Perspective Cube

A three-dimensional Visio 4.0 Shape

by *Michael A. Rundel*

1. Introduction - Different Projections of a cube

If someone is asked to sketch a cube on a peace of paper most people draw a parrallel projection of a cube (Figure 1-1). This is the representation of a cube most people are familiar with. We choose this projection, not because it is realistic, but because it is easy to sketch. Figure 1-2 shows a perspective projection of a cube, using 2 vanishing points. I've learned in school how to construct such perspective projections, given the measures of an object, a view plane and an eye point. Although this is a realistic representation of a cube, it resemlbes a very rare special case (you are looking at the horizont), almoust never found in a natural environemnt. It looks almost 'flat' when displayed against a perspective projection with three vanishing points (Figure 1-3). Since my school days I was fascinated by this projection.

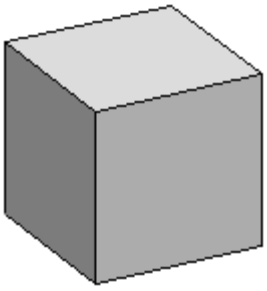

Figure 1-1. parallel projection

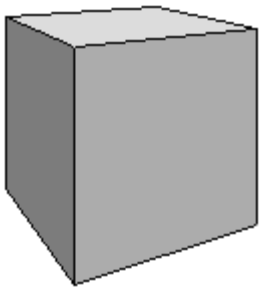

Figure 1-2. perspective projection (2 vanishing points)

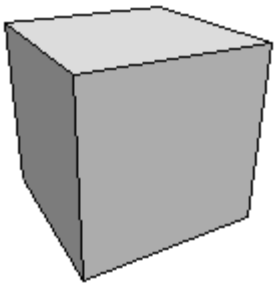

Figure 1-3. perspective projection (3 vanishing points)

2. Projecting Objects onto a view plane

2.1 Viewing System

In our case a simple four-parameter viewing system suffice (Figure 1-4). In this system a user specifies a *view point* (three parameters) and a *view plane (screen) distance* (one parameter). The view coordinate system (x_v, y_v, z_v) is established at the view point. A viewing direction (z_v) is established by the line form the view point to the world coordinate system. x_v beeing perpendicular to z_v and laying on a parallel plane to the x_w/x_w-plane through the view point.

The process of converting a three-dimensional world coordinate space to a two-dimensional screen coordinate system contains a *projective transformation* and a *viewing transformation*.

$$P = P_{world} \cdot T_{view} \cdot T_{pers}$$

2.1 Viewing Transformation T_{view}

First we apply the viewing transformation to all points constituting the object in the world coordinate system. The viewing transformation consists of four basic transformations:

1. A translation of the wolrd coordinate system to the position of the view point,

2. A rotation about the z'-axis, so that the x''-axis is now normal to the plane containing the viewing direction.
3. A rotation about the x''-axis, so that the z'''-axis passes through the origin of the world coordinate system.
4. Convert to a left-handed system.

Multiplying these matrices togehter results in the complete viewing transformation $\boldsymbol{T}_{view}$ (the full derivation of this view matrix can be found in Appendix A of [1]).

$$\begin{aligned} x &= \mu \sin\phi \sin\theta \\ y &= \mu \sin\phi \cos\theta \\ z &= \mu \cos\phi \end{aligned}$$

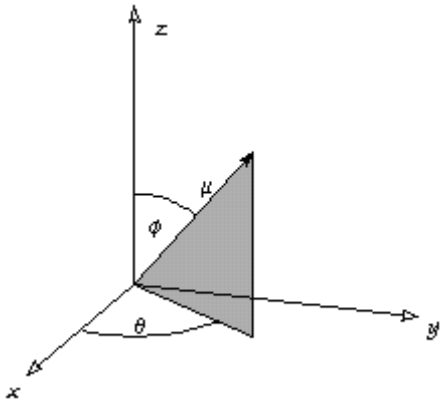

Figure 2-1. Spherical Coordinates

Specifying The view point in spherical coordinates (Figure. 2-1) yields to the following viewing matrix:

$$T_{view} = \begin{pmatrix} -\sin\theta & -\cos\theta\cos\phi & -\cos\theta\sin\phi & 0 \\ \cos\theta & -\cos\phi\sin\theta & -\sin\theta\sin\phi & 0 \\ 0 & \sin\phi & -\cos\phi & 0 \\ 0 & 0 & \mu & 1 \end{pmatrix}$$

2.3 Projective Transformation T_{pers}

To take care of foreshortening we have to apply a projective transformation $\boldsymbol{T}_{pers}$ (derived in [1]):

$$T_{pers} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \frac{1}{d} \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Dividing the view coordinates through the fourth coordinate we get the screen coordinates x_s and y_s:

$$\begin{pmatrix} x_s \\ y_s \end{pmatrix} = \begin{pmatrix} \frac{d(y\cos\theta - x\sin\theta)}{\mu - z\cos\phi - x\cos\theta\sin\phi - y\sin\theta\sin\phi} \\ \frac{d(-(x\cos\theta\cos\phi) - y\cos\phi\sin\theta + z\sin\phi)}{\mu - z\cos\phi - x\cos\theta\sin\phi - y\sin\theta\sin\phi} \end{pmatrix}$$

2.4 Back Face Elimination (Culling)

If you have a convex object (no two surfaces face each other), there is an very easy way to determine if a surface is visible form a given view point or not. Simply calculate the angle between the surface normal ***N*** and the line-of-sight vector ***V*** (the line from the base of the surface normal to the viewpoint). The surface is visible if, and only if, the angle between these two vectors is less than 90 *deg*. That is ***N.V*** > 0 (see [1] for details).

3. Visio 4.0 Shape "Perspective Cube"

3.1 Building the shape

Now that we are familiar with the basics of projections of three-dimensional objects, we are going to make the Visio shape you can see below (Figure 3-1)

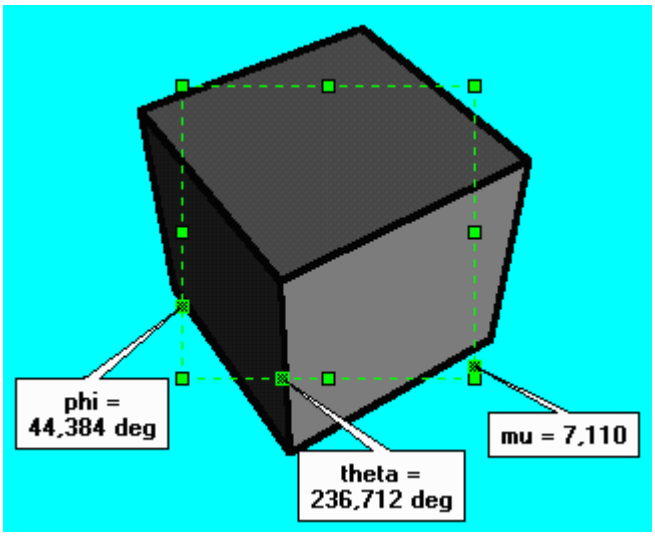

Figure 3-1. visio 4.0 shape
with controls tips

I started with a cube in the world coordinate system given by Figure 3-2. This particular cube was used for two reasons: First the origin of the world coordinate system lays in the center of the cube, thus garanteeing that any changes of the spherical angles will result in a rotation around the cubes center. Second, since all points in the world coordinate systems have eihter one or zero in their vector components the transformation expressions are kept as short as possible.

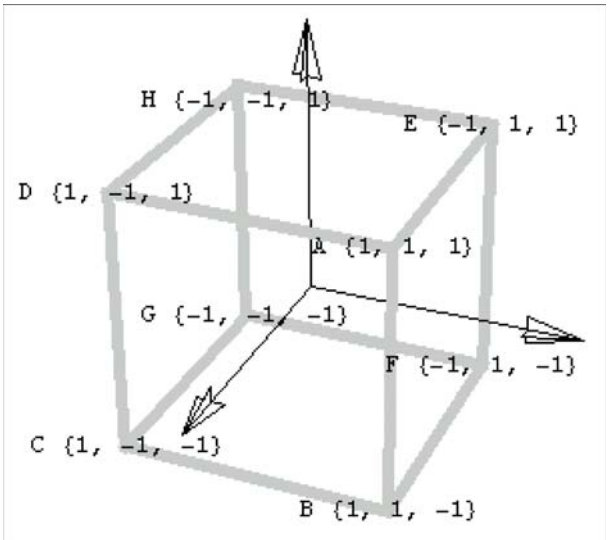

Figure 3-2. wireframe in world coordinates

The following sections give an overview of the shapes construction.

3.2 Group (Parent) Shape

To avoid recalculations the calculations of frequenlty used expressions, like Sine or Cosine of the spheric angles, are done in separate cells.

Cell	Contents
Scratch.C1	theta
Scratch.D1	phi
Scratch.C2	mu
Scratch.D2	view plane distance
Scratch.C3	Sin[theta]
Scratch.D3	Cos[theta]
Scratch.C4	Sin[phi]
Scratch.D4	Cos[phi]
Scratch.D5	scaling factor
Scratch.X1-X8	x-coordinate of Points 1-8
Scratch.Y1-Y8	y-coordinate of Points 1-8
Scratch.A1-A8	denominator of Points 1-8

3.3 Side Sub Shapes

For each side there exists a subshape, each representing one side of the cube. If a side is not visible from a given view point the shape's geometry is simply collapsed to a single point.

For checking visibility I used the midpoint of each surface, which are simple the points (1,0,0), (-1,0,0), (0,1,0),... These are not only the midpoints, but the surface normals as well. Note however, that the viewing transformation includes a conversion to a left-handed coordinate system, which has to be taken into account when calculating visibility.

Surface	Visible if
B-A-E-F	(mu sin(phi) sin(theta) - 1) > 0
C-D-H-G	(-mu sin(phi) sin(theta) - 1) > 0
C-B-A-D	(mu sin(phi) cos(theta) - 1) > 0
G-F-E-H	(-mu sin(phi) cos(theta) - 1) > 0
A-D-H-E	(mu cos(phi) - 1) > 0
C-B-F-G	(-mu cos(phi) - 1) > 0

All subshapes have identical structure:

Cell	Contents
Geometry.X1-X5	x-coordinate of corners
Geometry.Y1-Y8	y-coordinate of corners
Scratch.A1	visibility test

4. Reference

[1] **Watt, Alan**, *3D Computer Graphics*, second edition, Addison-Wesly (1992)