CS 457 Final Project a.k.a. The Red Planet

By: Rick Menzel **Email**: menzelr@oregonstate.edu

Link: https://media.oregonstate.edu/media/t/0_lzuvxk90

Proposal Text:

For my final project I want to produce an animated mask or face. Rather than animating the mask with facial movements and the like, however, I want to do a reverse explosion effect, such that it appears that the mask is assembled out of a collection of fragments. I envision this mask as having the following characteristics:

- Proper lighting (Ambient/Diffuse/Specular with sliders)
- Flat shaded. Possible toggle for Smooth Shading
- Mask will likely come from an imported obj
- I may also use a tessellation shader for the surface of the mask
- Adjustable material (metallic appearance and/or color)
- Animated either on a timer or using a slider to assemble/explode (geometry shader)
- If time permits and it is possible, I would like to see if I can do the reverse explosion not with pixels bit with entire polygons

Implementation:

My original proposal was to do a reverse explosion (assembly) effect on a mask object with a full lighting model. However, I had issues finding a good mask model and midway I got inspired by a shader space/starfield effect I saw which used a dispersed set of colored fragments to good effect. As a result, I decided to change my model to a planet and to add on a color shift to what I had previously proposed.

The final implementation uses the smoothstep and mix functions to shift the fragment colors from the base object color to pseudo-randomly generated "star" colors as the move further from the base position. Note that the full lighting model is retained even in this exploded state, and it is possible to adjust ambient, diffuse and specular lighting components at any time. Additionally, the light position can be freely set by the user, as well as any of the several base object colors available (which may be selected using check boxes and/or manually adjusted with the color picker). The overall effect is then one of a planet forming from cosmic dust.

The lighting model is mostly done in the vertex shader (this is where the computation primarily takes place). These values are then passed through the geometry shader on a per-vertex basis before they are interpolated through the rasterizer and assembled in the fragment shader. While I had considered including a smooth shading option, I ultimately opted to include only a flat shading model because I felt it accented the cartoony, low-polygon planet model I chose. It is worth noting that getting the lighting model to work took some work, first to understand how to get it to play nicely when the geometry itself was being produced as points rather than surfaces, but also because during intermediate stages the specular highlights were making weird things happen to the produced points, causing some triangles to clip in and out as the object was moved. Ultimately these issues were resolved.

The reverse explosion effect relies on the geometry shader to render the planet as a series of points by subdividing the original triangles. While I had initially contemplated keeping the polygons intact during the explosion (i.e. have a bunch of triangular pieces explode like confetti), I ultimately opted for the point option for two reasons. First and foremost, the lack of a triangle output topology in the geometry shader mean I would have had to uses point is some way regardless (though using triangle strips as the output did make for some cool effects). Second, the model described in class of giving the points velocities based on distances from the center of their triangle resulted, at least in my opinion, in a better spread and overall a closer effect to my vision than did attempting to keep the points grouped in triangles and imparting each group with a common velocity. After I came across the aforementioned star field shader, the choice to use points was made.

Screenshot(s):

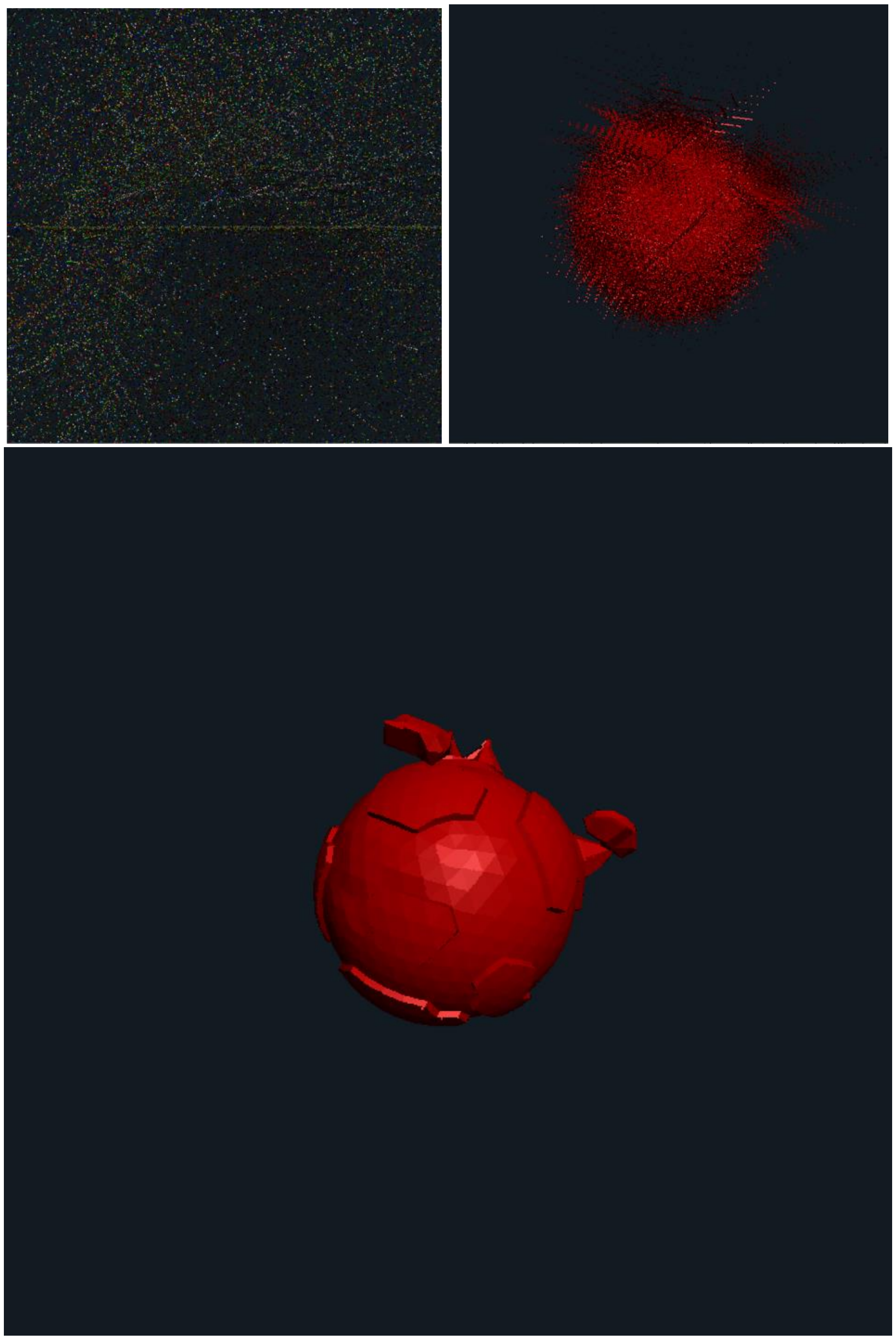

Fig 1. The Red Planet Coalesces from the Ether