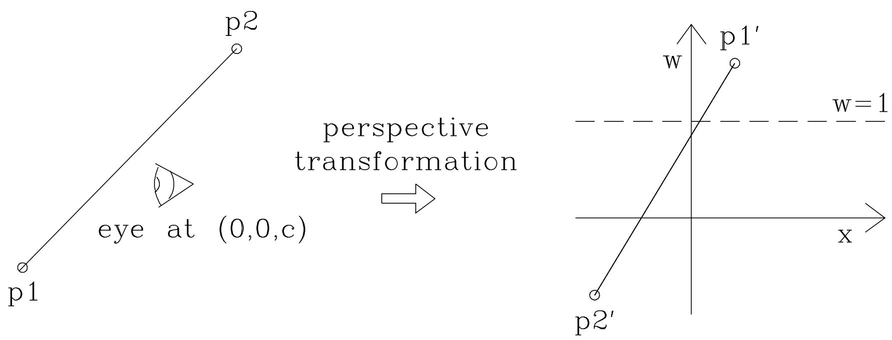


Figure 21: perspective transformation of the line $\mathbf{p}_1\mathbf{p}_2$

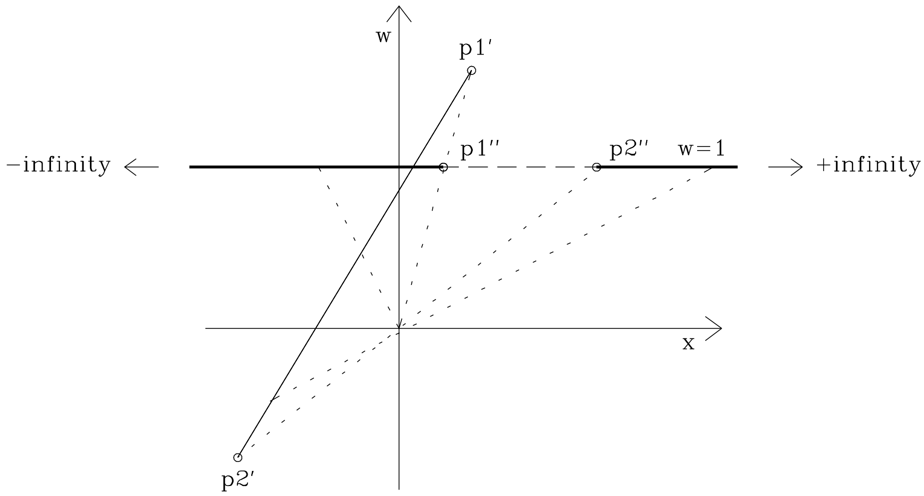


Figure 22: projecting line $\mathbf{p}_1'\mathbf{p}_2'$ *onto the w = 1 plane*

2.4.2 Clipping

In general we want the results of a perspective transformation to simulate what the eye would actually see. Since the eye can only see objects in front of it, items behind the eye should be clipped out. This can be done in one of two ways:

- Clip before the perspective transformation by removing all those parts with $z \geq c$.
- Clip after perspective transformation but *before* homogeneous division, in this case remove parts with $w \leq 0$.

After the homogeneous division it is impossible to distinguish between points which were originally behind and in front of the eye.

2.5 Projections

The 3D transformations examined so far have transformed one 3D object into another 3D object. There is another class of transformations which transform a 3D object into a 2D object by *projecting* it onto a plane. These transformations, known as *projections*, are of particular interest in computer graphics in which the finished picture must always be projected onto the flat viewing surface of a graphics display. Projecting a three dimensional object onto a plane is similar to casting a shadow of the object onto a flat surface. Figure 23 illustrates two cases, one where the light source is a finite distance from the object, and the other where the light source is a long distance from the object, so that the light rays hitting the object are approximately parallel. Similarly there are two types of projection. A perspective projection in which the centre of projection is a finite distance from the object, and a parallel projection in which the centre of projection is a long distance from the object, so that the projectors hitting the object are approximately parallel. Rather than a light source, a projection is identified by the centre of projection, projectors replace rays of light.

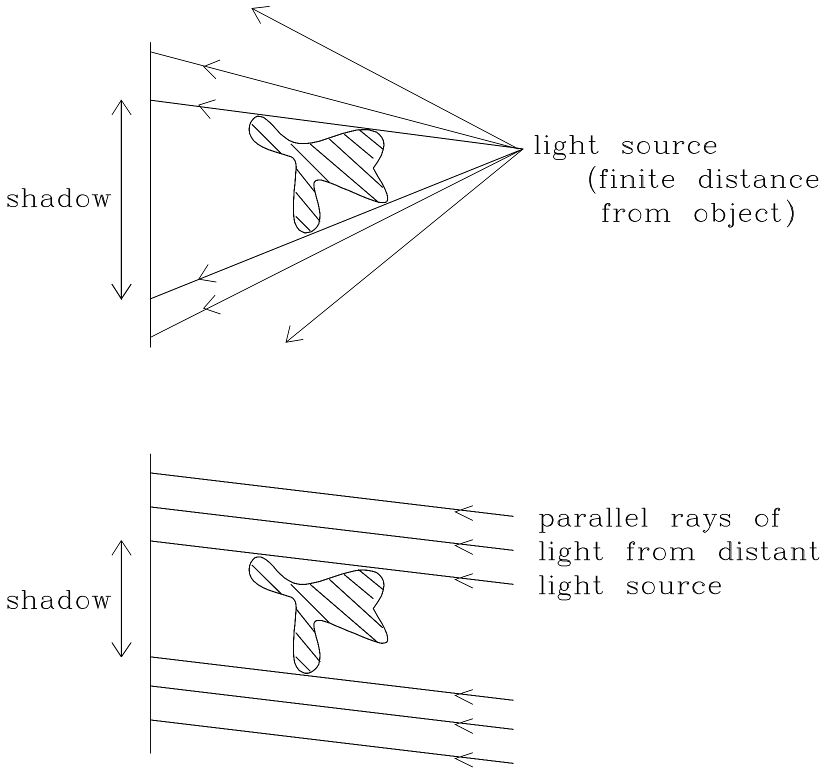


Figure 23: casting shadows

2.5.1 Perspective projections

A perspective projection is shown in Figure 24. It is defined by a centre of projection and a view plane onto which a 2D image of the 3D object is projected. The resulting image is the same as if the eye had been placed at the centre of projection. Figure 25 illustrates a perspective projection in which the centre of projection is the point (0, 0, c) and the view plane is the plane $z = 0$. The point **p** is projected onto the point **p**' by drawing a projector from the centre of projection, **e**, through **p** onto the view plane. The point at which the projector intersects the view plane is **p**'. From the figure, similar triangles **ep**'**b** and **pp**'**a** can be used to find the coordinates of **p**':

$$x'/c = (x' - x)/z$$

so that

$$x' = x/(1 - z/c)$$

Similarly,

$$y' = y/(1 - z/c)$$

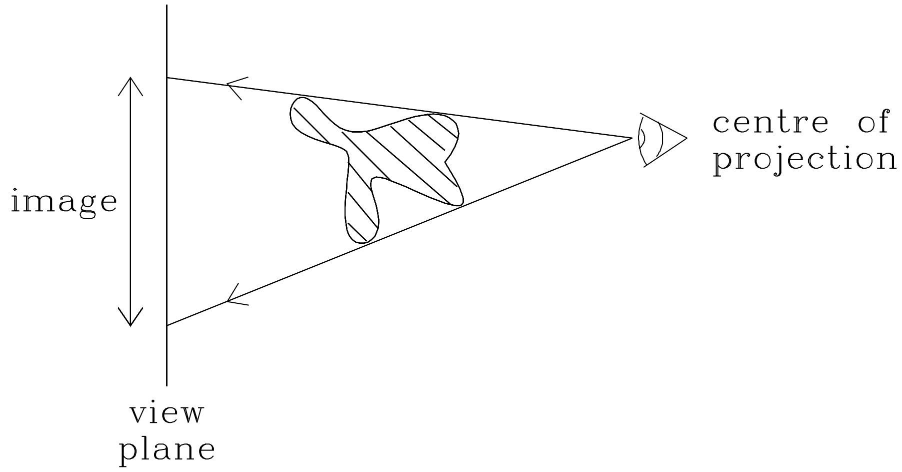


Figure 24: a perspective transformation

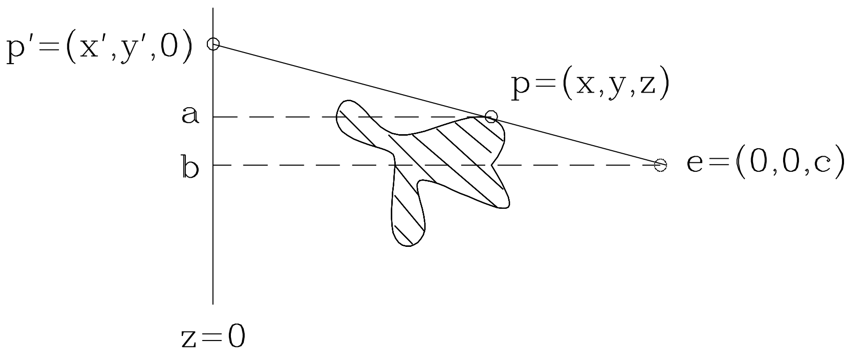


Figure 25: another perspective projection

We can see from this that the effect of a perspective projection on the *x*- and *y*-coordinates is the same as that of a perspective transformation. Since the view plane is the plane $z = 0$, all *z*-coordinates will be mapped to zero.

2.6 Parallel Projections

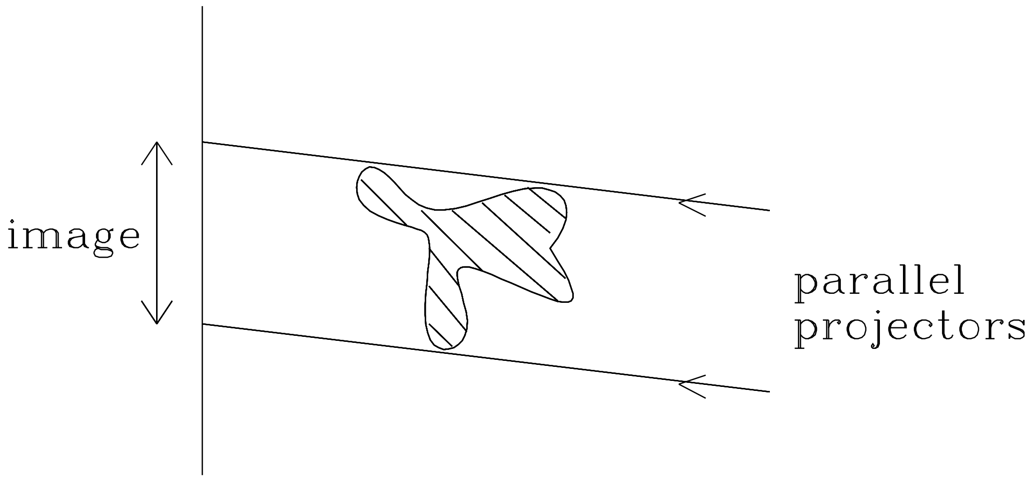


Figure 26: a parallel projection

A parallel projection is shown in Figure 26. This can be described as a perspective projection in which the centre of projection has been moved to infinity. In this case, rather than a centre of projection, the direction of the parallel projectors defines the projection together with the view plane. Although parallel projections tend to produce less realistic views of an object, they are useful as they usually maintain some of the original size information about the original object and can therefore be used for measurements.

2.7 Classification of Planar Geometric Projections

There are many different types of parallel and perspective projections. These are classified in Figure 27. Each type can be obtained by choosing a suitable view plane and centre of projection. This section gives a brief description of the different types of projection, how they are obtained and where they might be used. A thorough review can be found in Carlbom and Paciorek[3].

- **Parallel** Obtained when the centre of projection is at infinity, the projectors are always parallel. Parallel projections are divided into:
  - **Orthographic** The projectors are perpendicular to the view plane. There are two different types of parallel orthographic projection:
  - Multiview orthographic The view plane is parallel to one of the principal planes of the object. Usually a number of views are shown together. The exact shape of one face in the object is retained but it is usually hard to visualise the three dimensional shape from the projection. This type of projection is typically used in engineering drawings.
  - **Axonometric** The view plane is chosen so that the projection will illustrate the general three dimensional shape of the object. Parallel lines are equally foreshortened and remain parallel. Uniform foreshortening occurs along principal axes enabling measurements to be taken to scale along these axes. Axonometric projections are classified according to the angles made between the principal axes and the view plane:

- **Trimetric** The angles between the principal axes and the view plane are all different. This means that the foreshortening along each axis will be different.
- **Dimetric** Two of the angles between the principal axes and the view plane are equal and therefore foreshortening along two axes will be the same.
- **Isometric** All of the angles between the principal axes and the view plane are equal and therefore foreshortening along all three axes will be the same.

• **Oblique** The projectors are NOT perpendicular to the view plane. Usually the view plane is positioned parallel to the principal face of the object so that this face is projected without distortion. This allows direct measurements to be made of this face. The angle between the projectors and the view plane is chosen so as to best illustrate the third dimension, two examples are:

- **Cavalier** The angle is 45° - faces parallel and perpendicular to the view plane are projected at their true size.
- **Cabinet** The angle is arccot(0.5) (approximately 64°) - faces parallel to the view plane are projected at true size, and faces perpendicular to the view plane are projected at half size.

• **Perspective** Obtained when the centre of projection is a finite distance from the object, the projectors emanate from the centre of projection. Perspective projections are classified according to the number of vanishing points in the projection.

- **One point** When only one of the principal axes intersects the view plane.
- **Two point** When exactly two of the principal axes intersect the view plane.
- **Three point** When all three principal axes intersect the view plane.

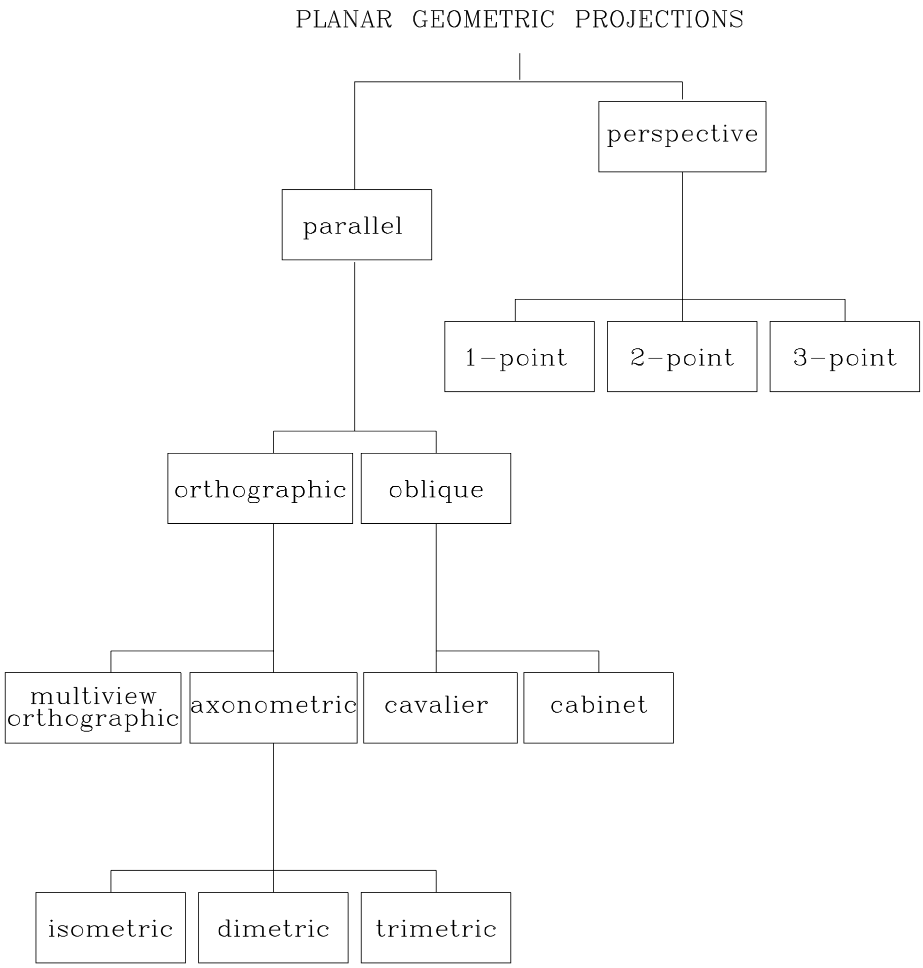


Figure 27: classification of the planar geometric projections

A References

1. Hearn D. and Baker M.P., *Computer Graphics*, Prentice-Hall 1986.

2. Blinn J.F. and Newell M.E., *Clipping Using Homogeneous Coordinates*, Proceedings of SIGGRAPH '78, pp. 245-251.

3. Carlbom I.B. and Paciorek J., *Planar Geometric Projections and Viewing Transformations*, Computing Surveys, Vol. 10 (December 1978), pp. 465–502.

4. Singleton K.M., *An Implementation of the GKS-3D/PHIGS Viewing Pipeline*, Proceedings of Eurographics 1986, North-Holland, Amsterdam.

5. Information Processing Systems - Computer Graphics - Graphical Kernel System for Three Dimensions (GKS-3D) Functional Description ISO DIS 8805.

6. Information Processing Systems - Computer Graphics - Programmer's Hierarchical Interactive Graphics System (PHIGS) ISO DIS 9592.

7. Foley J.D., van Dam A., Feiner S.K. and Hughes J.F., *Computer Graphics: Principles and Practice* (2nd Ed.), Addison Wesley 1990.

These materials have been produced as part of the Information Technology Training Initiative, funded by the Information Systems Committee of the Higher Education Funding Councils.

For further information on this and other modules please contact:

The ITTI Gravigs Project, Computer Graphics Unit, Manchester Computing Centre.
Tel: 0161 275 6095 Email: *gravigs@mcc.ac.uk*

To order additional copies of this or other modules please contact:

Mrs Jean Burgan, UCoSDA.
Tel:0114 272 5248 Email: *j.burgan@sheffield.ac.uk*

ISBN 1 85889 059 4