

Game Engines

Project 2: Wireframe renderer

Implementing the 3d wireframe renderer was pretty much straightforward after reading through the given [overview of the 3d transform process](#)¹ for the second time and realizing that what's needed is basically computing three matrixes, multiplying them together and using the resulting matrix to multiply with each vertex as a column vector, as shown in the given pseudocode.

The implementation consists of a HTML5 file, index.html, that includes jQuery Mobile for easy inclusion of a few UI widgets and then there is the wire frame rendering code in 3dWireframeRenderer.js

The function getCameraLoacationTransformMatrix implements what's described in section 3.1.1 of the overview, *Setting the camera location*, function getCameraLookTransformMatrix returns the matrix described in section 3.1.2, *Pointing and orienting the camera*, and the projection matrix from section 3.2 comes from the function getPerspectiveTransformMatrix

The test meshes are in meshdata.js and I got them from <http://threejs.org/editor/> and it's export function. One of the biggest difficulties was deciphering the faces array in the JSON export from there but then I found [some documentation](#)² on where the vertex indexes are located and the function getTriangleFaces does the job of extracting the vertices for each face.

When I had the function renderWireframe (basically like the given pseudocode) drawing vertices (I have them cover four pixels for clarity) and connecting them with lines, I had some difficulty finding a good combination of near and far values and camera Z position. Adding sliders for those values in the UI helped, but, near clipping happens quite far away from the camera as it seems - I haven't found a combination of near, far and camera Z position that allow the object to come near the camera without clipping, *except*, if I reverse the near and far values, for example set near to -300 and far to -10, and the camera Z position to 150, then the object (cube) renders happily close to the camera; is that a feature of the transformation matrixes or a bug in my implementation? I don't know...

The camera movement could be connected to the arrow / wasd keys and mouse but to see clearly the interplay between camera XYZ and the near and far planes is of most interest here so I'll let those sliders do.

I tried getting the width and height from given FoV, near plane position and aspect ratio, as discussed in the overview and that didn't play well with my UI so I abandoned that, but what I tried can be seen in the function getWidthAndHeightFromFOV.

¹ "Overview of transforms used in rendering"

<https://blog.itu.dk/MGAE-E2013/files/2013/09/transforms.pdf>

² three.js JSON Model format 3.1: <https://github.com/mrdoob/three.js/wiki/JSON-Model-format-3.1>