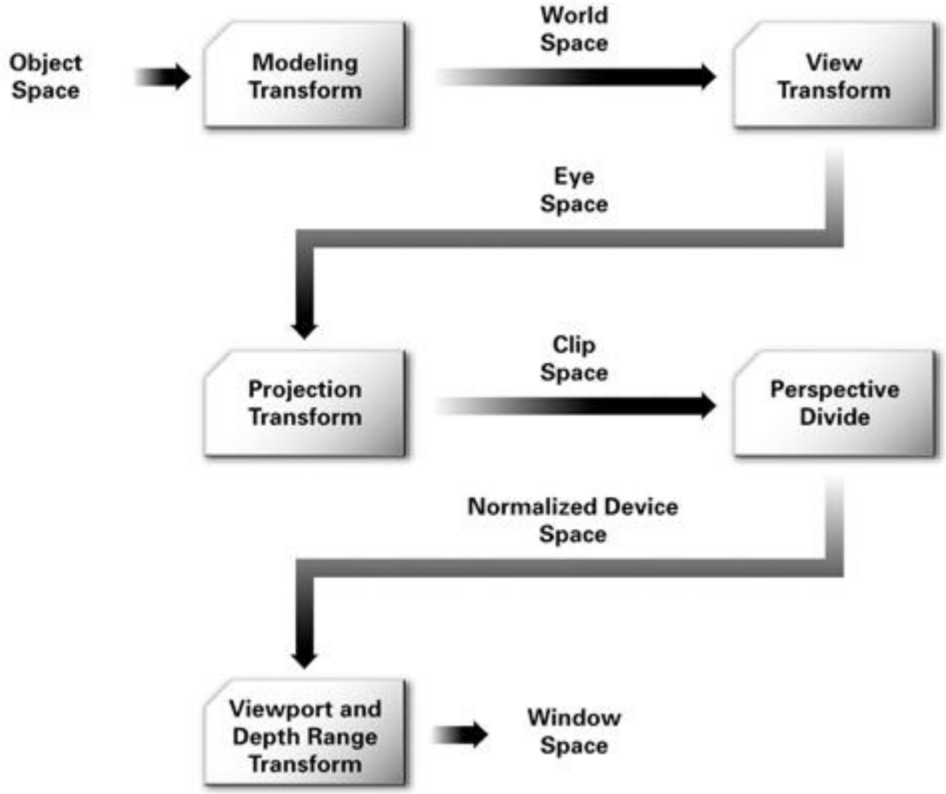
COORDINATE SYSTEMS, FRAMES AND AFFINE TRANSFORMERS

4.2 Coordinate systems and frames

- The purpose of the graphics pipeline is to create images and display them on your screen. The graphics pipeline takes geometric data representing an object or scene (typically in three dimensions) and creates a two-dimensional image from it.
- Your application supplies the geometric data as a collection of vertices that form polygons, lines, and points.
- The resulting image typically represents what an observer or camera would see from a particular vantage point.
- As the geometric data flows through the pipeline, the GPU's vertex processor transforms the constituent vertices into one or more different coordinate systems, each of which serves a particular purpose. Cg vertex programs provide a way for you to program these transformations yourself.

Figure 4-1 illustrates the conventional arrangement of transforms used to process vertex positions. The diagram annotates the transitions between each transform with the coordinate space used for vertex positions as the positions pass from one transform to the next.



4.4 Affine transformations

An affine transformation is an important class of linear 2-D geometric transformations which maps variables (*e.g.* pixel intensity values located at position (x_1, y_1) in an input image) into new variables (*e.g.* (x_2, y_2) in an output image) by applying a linear combination of translation, rotation, scaling and/or shearing (*i.e.* non-uniform scaling in some directions) operations.

The general affine transformation is commonly written in homogeneous coordinates as shown below:

$$\begin{vmatrix} x_2 \\ y_2 \end{vmatrix} = A \times \begin{vmatrix} x_1 \\ y_1 \end{vmatrix} + B$$

By defining only the *B* matrix, this transformation can carry out pure translation:

$$A = \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix}, B = \begin{vmatrix} b_1 \\ b_2 \end{vmatrix}$$

Pure rotation uses the *A* matrix and is defined as (for positive angles being clockwise rotations):

$$A = \begin{vmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{vmatrix}, B = \begin{vmatrix} 0 \\ 0 \end{vmatrix}$$

Here, we are working in image coordinates, so the y axis goes downward. Rotation formula can be defined for when the y axis goes upward.

Similarly, pure scaling is:

$$A = \begin{vmatrix} a_{11} & 0 \\ 0 & a_{22} \end{vmatrix}, B = \begin{vmatrix} 0 \\ 0 \end{vmatrix}$$

(Note that several different affine transformations are often combined to produce a resultant transformation. The order in which the transformations occur is significant since a translation followed by a rotation is not necessarily equivalent to the converse.)

Since the general affine transformation is defined by 6 constants, it is possible to define this transformation by specifying the new output image locations (x_2, y_2) of any three input image coordinate (x_1, y_1) pairs. (In practice, many more points are measured and a least squares method is used to find the best fitting transform.)