CMSI 371-01

COMPUTER GRAPHICS

Spring 2013

Assignment 0326 Feedback

For this assignment, outcomes *2a*, *2b*, *3d*, and *3e* max out at **|** because the requested functionality in this assignment do not yet reach the culmination of what these outcomes represent overall.

Britain Southwick

2a — You've taken a few more concrete steps toward full 3D transform proficiency—now on to using these functions in your scene! (**|**)

2b — The mechanics of your ortho and frustum functions look good. The next test will be to use them in your scene. (**|**)

3d — Your matrix library is certainly moving in the right direction. Actual "field testing" in your 3D scene code is up next. (**|**)

3e — Your matrices represent additional progress toward 3D scene rendering, but as mentioned will not top out this outcome yet because we haven't covered the full range of shader functionality yet. (**|**)

4a — The code that you have works well so far, and this is bolstered by having a unit test suite available to "keep it honest." (**+**)

4b — The biggest issue with your code is the way they are separated: some of your matrix function definitions are in the top-level namespace, and should be scoped to `Matrix4x4`. That way, the only name that you define at top-level is `Matrix4x4`. This kind of minimal impact to the JavaScript namespace is an important practice to perform. It is a big enough deal that it hurts this outcome. (**/**)

4c — Your matrix code is quite readable, but struck me as a little too tight in *matrix4x4-test.js*. I would say that, for such code, line-break a little more aggressively. I think that you were also trying to preserve the 4×4 matrix formatting with your indentation, but for the ones where the array starts far from the left side of the code, avoiding such a deep indent overrides the fidelity to strict 4×4. (**|**)

4d — Your work shows fine resource use, including leveraging the projection matrices that are already in the handouts. (**+**)

4e — Your commit phasing and messages show a decent pace, particularly with the separation of some work into distinct groups of functions. Make sure to establish this as a work habit: write the test; write the implementation; commit when the test succeeds. That gives your commit log a very logical, trackable evolutionary trajectory. (**+**)

4f — Submitted on time. (**+**)