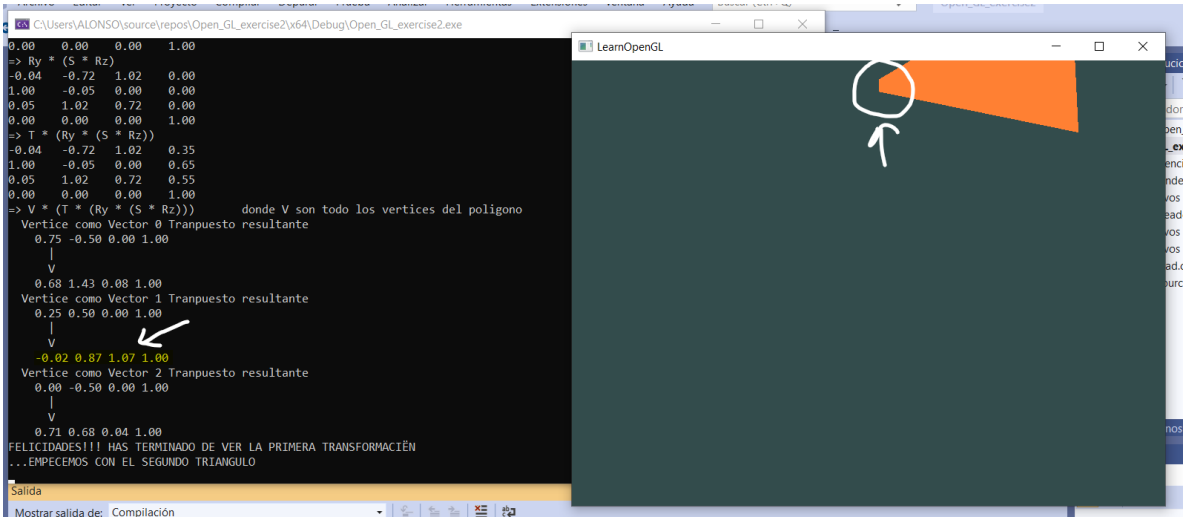
Pequeño documento que acompaña el código:

¡Hola! Me gustaría explicar mas o menos como funciona el código. Existen 2 varias clases implementadas por mí. Todas las clases están pensadas para ser usadas tanto por separado como en conjunto, por eso mismo he visto por conveniente solo usar un objeto “Matrix” que me brinda todas las matrices que necesito y lo mejor es que me las construye con los parámetros que yo deseo. Es por eso que la clase “GLM” no trabaja en esta ocasión, debido a que esta pensada para trabajar directamente sobre el polígono y no multiplicando varias matrices primero. La multiplicación de varis matrices primero es una gran mejora que reduce la complejidad de los cálculos.

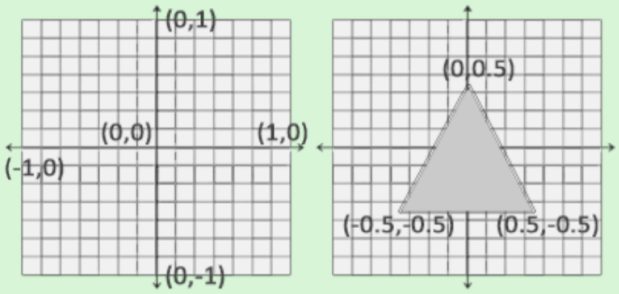
En resumen, solo trabajan las clases “Poligon” (que en lo personal me facilita todo el manejo de vértices y actualización) y la clase “Matrix” que me provee todas las matrices necesarias. Algo que quiero resaltar es el hecho de algunos puntos tienen coordenadas que se salen del espacio de la ventana y el más curioso de todos es Z que al igual que X e Y solo va de -1 a 1 por lo que en el primer triángulo al tener uno de los vértices un Z mayor que 1, no se visualiza en su totalidad.



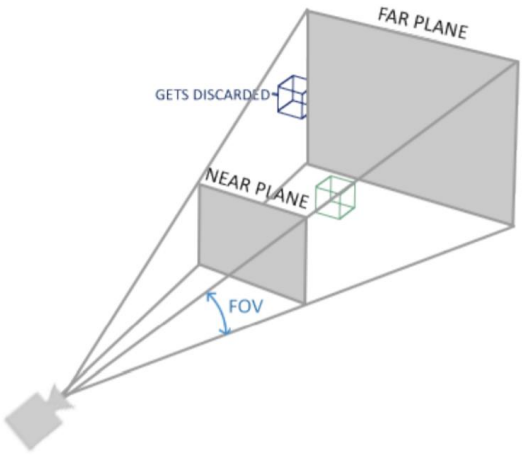
Esto hace parecer que tenemos un trapecio. Al principio pensé que mi implementación esta errónea pero después de leer esto:

Normalized Device Coordinates (NDC)

Once your vertex coordinates have been processed in the vertex shader, they should be in normalized device coordinates which is a small space where the `x`, `y` and `z` values vary from `-1.0` to `1.0`. Any coordinates that fall outside this range will be discarded/clipped and won't be visible on your screen. Below you can see the triangle we specified within normalized device coordinates (ignoring the `z` axis):



Entendí que lo que sucedía era que la figura se salía del espacio que cubre el eje Z en OpenGL. Se preguntará ¿No se puede agrandar el espacio de los ejes? Y la respuesta que encontré es que teóricamente no, pero se puede variar la perspectiva con el uso de una cámara que defina la zona en la que veamos nuestras figuras:



Donde todo lo que está entre el NEAR PLANE y FAR PLANE será lo que se visualice. Pero debido a que no se nos pidió eso, no lo consideré necesario implementarlo.

Finalmente, el programa solo avanzará en el proceso presionando la tecla “ESPACE” y a cada paso mostrará que matriz resultante se ha obtenido. Una vez finalizada la operación con las primeras operaciones (las de la columna L) pasa inmediatamente a resetear todo y comenzar con el proceso para las operaciones de la columna R. Eso sería todo.

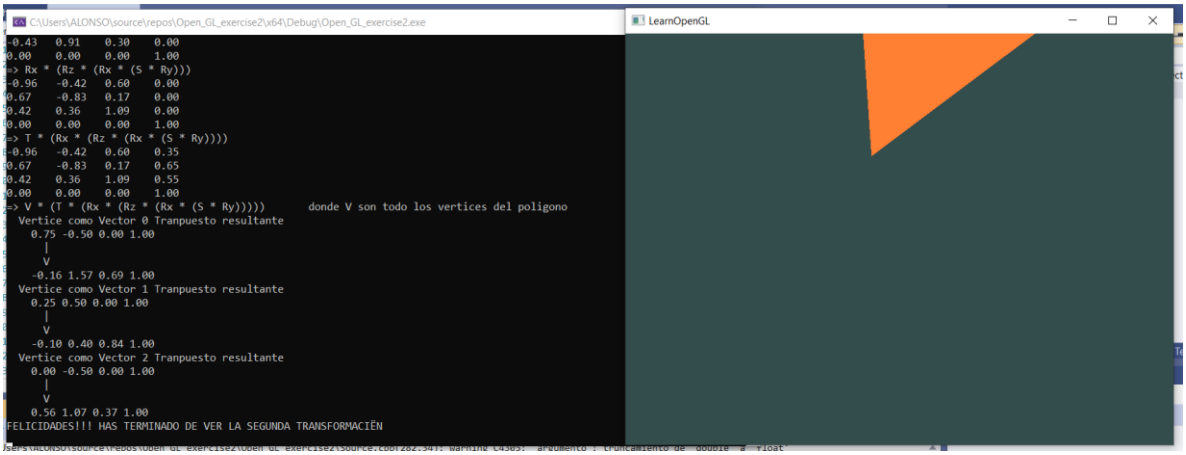


Imagen final del resultado de las operaciones de la columna R.