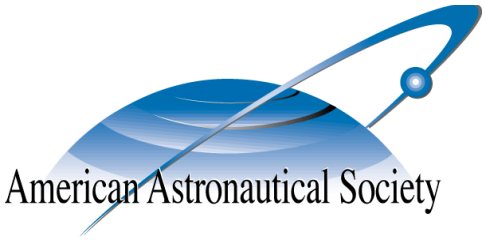
AAS 16-500



American Institute of
Aeronautics and Astronautics

HIGH GEOMETRIC FIDELITY MODELING OF SOLAR RADIATION PRESSURE USING GRAPHICS PROCESSING UNIT

Patrick W. Kenneally and Hanspeter Schaub.

AAS/AIAA 26th Spaceflight Mechanics Meeting

Napa, California **February 14-18, 2016**

AAS Publications Office, P.O. Box 28130, San Diego, CA 92198

(Preprint) AAS 16-500

HIGH GEOMETRIC FIDELITY MODELING OF SOLAR RADIATION PRESSURE USING GRAPHICS PROCESSING UNIT

Patrick W. Kenneally* and Hanspeter Schaub†

This paper presents a method for the fast computation of spacecraft force and torque due to solar radiation pressure (SRP). The method uses the highly parallel execution capabilities of commodity Graphics Processing Unit (GPU) and the Open Graphics Library (OpenGL) vector graphics software library to render a Computer Aided Design (CAD) generated spacecraft model on the GPU. The SRP forces and torques are resolved per model facet in the custom-developed render pipeline. Material properties are encoded with the model to provide realistic specular, diffuse and absorption surface light interactions.

INTRODUCTION

Effective orbit determination, maneuver and mission design, and numerical mission simulations require tools that enable accurate modeling of the spacecraft dynamical system. Solar radiation pressure (SRP), the momentum imparted to a body by impinging solar photons, can become the dominant non-conservative force above Low Earth Orbit (LEO)[1] regime. Given this importance of SRP, knowledge of the resultant forces upon a body due to SRP are a primary consideration in the modeling and analysis of spacecraft operating in the high LEO region and above.[2,3]

The video game and animated video industries have driven the pursuit to create more vivid and realistic artificial worlds. This pursuit has resulted in highly optimized vector graphics software and GPU computer hardware capable of carrying out many thousands of floating point operations in parallel.[4] While these artificial worlds are visually persuasive, their implementation of electromagnetic radiation physics is understandably inaccurate. However, it is the parallel hardware and efficient vector graphics software implementations which may be used to simplify the steps of the SRP computation with great effect.

The ability to model and compute, at orders of magnitude faster than real-time, the SRP forces and torques on flexible and time varying spacecraft structures presents compelling opportunities. Current SRP evaluation approaches are capable of modeling the resultant force of an articulated spacecraft where the articulation motion is known prior to evaluation.[5] However, there are many instances in which the articulation motion and the spacecraft state are dependent on the myriad spacecraft control inputs and constraints. Accounting for all possible permutations of the spacecraft dynamical state is further challenged by the inclusion of flexing in the spacecraft structure.

It is evident then that a method of SRP evaluation characterized by an ability to include time varying information of the spacecraft state has potential for a wide range of applications. For exam-

*Graduate Research Assistant, Aerospace Engineering, University of Colorado, Boulder.
†Alfred T. and Betty E. Look Professor of Engineering, Aerospace Engineering Sciences, University of Colorado at Boulder, 431 UCB, Colorado Center for Astrodynamics Research, Boulder, CO 80309-0431, AAS Fellow

ple, large deployable structures, such as the IKAROS solar sail, would gain the ability to iteratively evaluate with greater fidelity the time varying control actuation of the solar sail during and after deployment.[6] Additional opportunities exist in the design and simulation of atypical spacecraft maneuvers. Maneuvers which utilize SRP as an actuator are exemplified by the maneuver designed for the rescue of the Hayabusa spacecraft. Hayabusa lost attitude control due to a failure of the reaction control system. Upon returning to a power positive state ground teams regained attitude control via the electric propulsion system at the expense of valuable fuel reserves. As a result, a cruise maneuver was designed which incorporated SRP to balance the torque induced by the swirling electric thrusters and thus save fuel for the return to Earth.[7]

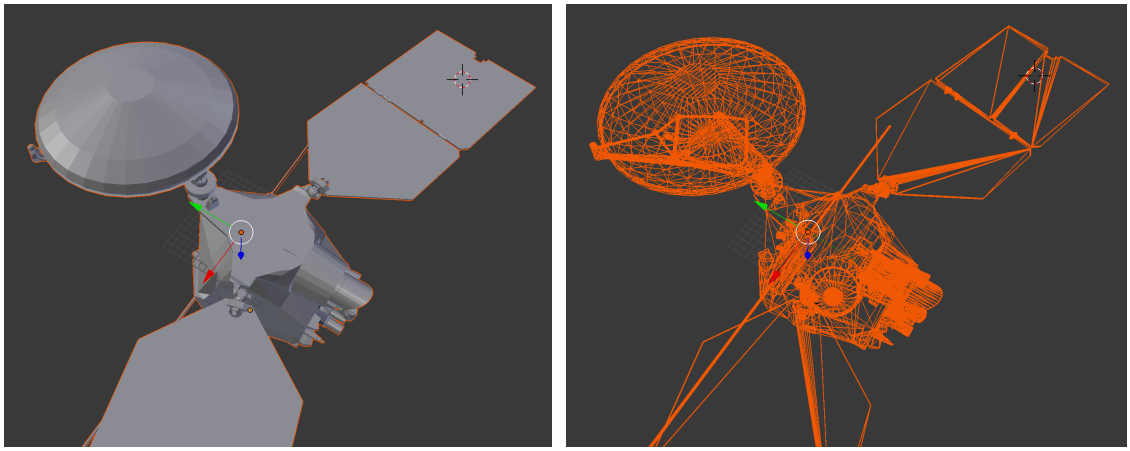
A survey of the current landscape of SRP research reveals a variety of approaches. Ziebart characterizes SRP evaluation as a two step process.[8] The first step is the development of an analytic model; the second is to compute a result from the analytic model.

A common approach and one of the most basic with regard to the analytic development is referred to as the cannonball model. The cannonball analytic model is given in Eq. (1), is computed from the surface area upon which radiation is incident A, solar flux $\Phi_{\odot}$, the spacecraft mass M, speed of light c, heliocentric distance to the spacecraft r and the reflection, absorption and emission characteristics of the spacecraft surface which are grouped together within the coefficient of reflection C_r. It is often the case that the C_r parameter is continually estimated and updated by an orbit determination effort. This model was most notably used during the LAEGOS missions and continues to prove useful for initial mission analysis.[9]

$$a_{\odot} = -C_{\mathrm{r}} \frac{A\Phi_{\odot}}{Mc} \left(\frac{1AU}{r} \right)^2 \hat{s} \tag{1}$$

Increased modeling accuracy is often achieved by departing from the cannonball assumption and defining shape approximations of the spacecraft. A common shape approximation is to model the spacecraft bus and solar panels as a box and panels respectively. Additionally the individual reflection, absorption and emission characteristics are kept distinct for each surface and set based on known spacecraft material properties.[10] However, common among shape approximation methods is that much of the modeling uncertainty occurs in an estimation process within the second step of the SRP evaluation. It is the model's computation, the second step of the process, in which much work is being done. Notably Zeibart, proposes a SRP evaluation procedure which computes the body forces over all 4π steradian attitude possibilities as step one.[8] Ziebart's approach is also capable of modeling self-shadowing by using ray-tracing techniques and spacecraft re-radiation via reduced spacecraft thermal model. McMahon and Scheeres extend such a model by aggregating the resultant SRP forces into a set of Fourier coefficients of a Fourier expansion.[10] The resulting Fourier expansion is available for both online and offline evaluation within a numerical integration process. Evaluation of the Fourier expansion in numerical simulation demonstrates successful prediction of the periodic and secular effects of SRP. Additionally, the Fourier coefficients may replace spacecraft material optical properties estimated during the orbit determination effort.

More recently methods which make use of the parallel processing nature of GPUs have been developed. Tanygin and Beatty employ modern GPU parallel processing techniques to provide a significant reduction in time-to-solution of Ziebart's "pixel array" method.[11] Inspiring some of the methods presented in this paper Tichey et al. use OpenGL, a vector graphics GPU software interface common in video games, to dynamically render the spacecraft model and evaluate the force of the



(a) Solid MRO CAD model (b) MRO triangle primitives

Figure 1. Example of Computer Aided Design (CAD) spacecraft model.

incident solar radiation across a spacecraft structure approximated by many thousands of facets.[12]

This paper presents a method for computation of the spacecraft accelerations using the parallel execution capabilities of a graphics processing unit (GPU). The method effectively leverages and repurposes software tools and techniques from the computer graphics discipline. The SRP forces and torques are resolved at a per facet level and summed over the surface of a CAD based spacecraft model. Material properties are encoded into the model to provide realistic secular, diffuse and absorptive surface light interactions. Finally, surface properties and spacecraft model articulation are available as inputs to define arbitrary time varying model parameters.

SRP EVALUATION PIPELINE

The proposed method utilizes a computer vector graphics rendering pipeline to compute the SRP forces over the spacecraft surface. Computer vector graphics systems process sets of points defined in three dimensions. These points, referred to as vertices, are combined into groups of three to define a triangle shape primitive. A computer aided design (CAD) model is approximated as a system of many thousands of small triangle primitives, combined into a data structure referred to as a mesh. Where a flat plate surface such as a solar panel may be modeled by a single rectangle resolved as two triangles, a curved surface may be approximated by many smaller triangles. Such an approximation is shown in Figures 1(a) and 1(b). In these figures it is evident that the Mars Reconnaissance Orbiter (MRO) high gain antenna shown Figure 1(a) is approximated by many thousands of primitives while the solar panels are approximated in Figure 1(b) by ten.

The entire spacecraft may be defined by one or more sub-meshes. Each mesh is assigned a material definition which describes the mesh's diffuse, specular and absorptive optical properties. Material definitions are typically and most conveniently defined within a CAD software tool, however, they also can be declared manually and assigned to each mesh in the custom-render pipleline presented in this paper. The density of primitives used to approximate the spacecraft's structure determines the geometric fidelity. Further, where relevant, an increased number of mesh material definitions will provide a more accurate SRP evaluation.

This method employs the Open Graphics Library (OpenGL) to facilitate the processing of the

spacecraft model primitives and evaluate the SRP forces and torques. OpenGL is a language independent application programming interface (API) for rendering computer vector graphics.[13] The API provides tools to send, retrieve and process data on an OpenGL compliant GPU.

The OpenGL rendering process, referred to as the pipeline, is divided into stages where each stage in turn operates on either a vertex or primitive input data type. Each stage is defined by a mini-program called a shader as shown in the pipeline Figure 2. For each pipeline stage many thousands of parallel instances of a shader program are executed, each performing processing on their specific input data type.[14] It is this highly parallel per primitive operation for which GPU devices have been specifically designed. The OpenGL method is implemented by a set of custom shader stages which facilitate the evaluation the spacecraft force and torque due to SRP.

At a minimum, an OpenGL pipeline requires vertex and fragment shader stages. The vertex shader's purpose is to prepare the vertex data for the following render stages by mapping the body frame defined vertex data to the inertial, camera view and screen projection coordinate spaces.[14] Each of these coordinate frame mappings is carried out to support the on screen rendering of the model. The fragment shader is not involved in the force and torque evaluation. It transforms the rendered model into screen space coordinates to provide the final screen space pixel definition. The on screen view generated by the vertex and fragment shader stages provides important first order confirmation of a correctly implemented custom shader pipeline.

The SRP force and torque calculations are carried out within the geometry shader stage. As shown in Figure 2 the geometry shader stage execution follows the vertex shader. The geometry shader receives vertex data along with values for solar flux $\Phi_{\odot}$ and the sun unit direction vector defined in the body frame $\hat{\mathbf{s}}_B$. The force for each k^{th} primitive, $\mathbf{F}_{\odot_k}$, is evaluated in the spacecraft body frame using the expression shown at Eq. (2), where $P(|\mathbf{r}_{\odot}|)$ is the solar radiation pressure scaled by the heliocentric distance to the spacecraft.[16]

$$\mathbf{F}_{\odot_k} = -P(|\mathbf{r}_{\odot}|)A_k \cos(\theta_k)\left\{(1-\rho_{s_k})\hat{\mathbf{s}} + \left[\frac{2}{3}\rho_{d_k} + 2\rho_{s_k}\cos(\theta_k)\right]\hat{\mathbf{n}}_k\right\} \tag{2}$$

The primitive area A_k is computed in Eq. (3), where $\mathbf{e}_1$ and $\mathbf{e}_2$ are edges of the primitive defined by the vertices.

$$\mathbf{e}_1 = \mathbf{v}_2 - \mathbf{v}_1 \tag{3a}$$

$$\mathbf{e}_2 = \mathbf{v}_3 - \mathbf{v}_1 \tag{3b}$$

$$A_k = \frac{1}{2}\|\mathbf{e}_1 \times \mathbf{e}_2\| \tag{3c}$$

The sun angle of incidence θ_k as given by Eq. (4), is simply the dot product of the primitive surface normal $\hat{\mathbf{n}}_k$ and the sun unit vector $\hat{\mathbf{s}}$.

$$\hat{\mathbf{n}}_k = \frac{\mathbf{e}_1 \times \mathbf{e}_2}{\|\mathbf{e}_1 \times \mathbf{e}_2\|} \tag{4a}$$

$$\theta_k = \hat{\mathbf{n}}_k \cdot \hat{\mathbf{s}} \tag{4b}$$

The parameters ρ_{s_k} and ρ_{d_k} in Eq. (2) are respectively the specular and diffuse reflection coefficients of the material definition associated with the primitive. Additional radiation pressure sources

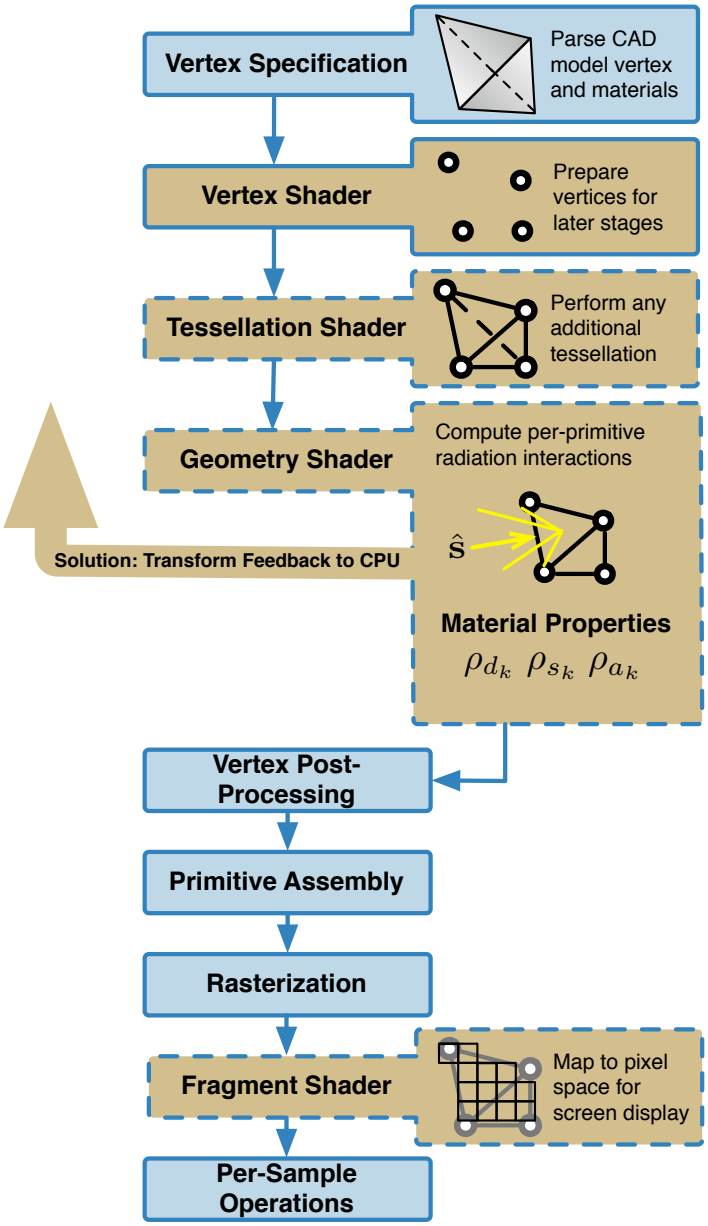


Figure 2. The custom OpenGL render pipeline demonstrating the discrete steps carried out by shader operations.[15] Gold fill stages are custom shader stage implementations. Blue full stages are OpenGL built-in shader stages. A Solid border stage indicates a required pipeline stage, while a broken border indicates optional stages.

such as albedo from planetary bodies may be accounted for through an additional contribution in the $P(|r_{\odot}|)$ term. Following the force computation, the torque $\mathbf{L}_k$ contribution of a single primitive, as given in Eq. (5), is computed as the cross product of the vector defined from the body frame origin to the primitive's centroid $\mathbf{c}_k$ and the per primitive force $\mathbf{F}_{\odot_k}$.

$$\mathbf{L}_k = \mathbf{c}_k \times \mathbf{F}_{\odot_k} \tag{5}$$

A simplified geometry shader code implementation is shown in Listing 3. To output the final evaluated primitive the geometry shader must make three calls to `EmitPrimitve()`, one for each vertex, and a final call to `EmitPrimitve()` as shown on lines 39 and 42 in the listing in Figure 3. The force and torque computations are made when processing the first vertex or a primitive and no operations are executed for the remaining two vertices. The GPU executes many thousands of geometry shader instances in parallel resulting in the force and torque contributions of many thousands of primitive being evaluated simultaneously. To retrieve the resulting force and torque from the GPU a Transform Feedback buffer mechanism is used. The Transform Feedback feature of OpenGL allows for the retrieval of data from the GPU process to the CPU process via an OpenGL buffer. Once the force and torque data is returned to the CPU the values are summed to evaluate the total spacecraft force and torque vectors due to SRP.

The OpenGL method provides an ease of setup and configuration, which the authors believe, is not present in other techniques. The easy model import process allows the user to select the level of model vertex detail and materials in familiar CAD software tools. The only user required input to the modeling is the spacecraft CAD model. The remainder of the inputs such as solar flux, planetary ephemeris and spacecraft dynamics states are supplied by numerical simulation software. This modeling approach is integrated into a stand alone tool which can be incorporated into any trajectory numerical simulation. At each integration step the model is provided with updated vehicle material and articulation parameters. The pipeline is executed returning the force and torque results for each primitive. The final total force and torque are found by summing, on the CPU, the force and torque contributions of each primitive.

MODEL VALIDATION

The initial validation is performed by comparing results from an analytic cannonball model and a sphere shaped spacecraft model within the OpenGL method. The values for the LAGEOS II spacecraft, given in Table 1, are used in both evaluations. A spherical model spacecraft is generated to replicate the LAGEOS II spacecraft parameters. Figure 4 illustrates the spherical spacecraft model being evaluated using the OpenGL method. The perturbative acceleration due to SRP as given by the cannonball model and the OpenGL model are given in Table 2. The OpenGL model yields a resultant torque of 1.51×10^{-12} Nm. However, it is expected that a perfectly spherical object yields zero torque. The non-zero torque value returned by the OpenGL model is due to the faceted nature of the model not being perfectly spherical. It is observed that by increasing the number of vertices and therefore primitives in the model (better approximating a sphere), the resulting torque value approaches zero. The agreement between the two simple evaluations provides confidence of the methods correctness.

Additional validation is carried out using an equivalent implementation and data structures in MATLAB. The MATLAB validation tool has been used to validate single time step evaluations of

```
for (int i = 0; i < 3; i++)
    {
        if (i == 0) {
            dvec3 edge1 = dvec3(gs_in[1].position_modelSpace - gs_in[0].↩
                position_modelSpace);
            dvec3 edge2 = gs_in[2].position_modelSpace - gs_in[0].position_modelSpace;
            dvec3 faceNormal = cross(edge1,edge2);
            dvec3 nHat_B = normalize(faceNormal);
            cosTheta = dot(nHat_B, sHat_B);
            double P_sun = solarFlux/c; // [N/m^2] solar radiation pressure

            if (cosTheta > 0.0 && cosTheta <= 1.0) {
                area = 0.5*length(faceNormal);
                V[0] = gs_in[0].position_modelSpace.xyz;
                V[1] = gs_in[1].position_modelSpace.xyz;
                V[2] = gs_in[2].position_modelSpace.xyz;
                cg = (V[0] + V[1] + V[2])/3.0;

                forceSrp = -P_sun*area*cosTheta*(rho_a*sHat_B + 2.0*rho_s*cosTheta*↩
                    nHat_B + rho_d*(sHat_B + (2.0/3.0)*nHat_B));
                torqueSrp = cross(cg,forceSrp);
            } else {
                area = 0.0;
                forceSrp = dvec3(0.0,0.0,0.0);
                torqueSrp = dvec3(0.0,0.0,0.0);
            }
        } else {
            forceSrp = dvec3(0.0,0.0,0.0);
            torqueSrp = dvec3(0.0,0.0,0.0);
            cg = dvec3(0.0,0.0,0.0);
            area = 0.0;
            cosTheta = 0.0;
        }
        // Pass through geometry shader per vertex values
        gs_out.position_worldSpace = vec3(gs_in[i].position_worldSpace);
        gs_out.normal_cameraSpace = vec3(gs_in[i].normal_cameraSpace);
        gs_out.UV = vec2(gs_in[i].UV);
        gs_out.eyeDir_cameraSpace = vec3(gs_in[i].eyeDir_cameraSpace);
        gs_out.lightDir_cameraSpace = vec3(gs_in[i].lightDir_cameraSpace);
        gl_Position = gl_in[i].gl_Position;
        EmitVertex();
    }

    EndPrimitive();
```

Figure 3. Core OpenGL shader language code of the custom geometry shader. The force and torque computation as given at lines 18 and 19.

Table 1. LAGEOS II spacecraft parameters used for computation of SRP by cannonball model and OpenGL model.

LAGEOS II Attribute	Value
mass	405.38 [kg]
area	0.2817 [m^2]
Φ (at 1 AU)	1.38×10^3 [W/m^2]
C_r	1.12

Table 2. LAGEOS II spacecraft SRP induced acceleration computed by cannonball and OpenGL models.

Model	SRP Acceleration $a_{\odot}$
Cannonball	3.56×10^{-9} [m/s^2]
OpenGL	3.60×10^{-9} [m/s^2]

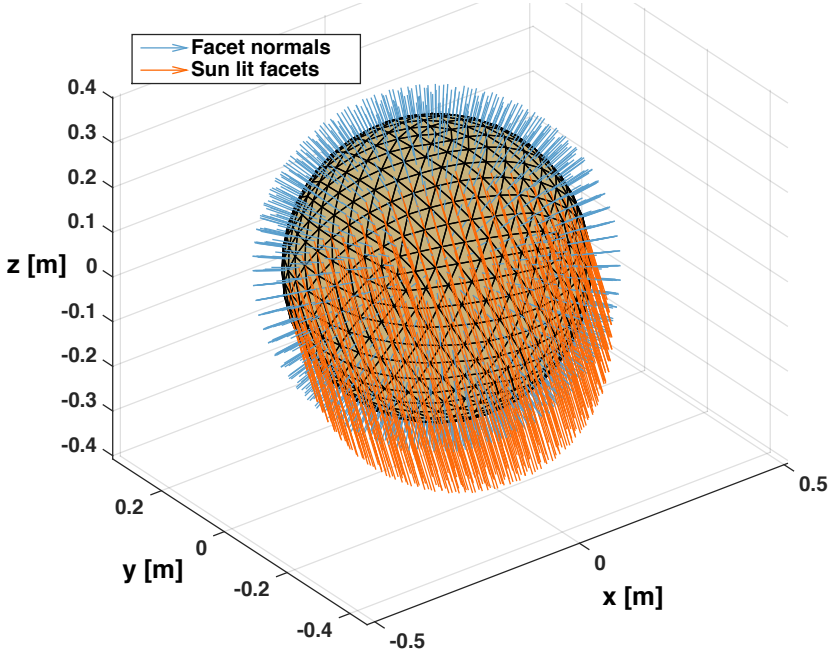


Figure 4. A visualized single evaluation of a 1280 primitive, LAGEOS size satellite. Orange vectors indicate $\hat{\mathbf{s}}_B$ incident on primitives. Blue vectors indicate primitive face unit normal vector $\hat{\mathbf{n}}_B$

more complex spacecraft model geometries. An example evaluation of a complex spacecraft geometry is shown in Figure 5. In this evaluation a 14750 primitive model of the Mars Reconnaissance Orbiter (MRO) is processed at a heliocentric distance of 1AU where the sun vector in defined in the body frame is $\hat{\mathbf{s}}_B = [0, -1, 0]^T$. Approximate material optical properties are assigned to the spacecraft bus whereas the solar array emmissivity and diffusivity, as quoted by You et. al. are set at 0.12 and 0.05 respectively. The SRP force value as determined by the MRO navigation team post launch of the spacecraft is $a_{\odot} \approx 9\times10^{-11}$ km/m^2.[17] The OpenGL method computed SRP acceleration is $a_{\odot} = 8.51\times10^{-11}$ km/m^2. The small difference between these two results is a promising indication that the OpenGL modeling method can offer a pre-launch SRP force accuracy close to that achieved after on orbit small force calibration exercises. More promising is to acknowledge that this OpenGL method result does not yet include modeling of thermal re-radiation and secondary photon impacts. The MRO navigation team found that thermal re-radiation in particular was a significant contributor to the total observed small forces due to radiation pressure.[17] The authors are confident that future near term model process additions such as thermal re-radiation and secondary photon impacts will allow for a fidelity of the spacecraft physical processes.

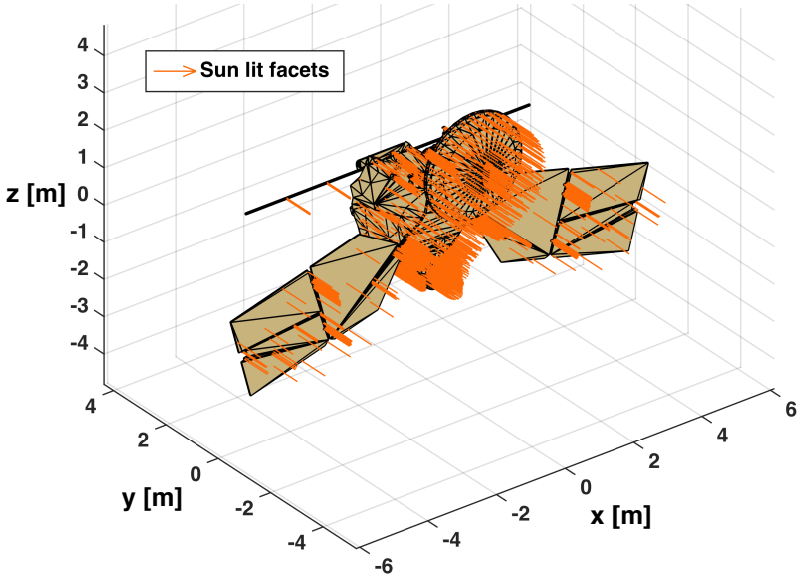


Figure 5. A single evaluation of a 14750 primitive, Mars Reconnaissance Orbiter. Orange vectors indicate $\hat{\mathrm{s}}_B$ incident on primitives.

COMPUTATIONAL PERFORMANCE

It is evident that computational performance is a goal of the OpenGL modeling method. To provide a crude demonstration of the increase in evaluation speed afforded by the OpenGL method a single evaluation of the MRO model shown in Figure 4 is carried out using the MATLAB verification tool and the OpenGL method. This comparison is termed 'crude' as the same serial evaluation implemented in MATLAB would undoubtedly execute in less time if implemented using a non-interpreted programming language such as C++. However, in a C++ implementation a difference in execution time would remain making the crude demonstration instructive. The reuslt of the two evaluations are given in Table 3. While it is clear that the highly parallel OpenGL method evaluates the model up to three orders of magnitude faster than the serial MATLAB implementation, much work remains to optimize and further decrease the OpenGL method execution time. Primary areas of optimization include moving the final per primitive force and torque summation process on to the GPU and performing analysis as to which computations do and do not require double precision variables in the shader code. Further, and most importantly, the authors are assessing the potential to utilize a general computing shader stage called "compute shader" in which to execute the entire radiation modeling process.

Table 3. Execution time for single evaluation of MRO model using serial MATLAB verification tool and the OpenGL method. (OpenGL 4.1 on 2015 MacBook Pro 3.1 GHz Intel Core i7, 8GB Ram, Intel Iris 6100 1536 MB).

Model Implementation	Execution Time [sec] 14750 Primitives
MATLAB	5.7
OpenGL Method	0.002

ONGOING MODEL DEVELOPMENT

In the SRP analysis of particularly complex spacecraft structures, secondary photon impacts cause significant variation of the final force and torque results.[5] Further work is being undertaken to implement the ability for the OpenGL method to compute secondary photon impacts. The addition of a reduced spacecraft thermal inertia model and thermal re-radiation will similarly be developed for on GPU execution.

CONCLUSION

The ability to resolve the SRP forces on a dynamically articulated spacecraft composed of heterogeneous surface materials presents opportunities in the areas of attitude maneuver design and spacecraft control. This paper demonstrates how complex SRP forces can be resolved more accurately and at high speed using an OpenGL/GPU solution. The greater geometric fidelity provided by a vertex based model allows easy initialization and subsequent evaluation of spacecraft dynamics given time varying spacecraft parameters.

ACKNOWLEDGMENT

The authors would like to thank Dr. Jay McMahon of the University of Colorado at Boulder for his guidance on the current state of SRP modeling and evaluation methods.

REFERENCES

[1] D. Vallado, *Fundamentals of astrodynamics and applications*. New York: Springer, 2007.

[2] H. F. Fliegel and T. E. Gallini, "Solar force modeling of block IIR Global Positioning System satellites," *Journal of Spacecraft and Rockets*, Vol. 33, No. 6, 1996, pp. 863–866, 10.2514/3.26851.

[3] J. A. Marshall, S. B. Luthcke, P. G. Antreasian, and G. W. Rosborough, "Modeling Radiation Forces Acting on TOPEX/Poseidon for Precision Orbit Determination," tech. rep., 1992.

[4] J. D. Owens, M. Houston, D. Luebke, S. Green, J. E. Stone, and J. C. Phillips, "GPU Computing," *Proceedings of the IEEE*, Vol. 96, No. 5, 2008, 10.1109/JPROC.2008.917757.

[5] M. Ziebart, S. Adhya, a. Sibthorpe, S. Edwards, and P. Cross, "Combined radiation pressure and thermal modelling of complex satellites: Algorithms and on-orbit tests," *Advances in Space Research*, Vol. 36, No. 3, 2005, pp. 424–430, 10.1016/j.asr.2005.01.014.

[6] R. Funase, Y. Shirasawa, Y. Mimasu, O. Mori, Y. Tsuda, T. Saiki, and J. Kawaguchi, "On-orbit verification of fuel-free attitude control system for spinning solar sail utilizing solar radiation pressure," *Advances in Space Research*, Vol. 48, No. 11, 2011, pp. 1740 – 1746. Solar Sailing: Concepts, Technology And Missions, http://dx.doi.org/10.1016/j.asr.2011.02.022.

[7] H. Kuninaka and J. Kawaguchi, "Lessons learned from round trip of Hayabusa asteroid explorer in deep space," *2011 Aerospace Conference*, 2011, pp. 1–8, 10.1109/AERO.2011.5747599.

[8] M. Ziebart, *High Precision Analytical Solar Radiation Pressure Modelling for GNSS Spacecraft*. PhD thesis, University of East London, 2001.

[9] D. M. Lucchesi, "Reassessment of the error modelling of non-gravitational perturbations on LAGEOS II and their impact in the Lense–Thirring derivation—Part II," *Planetary and Space Science*, Vol. 50, No. 10-11, 2002, pp. 1067–1100, 10.1016/S0032-0633(02)00052-1.

[10] J. W. McMahon and D. J. Scheeres, "New Solar Radiation Pressure Force Model for Navigation," 2010, 10.2514/1.48434.

[11] S. Tanygin and M. Beatty, Gregory, "GPU-Accelerated Computation of SRP Forces with Graphical Encoding of Surface Normals," *AAS/AIAA Astrodynamics Specialist Conference, At Vail, CO*, No. AUGUST, 2015.

[12] J. Tichy, A. Brown, M. Demoret, B. Schilling, and D. Raleigh, "Fast Finite Solar Radiation Pressure Model Integration Using OpenGL," 2014.

[13] "Khronos Group, OpenGL "https://www.opengl.org"," 10 2015.

[14] D. Shreiner, G. Sellers, J. Kessenich, and B. Licea-Kane, *OpenGL programming guide : the official guide to learning OpenGL, version 4.3*. Upper Saddle River, NJ: Addison-Wesley, 2013.

[15] "The Graphics Pipeline https://open.gl/drawing," Image, 10 2015.

[16] B. Wie, *Space vehicle dynamics and control*. Reston, VA: American Institute of Aeronautics and Astronautics, 2008.

[17] T.-H. You, E. Graat, A. Halsell, D. Highsmith, S. Long, R. Bhat, S. Demcak, E. Higa, N. Mottinger, and M. Jah, "Mars Reconnaissance Orbiter Interplanetary Cruise Navigation," *20th International Symposium on Space Flight Dynamics*, 2007.