Finite Solar Radiation Pressure Modeling using OpenGL

Jason Tichy, Abel Brown, and Michael Demoret
a.i. solutions,10001 Derekwood Ln, Lanham, MD 20706, 301-306-1756
first.last@ai-solutions.com

Keywords: *SRP, OpenGL, CUDA, Solar Sail, James Webb Space Telescope*

ABSTRACT

This paper describes an OpenGL approach for calculating high fidelity solar radiation pressure effects using general purpose graphical processing units (GPGPUs). For some missions solar radiation pressure (SRP) is a significant consideration wherein the simplified plate SRP model will not suffice. By modeling the spacecraft using many small polygons in OpenGL, incident light on the individual geometric pieces is easily calculated. The accuracy of this OpenGL solution is only limited by the polygon approximation of the spacecraft and knowledge of the reflective properties of the materials. As an example, we will investigate the SRP effects on the James Webb Space Telescope (JWST) whose sensor solar shade also acts as a giant solar sail.

The traditional way of computing SRP on a surface involves reducing a spacecraft down to simple area and a normal vector. Solar radiation pressure comes by reducing Maxwell's equation to become

$$P = \frac{E_f}{c} \cos \propto$$

Where E_f is the energy flux, c is the speed of light, and $\propto$ is the angle between the surface normal and the source of incident radiation. The force exerted by the SRP is determined by the product of pressure and the approximated exposure area. Clearly the problem here is that almost no spacecraft fits this description of a flat surface area.

As the number of polygons increase so too the fidelity of the SRP model. Here each polygon can be processed independently where the resultant force is the sum total over all polygons. That is, each plate (or polygon) can be processed in parallel. We natively represent vertices, polygons, and light sources on the GPU using OpenGL. OpenGL is a highly optimized GPU framework for drawing polygons and rendering lifelike scenes. The implementation of the flat plate model using OpenGL would be as simple as defining a polygon by its vertices and normal vector. However, OpenGL is optimized to look correct rather than be physically accurate. Meaning, care must be taken when extracting information from the rendered scene. Employing OpenGL allows for more complex shading schemes by associating normal vectors with a polygon's vertices. These vertex normals represent curvature along the polygon surface which is then "shaded" using built-in OpenGL models. Figure 1 demonstrates this concept showing how OpenGL renders polygons with vertex normals.

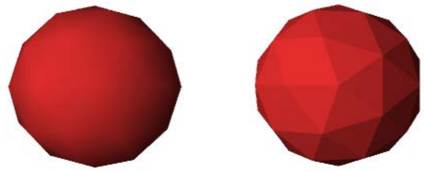

Figure 1: Sphere of polygons with and without defined vertex normals

The light properties for each portion of the model are extracted using OpenGL Shaders. Shaders are the low level algorithm which executes in parallel on the GPU and accounts for the tessellation, texture, and thus reflective properties for each polygon in the model. The shader algorithms report the light and coloration for each point on the spacecraft model which is converted to intensity, $E_f \cos \propto$, for the purposes of SRP modeling. Notably, the routines for determining the shade properties are built into the OpenGL framework. This includes light from multiple sources; therefor it is easy to include light from the Sun as well as albedo from objects like the Earth and moon with appropriate reflections for land, sea, and ice respectively.

This paper demonstrates how complex SRP forces could be better modeled using an OpenGL/GPU solution. Since the capabilities for computing advanced lighting and shading are built into the core language, OpenGL is the natural choice for such endeavors. Furthermore, it is possible to model an arbitrary spacecraft using very large numbers of polygons since each polygon can be processed in parallel. Finally, we present an application to JWST, which yield significant improvements in speed and fidelity with respect to existing tools.