Graphics II – Gouraud Shading and Such

Ross, Hugues
<hugues.ross@mymail.champlain.edu>

March 2, 2015

Gouraud shading is like Phong shading's little brother, unable to live up to the expectations that the elder created, but still trying his darndest. Ironically, according to Wikipedia Gouraud is slightly older. Whereas Phong Shading prepares some things in the vertex shader, then leaves the pixel shader to finish them, Gouraud shading just sticks it all in the vertex shader. This leads to the unfortunate fact that Gouraud is only as smooth as its polycount, as it produces a set of vertex colors. The process goes a bit like this:

1. Get the normals and vertex position into world space
2. Set the ambient to a static value
3. Get a vector from the (world)vertex to the light, and normalize it
4. Dot the result to the (world)normal to find the verts that face the light. That gets you diffuse.
5. Get a vector from the (world)vertex to the Eye of Sauron (Frodo Baggins says hi)
6. Reflect the light vector over the normal(effectively, see how the light bounces)
7. Dot the result to the Eye of Sauron to see whether the One Ring(light vector reflected) reaches His Fiery Gaze(the camera). That makes specular.
8. Combine the above with any interesting colors/samplers/whatevers to get your final color

1 That one bug

It deserves its own heading. I had a bug that took 3+ days to debug, where no matter what I did my specular component was wrong. Not in any particular fashion, it changed in a variety of interesting and completely wrong ways, all while looking almost exactly like any other Gouraud implementation. (By the way, that's why my gouraud implementation looks just like Johnathan's Phong – His worked before mine did, and I tried rearranging and changing several parts to match his. Spoiler: It did not fix the bug, even when I went to the point of renaming every variable to make the lines exactly the same as his.) Ultimately, I'm still unsure how I fixed it, but it's gone now and good riddance.