International Journal of Research (IJR) Vol-1, Issue-9 October 2014 **ISSN 2348-6848**

3-Dimensional Transformation

Priyanka Sahni & Nonika Sharma

Information Technology, Dronacharya College of Engineering, Gurgaon, Haryana, India

priyanka.sahni@yahoo.com.au, Nonikasharma1@gmail.com

Abstract –

*Transformation is often used to work on vertices and vectors. Transformation is performed by the use of multiplication with a matrix. There are characteristically three types of primal transformations that can be performed on vertices: translation, rotation and scaling. In addition to these, projection transformation is used to go from view space to projection space. The D3DX library contains APIs that can handily construct a matrix for many purposes such as translation, rotation, scaling , world-to-view transformation , view-to-projection transformation, etc. .Similar to 2D transformations, which used 3*3 matrices,3D transformations used 4*4 matrices(X,Y,Z,W).*

3D Translation – It refers to moving or displacing a certain distance in space.

3D Rotation – For rotation we rotate about an axis .This can be done along the x axis , y-axis and z-axis .

3D Scaling –It refers to magnifying or shrinking the size of vector components along axis directions.

Introduction:

3D graphics are graphics that use a 3D representation of geometric data. For the reason of performance this is stored in the computer. This includes images that may be for later display or for real-time viewing.

In spite of these differences, 3D computer graphics depend upon similar algorithms as 2D computer graphics do in the frame and raster graphics (like in 2D) in the final rendered display. In computer graphics software, the distinction between 2D and 3D is infrequently blurred; 2D applications may use 3D techniques to achieve effects such as lighting, and chiefly 3D may use 2D depiction techniques.

3D computer graphics are the same as 3D models. The model is contained within the graphical data file, apart from the depiction. However, there are differences that include the 3D model is the representation of any 3D object. Until visually displayed a model is not graphic. Due to printing, 3D models are not only restricted to virtual space. 3D rendering is how a model can be displayed. Also can be used in non-graphical computer simulations and calculations.

- ## *3D Translation*

Translation refers to moving or displacing for a certain distance in space. In 3D, the matrix used for translation has the form

1 0 0 0
0 1 0 0
0 0 1 0
a b c 1

where(a, b, c) is the vector that defines the direction and distance to move. For example, to move a vertex -5 unit along the X axis (negative X direction), we can multiply it with this matrix:

1 0 0 0
0 1 0 0
0 0 1 0
-5 0 0 1

If we apply this to a cube object centered at origin, the result is that the box is moved 5 units towards the negative X axis, as figure 5 shows, after translation is applied.

- ***The effect of translation***

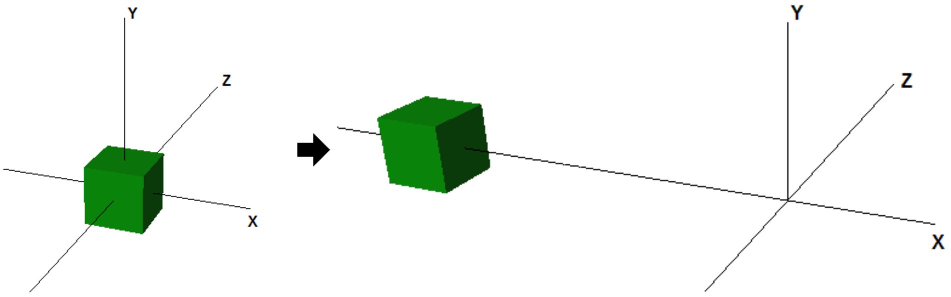

FIG1:SHOWING THE EFFECT OF TRANSLATION

In 3D, a space is typically defined by an origin and three sole axes from the origin: X, Y and Z. There are several spaces commonly used in computer graphics: object space, world space, view space, projection space, and screen space.

- ***3D Rotation***

Rotation refers to rotating vertices about an axis going through the origin. Three such axes are the X, Y, and Z axes in the space. An example in 2D would be rotating the vector [1 0] 90 degrees counter-clockwise. The result from the rotation is the vector [0 1]. The matrix used for rotating? degrees clockwise about the Y axis looks like this:

cos? 0 -sin? 0
0 1 0 0
sin? 0 cos? 0
0 0 0 1

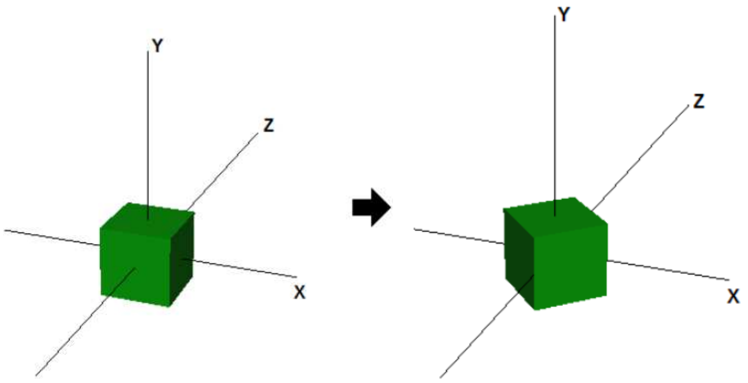

Fig 2: shows the effect of rotating a cube centered at origin for 45 degrees about the Y axis

• *3D Scaling*

Scaling refers to magnifying or shrinking the size of vector components along axis directions. For example, a vector can be scaled up along all directions or scaled down along the X axis only. To scale, we usually apply the scaling matrix below:

p 0 0 0
0 q 0 0
0 0 r 0
0 0 0 1

Where p, q, and r are the scaling factor along the X, Y, and Z direction, respectively. The figure below shows the effect of scaling by 2 along the X axis and scaling by 0.5 along the Y axis.

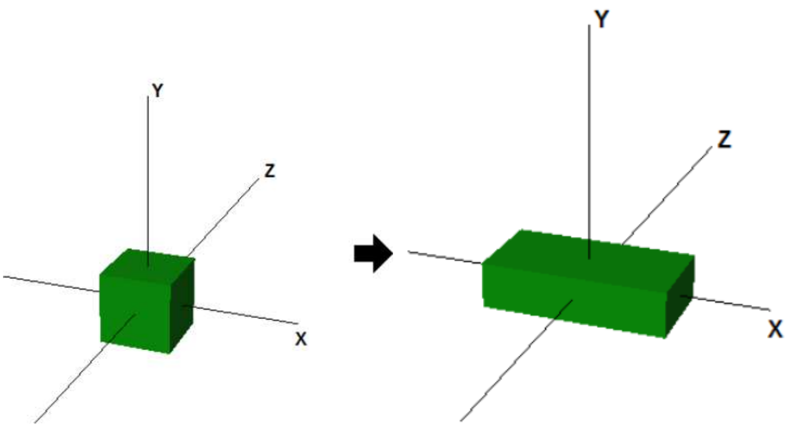

FIG 3:SHOWING THE EFFECT OF SCALING

• *Projection Transformations*

After the viewing transformation we have everything leaning as we would like them to appear in the final image. All that remains is to project out the depth, or z-dimension, so that the the three-dimensional view-space primitives are reduced to two-dimensional screen-space primitives.

There are many different types of projection. The simplest, is to simply ignore the z-dimension. This form of projection is called ***orthographic*** *or* ***parallel****. It is the common form of projection used by drafts people for top, bottom, and side views. The advantage of parallel projection is that you can make accurate measurements of image features in the two dimensions that remain. The disadvantage is that the images don't appear natural.*

The projection matrix for orthographic projection is very simple

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

There are some problems with this simple form, however. To begin with the units of the transformed points are still the same as the model. This is great for drafting, but in our case we'd like for the units to be in pixels.

We can incorporate this change of units, and perform the flip of the y-axis required for raster coordinates into our projection matrix as follows.

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{width}{right-left} & 0 & 0 & \frac{-left \times width}{right-left} \\ 0 & \frac{height}{bottom-top} & 0 & \frac{-top \times height}{bottom-top} \\ 0 & 0 & \frac{z_{max}}{far-near} & \frac{-near \times z_{max}}{far-near} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

For realistic three-dimensional viewing we'd like for our objects to be displayed with proper perspective. Perspective causes objects nearer to the viewing position to appear larger that the same object would when place farther from the viewpoint.

This is an important attribute of lines in projective spaces, they always intersect at a point. The projection matrix for perspective projection matrix is:

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

Notice how similar this transform is to the original parallel projection. It also has all of the disadvantages of the parallel form; its units are not screen space units. The following transformation accomplishes the projection and the conversion to pixels in a single transform.

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{width \times near}{right - left} & 0 & \frac{-left \times width}{right - left} & 0 \\ 0 & \frac{height \times near}{bottom - top} & \frac{-height \times top}{bottom - top} & 0 \\ 0 & 0 & \frac{z_{\max} \times far}{far - near} & \frac{-z_{\max} \times far \times near}{far - near} \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

For the purpose of clipping we can modify this transformation so that it is mapped into a canonical space. Following is a canonical space mapping for orthographic projections.

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{2}{right - left} & 0 & 0 & \frac{-(right + left)}{right - left} \\ 0 & \frac{2}{bottom - top} & 0 & \frac{-(bottom + top)}{bottom - top} \\ 0 & 0 & \frac{2}{far - near} & \frac{-(far + near)}{far - near} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Next is a canonical space mapping for perspective projections.

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{2 \times near}{right - left} & 0 & \frac{-(right + left)}{right - left} & 0 \\ 0 & \frac{2 \times near}{bottom - top} & \frac{-(bottom + top)}{bottom - top} & 0 \\ 0 & 0 & \frac{far + near}{far - near} & \frac{-2 \times far \times near}{far - near} \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

References

[1] Penzien, J., & Watabe, M. (1974). Characteristics of 3-dimensional earthquake ground motions. *Earthquake engineering & structural dynamics*, *3*(4), 365-373.

[2] Prasad, B. V., Prevelige, P. E., Marietta, E., Chen, R. O., Thomas, D., King, J., & Chiu, W. (1993). Three-dimensional transformation of capsids associated with genome packaging in a bacterial virus. *Journal of molecular biology*, *231*(1), 65-74.

[3] Rubin, S. M., & Whitted, T. (1980, July). A 3-dimensional representation for fast rendering of complex scenes. In *ACM SIGGRAPH Computer Graphics* (Vol. 14, No. 3, pp. 110-116). ACM.

[4] Weinberg, M. C. (1991). Surface nucleated transformation kinetics in 2-and 3-dimensional finite systems. *Journal of non-crystalline solids*, *134*(1), 116-122.

[5] Gordon, D., & Anthony Reynolds, R. (1985). Image space shading of 3-dimensional objects. *Computer Vision, Graphics, and Image Processing*, *29*(3), 361-376.