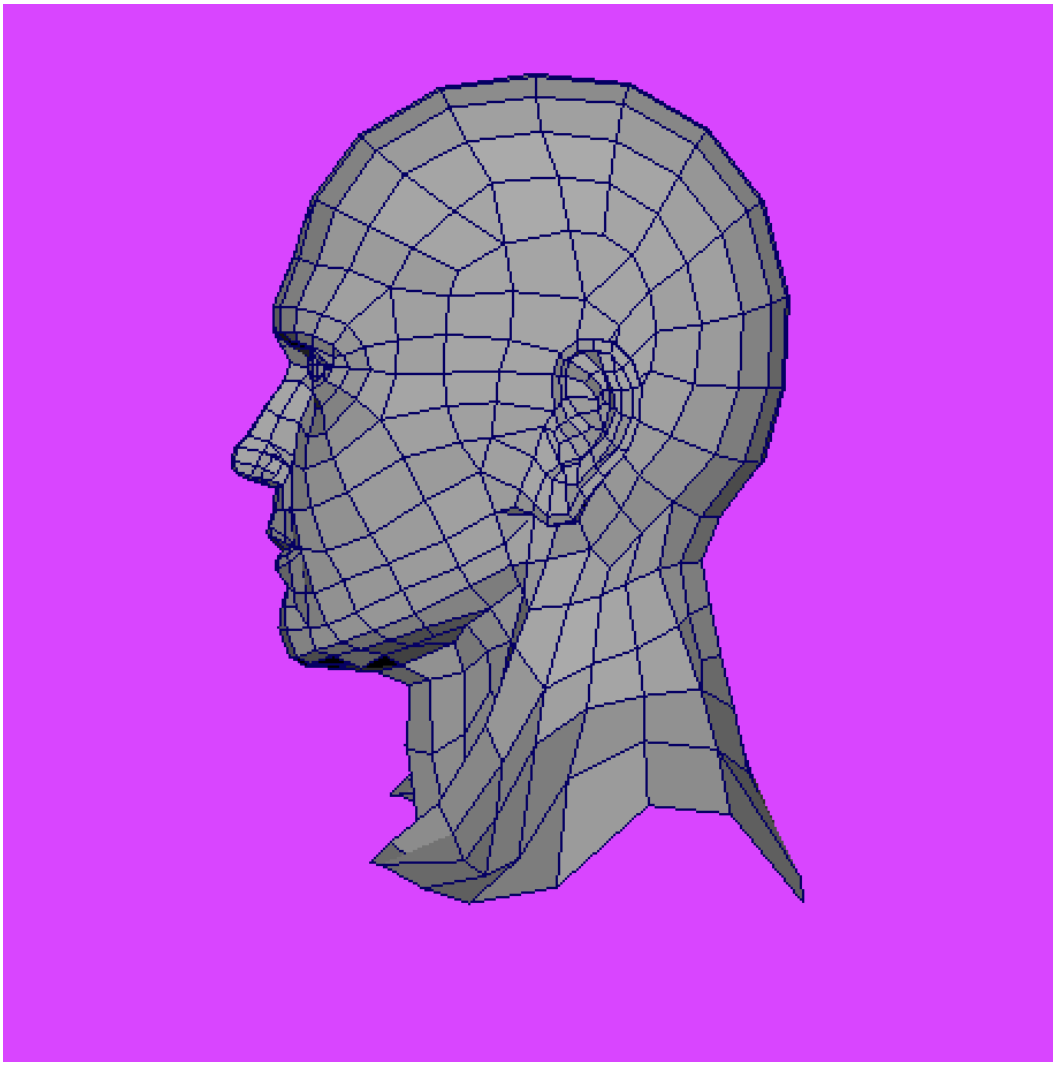


A Demonstration of Operations in Computer Graphics Using Maya

AND THEIR RELATION TO MATRIX MANIPULATION

Tristan Bowler | Math 2270 Linear Algebra | May 2018

Abstract

The usage of linear transformations such as dilation, contraction, rotation, reflection, shear, and projection is vital in the creation and variation of computer graphics in game design. The ability to transform and duplicate game objects is an important part of creating a realistic game universe and the tools of linear algebra and linear transformations are a convenient tool for game designers and animators to accomplish this. The use of such variations on objects saves time and effort on behalf of modelers and programmers because they do not have to create, rig and program actions for an entirely new object for every slightly different instance that they wish to render. Particularly, this applies to linear algebra because each object/ node in a computer graphic is backed with a matrix to allow for easier manipulation and transformations.

BACKGROUND

To illustrate the points of this paper, I will use a program called Maya, which is a 3D-Modeling Software used by animators and game designers to create three dimensional virtual objects. Maya has two underlying scripting languages, MEL and Python. MEL (Maya Embedded Language) is a language that was designed by Autodesk, the creators of Maya, to be used specifically in Maya, but MEL is limited in some respects, so Python was added to the program for varied functionality. Scripting languages in Maya allow users to manipulate objects without having to use strictly GUI operations like push, pull, scale and rotate for every operation they wish to perform, in addition to being able to accomplish some more complex tasks that would take possibly hours to do manually.

Demonstrations

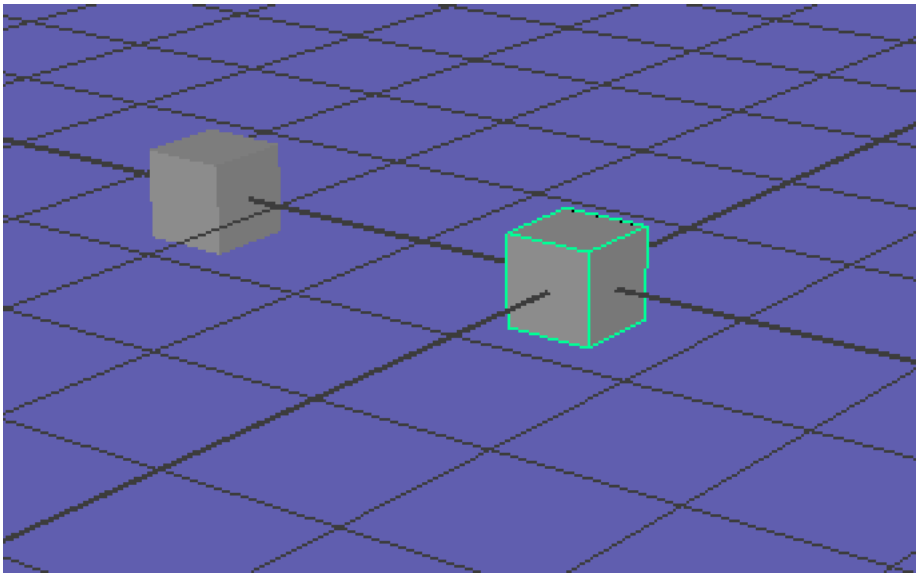
SCALE

Scale is probably the easiest to observe. The command in MEL is fairly simple:

setAttr "cube.scaleX" 2

Where we may scale in the x, y, or z direction by simply changing the line to scaleY or scaleZ respectively. The 2 in this case refers to the value which we would like to scale by.

In this example, we start with two 1x1x1 cubes: one which we will manipulate and one for comparison.



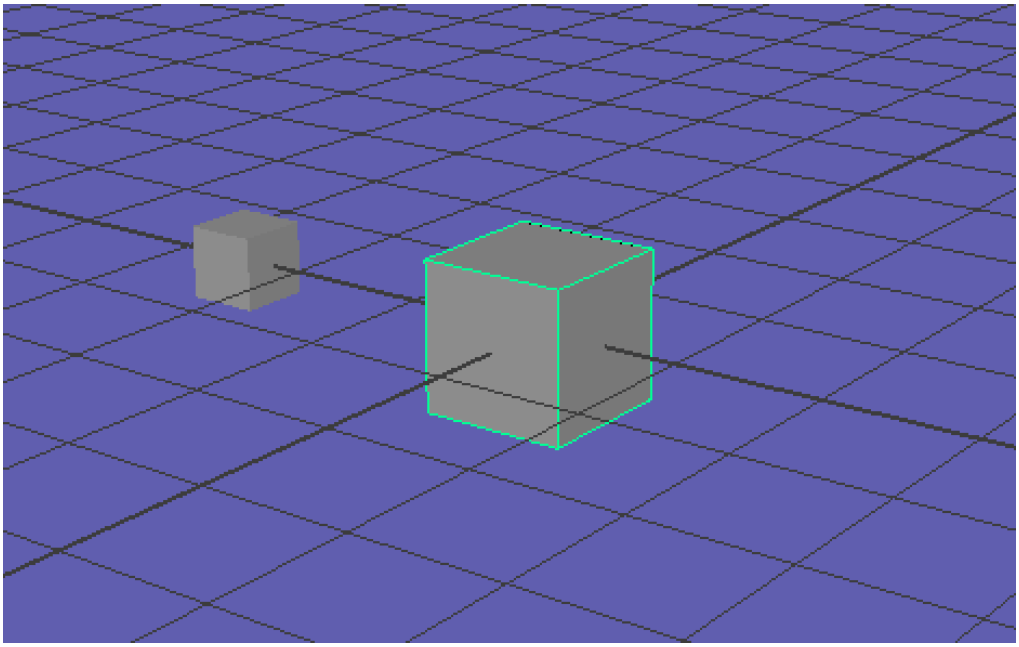
Then we perform the following operations in Maya:

```
//scale by 2 in x,y,z
getAttr cube.matrix;
//Result: 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 //
setAttr "cube.scaleX" 2;
setAttr "cube.scaleY" 2;
setAttr "cube.scaleZ" 2;
getAttr cube.matrix;
//Result: 2 0 0 0 0 2 0 0 0 0 2 0 0 0 0 1 //
//Expected result;
```

The first getAttr command provides us with the matrix to start, which we observe is the 4X4 identity matrix, because the cube is centered on the origin and has not been transformed in any way yet. Then the next three commands scale the cube by two in the x,y and z directions. Finally, we obtain the final matrix which when plugged into Maple appears as:

$$\begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The final cubes appear as follows:



So, we see that the cube has scaled in comparison to its partner. The matrix above makes sense as a result when we consider the following Maple experiment.

We wat to scale by 2 in every direction so that gives us the column vector

$$\begin{bmatrix} 2x \\ 2y \\ 2z \\ 1 \end{bmatrix}$$

And we need a 4x4 matrix such that

$$\begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} 2x \\ 2y \\ 2z \\ 1 \end{bmatrix}$$

We see that the matrix on the left is our result, so this is our transformation matrix, and it is easily checked that this statement is true.

TRANSLATION

Next, we will move on to translation. Again, we start with the two original cubes and do a translation by 5 in the x direction, a translation by 2 in the y direction, and a translation by 1 in the z direction. The MEL code is as follows:

```
//Translation exercise

getAttr cube.matrix;

// Result: 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 //
```

setAttr "cube.translateX" 5;

setAttr "cube.translateY" 2;

setAttr "cube.translateZ" 1;

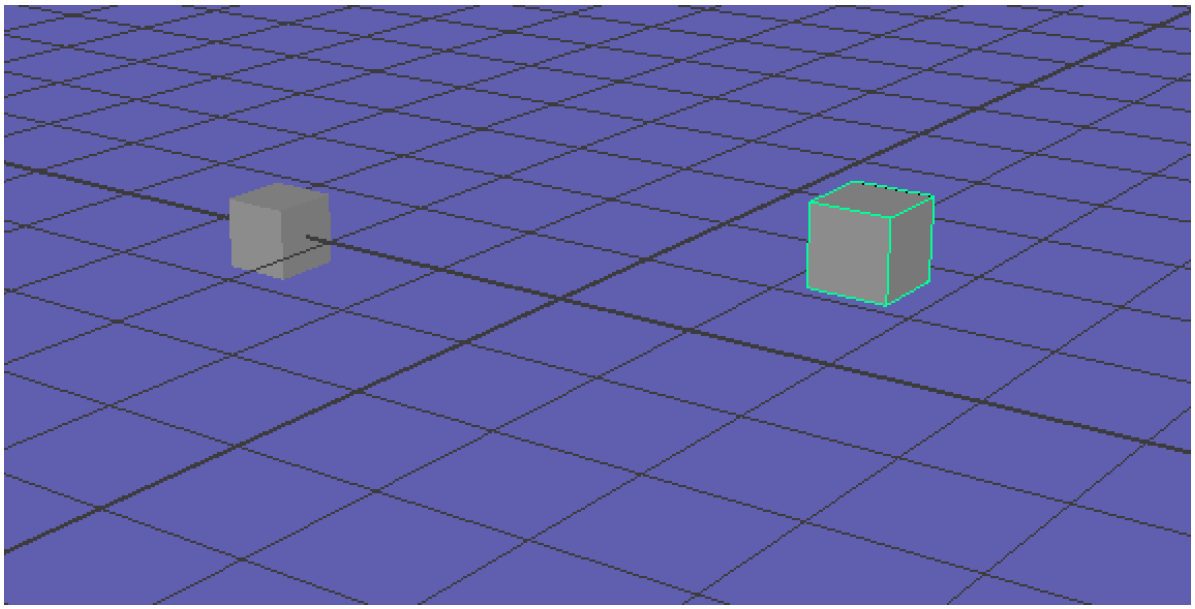
getAttr cube.matrix;

// Result: 1 0 0 0 0 1 0 0 0 0 1 0 5 2 1 1 //

Our final matrix appears as:

$$\begin{bmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

And we obtain the following visual result:



Now when we do a conformation in Maple, we start with the column vector that shows the transformation we wish to make:

$$\begin{bmatrix} x+dx \\ y+dy \\ z+dz \\ 1 \end{bmatrix} = \begin{bmatrix} x+5 \\ y+2 \\ z+1 \\ 1 \end{bmatrix} = \begin{bmatrix} 5 \\ 2 \\ 1 \\ 1 \end{bmatrix}$$

where dz, dy, and dz are the amounts we wish to move by. In this case, because the cube is centered on the origin, the values of x, y and z are all 0, so the vector simplifies to the vector on the right.

Next, we need a 4x4 matrix that satisfies:

$$\begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} 5 \\ 2 \\ 1 \\ 1 \end{bmatrix}$$

It is easily verified by a Maple calculation, or even by hand, that this matrix would be

$$\begin{bmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

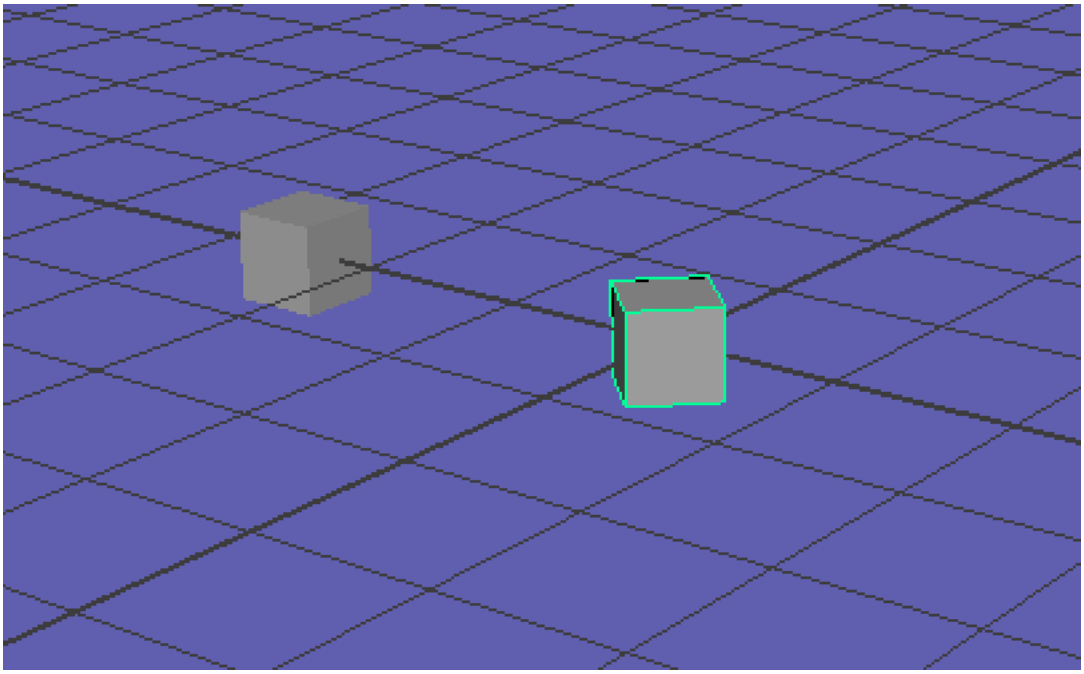
Which is the same as our result from the Maya operations.

ROTATIONS

Finally, we shall demonstrate rotation. Rotations are slightly more complex to combine multiple operations so this example will be more interesting. We shall start with the two 1x1x1 cubes and do the following operations, a rotation by 90 on the x axis and a rotation by 45 on the y axis. The code in MEL goes as follows:

```
setAttr cube.matrix;
// Result: 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 //
setAttr "cube.rotateX" 90;
setAttr "cube.rotateY" 45;
getAttr cube.matrix;
// Result: 0.707107 0 -0.707107 0 0.707107 0 0.707107 0 0 -1 0 0 0 0 0 1 //
```

And we achieve the following visual result:



Next, we need to test this result in Maple. First, we begin with the two forms of Rotation Transformation matrices for the x and y directions for a 4x4 matrix.

We shall call them Rx and Ry respectively:

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

Rx Ry

In this case, we are rotation by 90 deg in the x direction and by 45 deg in the y direction, we know these to be pi/2 rad and pi/4 rad respectively.

If we input these values into Maple we get:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\left(\frac{1}{2}\pi\right) & -\sin\left(\frac{1}{2}\pi\right) & 0 \\ 0 & \sin\left(\frac{1}{2}\pi\right) & \cos\left(\frac{1}{2}\pi\right) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rx(pi/2)

$$\begin{bmatrix} \cos\left(\frac{1}{4}\pi\right) & 0 & -\sin\left(\frac{1}{4}\pi\right) & 0 \\ 0 & 1 & 0 & 0 \\ \sin\left(\frac{1}{4}\pi\right) & 0 & \cos\left(\frac{1}{4}\pi\right) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\sqrt{2} & 0 & -\frac{1}{2}\sqrt{2} & 0 \\ 0 & 1 & 0 & 0 \\ \frac{1}{2}\sqrt{2} & 0 & \frac{1}{2}\sqrt{2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Ry(pi/4)

Next, to get the combined rotations we multiply Rx and Ry

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{2}\sqrt{2} & 0 & -\frac{1}{2}\sqrt{2} & 0 \\ 0 & 1 & 0 & 0 \\ \frac{1}{2}\sqrt{2} & 0 & \frac{1}{2}\sqrt{2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\sqrt{2} & 0 & -\frac{1}{2}\sqrt{2} & 0 \\ -\frac{1}{2}\sqrt{2} & 0 & -\frac{1}{2}\sqrt{2} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

When we put in approximate values, we obtain:

$$\begin{bmatrix} 0.707 & 0 & -0.707 & 0 \\ -0.707 & 0 & -0.707 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

It is immediate that this is the same matrix as our Maya result.

Conclusion

To conclude, the exploration of matrix applications in Maya helped me broaden my understanding od scripting and underlying implementations of procedures in Maya. As I have an emphasis in Entertainment Arts and Engineering, it is a great help to tie in those topics with the math that runs these programs.

SOURCES

Forms of Transformation matrices: https://gamedev.stackexchange.com/questions/68522/what-does-a-matrix-represent/68525

Maya API
http://download.autodesk.com/global/docs/maya2014/en_us/index.html?url=files/Arrays_vectors_and_matrices_Matrices.htm,topicNumber=d30e790407

MEL Wikipedia Article https://en.wikipedia.org/wiki/Maya_Embedded_Language