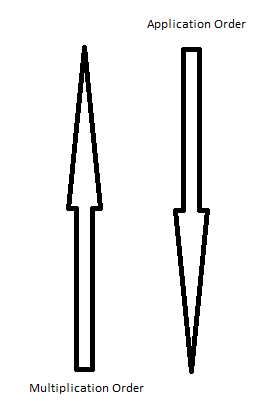
A GUIDE TO OPENGL MATRIX TRANSFORMATIONS

By: Jared Fowler

The purpose of this document is primarily to remind me, the writer, of the important discoveries and “ah-hah” moments I’ve had while learning matrix transformations while using openGL.

**LAW**: What you see when you look at the screen, call it your camera view, NEVER MOVES! Instead, we move the models via translations and rotations to give a sense of the individual model movement as well as the sense of the camera view movement. As such, what I’ll call the natural or true axis (x, y, z) of the window NEVER CHANGES.

What happens in the Model-View-Projection Matrix (MVP Matrix):



We first off should note that the order of matrix multiplication will in fact be the reverse order of application to the individual vectors. We would, for example, lastly multiply the scale matrix to the Model Matrix, but scaling would in turn be the first modification we apply to the model’s vectors.

I have found it generally best to keep an updated copy of a model’s translation, rotation, and scale matrices. The order of translation and rotation greatly matters. Remember that we are **ALWAYS** working off the true origin. Rotations are done around a specified vector axis which passes through this origin. For this reason, if we want an orbiting effect we should translate out first and then rotate, versus if we want to rotate the object about itself we should first rotate it and then translate it out to where we want it. We now have our model matrix.

We SHOULD NOT think of the model matrix and view matrix as two completely separate things. We will multiply them and they will be applied to the model’s vectors. This for me was a major “ah-hah” moment. When moving the camera, (especially in a free cam mode), we should remember that we are not rotating the camera, rather, we are rotating (in an orbital fashion around the origin which is where the camera is located at!) all the models which gives the illusion that the camera is rotating. This was another “ah-hah” moment for myself when I realized that the axis of rotation was therefore the true (x, y, z) axis!

