# COMPUTER GRAPHICS FOR SPACE DEBRIS

*Piyush M. Mehta*[1], *Gonzalo Blanco Arnao*[2], *Davide Bonetti*[2], *Edmondo Minisci*[1], *and Massimiliano Vasile*[1]

[1]Dept. of Mech. & Aerospace Engr., Univ. of Stratchylde, 75 Montrose Street, Glasgow G1 1XJ, UK
[2]Deimos Space S.L.U, Ronda de Poniente 19, 28760 Tres Cantos, Spain

## ABSTRACT

The number of resident space objects re-entering the atmosphere is expected to rise with increased space activity over recent years and future projections. Predicting the probable survival and impact location of the medium to large sized re-entering objects becomes important as they can cause on-ground casualties and damage to property. Uncertainties associated with the re-entry makes necessary an expensive probabilistic approach. We present development and application of a new tool for quick estimation of aerodynamics and aerothermodynamics properties of the re-entering objects required for a probabilistic analysis. The novel method uses primitive geometries to develop a complex object and voxelization (voxels are the 3D equivalent of pixels in 2D images) for computing the shading/visibility factors to quickly estimate the aerodynamic and aerothermodynamics properties of the re-entering object. The tool can be used as a module of object oriented codes to support preliminary End of Life analyses.

***Index Terms***— Re-entry; Visibility/Shading; Voxelization; aerodynamics; aero-thermodynamics

## 1. INTRODUCTION

Ever since the inception of the space age, the number of resident space objects (RSOs) including spacecraft and debris in the different orbital regimes has been growing steadily. Warnings against the saturation of the space environment leading to a *cascade effect* have been sounded since very early in the space age; most cited of which is the work of NASA (National Aeronautics and Space Administration) scientist Donald Kessler in 1978 [1]. Several un-natural events such as collisions and explosions in orbit combined with the anticipated exponential growth in space activities is expected to significantly increase the RSO population including objects whose orbits traverse the low Earth orbit (LEO) regime. The atmosphere in LEO perturbs the orbit of the space object towards eventual re-entry into the Earth's atmosphere.

Because parts of medium-to-large objects with mass larger than a ton, re-entering the atmosphere can survive the harsh environment and impact the ground potentially causing damage and casualties, the two biggest space administrations NASA and ESA (European Space Agency) have set out guidelines and requirements for the allowable risk due to any re-entering object [2, 3]. Complying with these requirements require an end-of-life analysis for all planned missions. Several tools have been developed by different organizations over the years for modeling and simulation of re-entry and for verification of compliance. Most tools can be classified as either spacecraft- or object- oriented.

Spacecraft oriented tools perform the analysis with higher fidelity compared to object oriented tools, however they are harder to use and computationally expensive. Therefore, a logical approach involves using object-oriented codes for the preliminary analysis, followed by a more concentrated campaign with spacecraft oriented tools based on the preliminary results from the object-oriented tools. The only spacecraft oriented tool known to exist is SCARAB (Spacecraft Atmospheric Re-entry and Aero-thermal Break-up) developed HTG under a contract from ESA [4].

The object oriented approach uses a simplified representation of the re-entering object made up of primitive shapes such as sphere, cylinder, cone, etc. The approach assumes or calculates a demise altitude following which the object is assumed to break-up into multiple objects, represented by the individual primitive geometries [5, 6, 7].

The end-of-life analysis requires a Monte Carlo campaign due to the uncertainties involved and is computationally expensive but can be enabled with aerodynamics and aerothermodynamics models that can be evaluated quickly, possibly real time. The aerodynamic and aerothermodynamics models are used in predicting the trajectory, break-up, and survival of the re-entering the objects. Figure 1 shows an example of a Monte Carlo campaign run using Deimos' object-oriented tool DEBRIS [7].

The paper is presented with the following structure: section 2 discusses current methods used for visiblity/shading analysis and closed-form solutions required for quick evaluation of aerodynamic and aerothermodynamic models. Section 3 describes in detail the methodology behind and application of a new ray-tracing approach called *Pixelator*. Section 4 describes in detail a new and novel approach for conducting visibility/shading analysis and quick computation of aerodynamics and aerothermodynamics, called the *Voxelator*. Sections 5 and 6 describe the test cases develop for the validation of the *Voxelator* and the validation results, respectively. Section 7

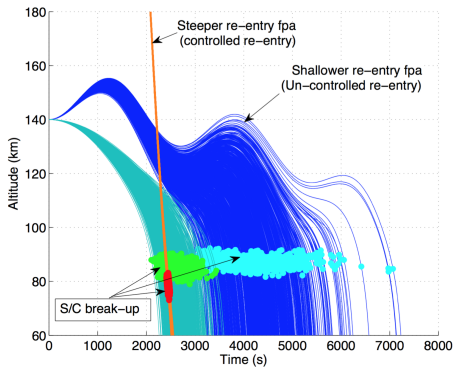



**Fig. 1**: Monte Carlo analysis and the concept behind an object-oriented tool [7]. fpa = flight-path angle

draws conclusions and hints at future work.

## 2. VISIBILITY/SHADING ANALYSIS

Computing the aerodynamic and aerothermodynamic properties of a re-entering object requires performing a shading analysis for part of the object visible to the flow. The shading analysis is typically performed using standard ray-tracing algorithm and can be expensive depending on the complexity of the object.

Figure 2 shows the working of a standard ray-tracer. For the purpose of re-entering objects, the light source and camera are coincident and represent the direction of the flow. The scene object is replaced with the triangulated version. Rays emitting from the light (point) source pass through a pixelated screen (image) and intersects with the object. In the limit that the light source is at an infinite or large finite distance, the rays become parallel when passing through the pixels.

The re-entering object is decomposed into triangular or tetrahedral mesh facets as shown for a sphere in Figure 2. Rays through each pixel are checked for intersection with each facet and in the case that the ray intersects with multiple facets, the intersected facet closest to the pixel screen is determined to be visible.

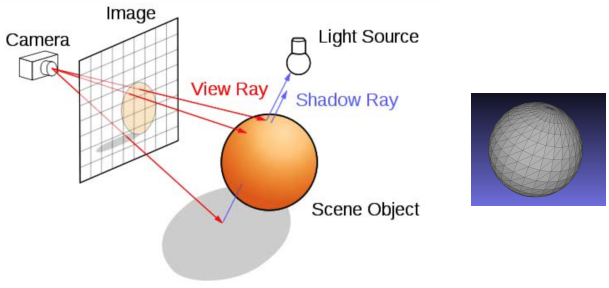



**Fig. 2**: Schematic of a standard ray-tracer

Once the visible facets are determined, orientation-specific closed-form solutions for aerodynamics and aerothermodynamics are applied to the visible facets. The choice of closed-form solution depends on the characterization of the flow. The flow is characterized as continuum, transition or free-molecular based on the value of Knudsen number *Kn*, which is given as the ratio of the mean-free-path ($\lambda$) to the characteristic length of the object ($L$).

$$Kn = \frac{\lambda}{L} \tag{1}$$

*Aerodynamics*: In the continuum regime, the pressure contribution from each visible facet is computed using Newtonian Theory [8] while the shear contribution is assumed to be zero.

$$C_p = C_{p_{max}} sin^2\theta \tag{2}$$

where $C_p$ is the local pressure coefficient, $C_{p_{max}}$ is the maximum or stagnation point pressure coefficient.

The pressure and shear contribution of each facet in the free molecular (FM) regime is computed using Schaaf and Chambre's analytic model given in Eqs. 3 and 4 that accounts for both pressure and shear contributions [9] .

$$C_P = \frac{1}{s^2}\left[\left(\frac{2-\sigma_N}{\sqrt{\pi}} s\sin\theta + \frac{\sigma_N}{2}\sqrt{\frac{T_w}{T_\infty}}\right) e^{-(s\sin\theta)^2} + \left\{(2-\sigma_N)\left((s\sin\theta)^2 + \frac{1}{2}\right) + \frac{\sigma_N}{2}\sqrt{\frac{\pi T_w}{T_\infty}} s\sin\theta\right\} (1 + erf(s\sin\theta))\right] \tag{3}$$

$$C_\tau = -\frac{\sigma_T cos\theta}{s\sqrt{\pi}}\left[e^{-(s\sin\theta)^2} + \sqrt{\pi} s\sin\theta(1 + erf(s\sin\theta)\right] \tag{4}$$

where $C_p$ and $C_\tau$ are the pressure and shear coefficients, respectively, $\sigma_N$ and $\sigma_T$ are the normal and tangential momentum accommodation coefficients, respectively, $T_w$ is the surface or body wall temperature, $T_\infty$ is the free stream translational temperature, $V_\infty$ is the object or free stream velocity, erf( ) is the error function, and $s$ is the speed ratio given as:

$$s = \frac{V_\infty}{\sqrt{2RT_\infty}} \tag{5}$$

where $R$ is the universal gas constant. The axial and normal forces are integrals of the pressure and shear stress distributions over the surface.

Aerodynamic coefficients in the transition flow regime are computed using a bridging function. Wilmoth *et. al.* [10] derived bridging function that have been used for over a decade. Mehta *et. al.* [11] recently derived sigmoid-based bridging functions that is able to better track data across the transition regime.

*Aerothermodynamics*: Modeling heat transfer is a lot less trivial compared to aerodynamics. Closed-form solutions for heat flux are applicable only at the stagnation point and are

limited to objects that have a finite nose radius such as a sphere and re-entry capsules. Several different assumptions are made for cases where the closed-form solutions are not applicable that may or may not be valid.

In the continuum flow regime, several closed-form solutions exist, however, the most widely used model is given by Fay-Riddell [12].

$$\dot{Q}_C = 0.94\,(\rho_w \mu_w)^{0.1}\,(\rho_s \mu_s)^{0.4}\,(h_s - h_w)\sqrt{\left(\frac{du_e}{dx}\right)_s} \quad (6)$$

where [$\rho_w$, $\rho_s$], [$\mu_w$, $\mu_s$], and [$h_w$, $h_s$] are density, viscosity, and enthalpy at the wall and stagnation point, respectively, and the last term is the velocity gradient at the stagnation point computed as:

$$\sqrt{\left(\frac{du_e}{dx}\right)_s} = \frac{1}{r_N}\sqrt{\frac{2\,(p_s - p_\infty)}{\rho_s}} \quad (7)$$

where ($du_e/dx$) is the velocity gradient, and $p_s$ and $p_\infty$ are stagnation point and free stream pressure, respectively.

Orientation-specific heat flux in the free-molecular-flow regime is computed using the model of Regan and AnandaKrishnan [13].

$$\dot{Q} = \frac{\alpha \rho_\infty V_\infty^3}{4 s^3 \sqrt{\pi}}\left[\left\{s^2 + \frac{\gamma}{\gamma - 1} - \frac{\gamma + 1}{2\,(\gamma - 1)}\frac{Tw}{T_\infty}\right\}\right.$$
$$\left.\left\{e^{-s^2 \sin^2\theta} + \sqrt{\pi} s \sin\theta\,[1 + erf(s \sin\theta)]\right\} - \frac{1}{2}e^{-s^2 \sin^2\theta}\right] \quad (8)$$

where $\alpha$ is the thermal accommodation coefficient and $\gamma$ is the specific heat ratio.

Stagnation heat flux in the transition regime is computed using the bridging function of Legge [14]. Different orientation-specific surface distribution models that can be used in the continuum and tansition flow regime exist and choosing between the two is a difficult task.

$$\dot{Q}(\theta) = \dot{Q}_s(0.1 + 0.9 cos\theta) \quad (9)$$

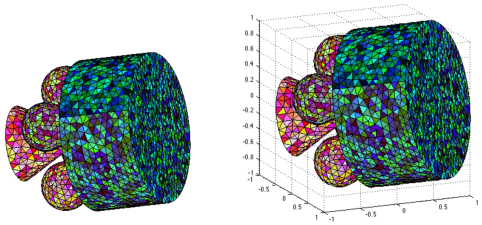
$$\dot{Q}(\theta) = \dot{Q}_s(0.7 cos\theta) \quad (10)$$

## 3. PIXELATOR: MESH BASED SHADING ANALYSIS

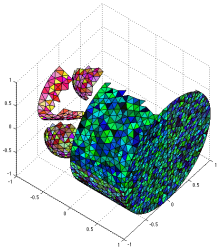
As part of this work, we first develop a new ray-tracing approach that uses the 2-Dimensional pixel data to avoid computing expensive ray-facet intersections. We call this new method '*Pixelator*'. The new approach developed in MATLAB® uses the following outlined steps:

1. Develop a triangular mesh for the object of interest.
2. Randomly generate and assign unique colors on the RGB (red-green-blue) spectrum to all the facets in the object geometry.
3. Generate a database for all the facets and color association.
4. Plot the colored object in a figure environment.
5. Take a screen shot of the colored object looking at it from the flow direction. Optimize pixel resolution for speed and accuracy.
6. Read the *c.data* for the screen shot and extract the visible colors.
7. Use the database of step 3 as a *key* and the extracted visible colors of step 6 to determine the visible facets.
8. Apply the closed-form aerodynamic and aerothermodynamic solutions to the visible facets

Figure 3 shows the application of the *Pixelator* on a simplified representation of an upper stage. Figure 3a shows the upper stage at a random orientation with unique colors assigned to the triangular facets. Figure 3b shows the visible facets as derived using the *Pixelator*. Figure 3c shows the Visible section of the upper stage rotated to confirm deletion of triangles not visible to the flow. In this specific case of the upper stage, the standard ray-tracer implemented in *C* programming language takes a between 1-2 seconds whereas the *Pixelator* implemented in MATLAB® takes only a fraction of a second.



(a) Complete Object (b) Visible Object



(c) Rotated Visible Object

**Fig. 3**: Pixelator applied to a simplified representation of an upper stage

Note: The RGB color model has a limited number of unique colors (65535). Therefore, the pixelator's application is limited to objects that have fewer triangular facets than the

number of unique colors in the RGB color model. However, most objects can be developed using fewer triangles than the number of unique colors. Also, the limitation can possibly be overcome by using the CYMK (cyan-magenta-yellow-black) color model.

## 4. VOXELATOR: VOXEL BASED SHADING ANALYSIS

The main idea behind this paper is a new, novel approach for performing the shading analysis based on voxelization. The method draws inspiration from the state-of-the-art computer graphics approach used for fast, real time rendering of objects, typically in a dynamic scene inside video games. Figure 4 shows the application of voxelization in rendering a scene inside a video game in real-time.

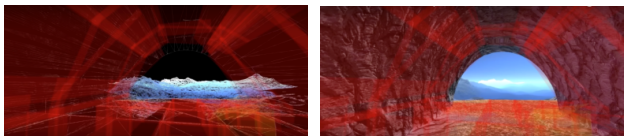
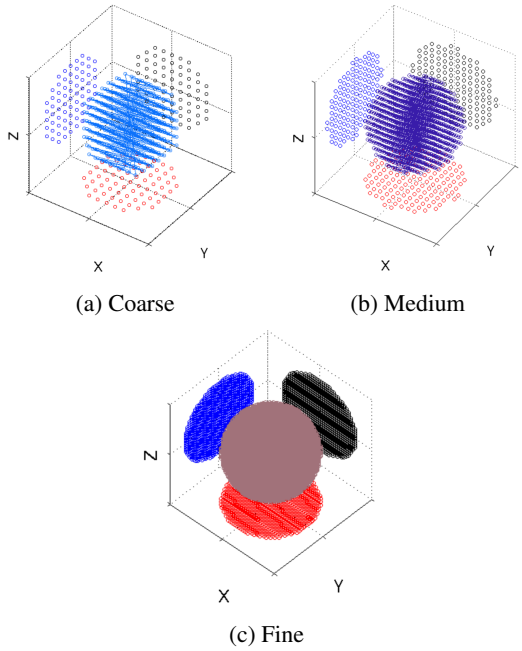


**Fig. 4**: Use of voxelization in computer games for real-time rendering

The method places voxelized objects - a simple primitive, combination of primitives or a complex shape composed of primitives - inside the domain of interest. Figure 5 shows voxelized representations of a sphere under different resolution ratios. Resolution ratio is defined as voxels per unit length and can be optimized for computational time (expense) and accuracy. The primitives are assigned relative and absolute orientations to create the desired complex geometry with each primitive assigned a unique ID. In the current version developed in MATLAB®, the object is always viewed along the X-axis, if a Cartesian co-ordinate system is assumed. The primitives and hence the object is rotated to the desired absolute orientation angle, keeping the relative orientations fixed. The primitives that are currently modeled in the tool are a sphere, a box (that can be used to model flat plates with a finite thickness), a cylinder, and a cone.

The method uses a pixel grid similar to that used in the standard ray-tracer, however, there are no rays traced from a point source. The method look down each pixel along the direction of the flow and stores the first voxel encountered as visible. The visiblity factor for each primitive is defined as the ratio of the visible voxels in the scene - determined using the corresponding unique ID - to the total number of visible voxels if the primitive was placed in the domain by itself - number of voxels in the projection of the primitive in the flow direction.

The method uses pre-computed aerodynamic and aerothermodynamic databases for primitive geometries in combination with the visibility factors computed using *Voxelator* to




(a) Coarse (b) Medium

(c) Fine

**Fig. 5**: Voxelized representations of a sphere with different resolution ratios

quickly compute aerodynamic and aerothermodynamic solutions for the complex shaped object.

$$X_{complex} = \sum_{i=1}^{N} X_{i,primitive} \times V_{i,primitive} \qquad (11)$$

where $X$ is the solution of interest, $N$ is the number of primitives that make up the complex object, and $V$ is the corresponding visibility factors.

### 4.1. Assumptions

The technique is developed with several inherent assumptions about its applicability:

1. The ability of the primitives to accurately represent the actual geometry of the re-entering object. The aerodynamic and aerothermodynamic models are strongly dependent on the geometric shape of the re-entering object. It is expected that additional primitives or parts of are required for accurately representing some of the complex geometries.

2. Pre-computed databases exist for the primitives used to create the simple and complex geometries. Presently, databases can be created for the different geometries using analytical approaches such as the Modified Newtonian Theory, which is an assumption in itself albeit a good one.

3. The method is expected to result in large errors at certain orientation, however, under the assumption of random tumbling, the mean error is representative of the performance.

### 4.2. Advantages

The Voxelator also brings several inherent advantages to the process of shadow analysis:

1. The biggest advantage is avoiding the need to perform meshing. Meshing is a complicated and expensive process that requires optimization for speed and accuracy.

2. Object-oriented tools typically use a lumped mass approach for heat transfer when predicting survival. The assumption of using aerothermodynamic databases for primitives works very well with the idea of a lumped mass approach since it allows easy tracking of heat transfer to the object.

3. For non-object oriented applications, the weighting approach for aerothermodynamics as given in eq. (11) provides a sensible and realistic approach for cases where closed-form solutions do not apply.

## 5. VOXELATOR: TEST CASES

In order to validate the methodology and development of the proposed technique, a total of 5 test cases were created with increasing complexity. The simple cases were chosen based on the ability to compute 'true' shading factors for validation. Also, the test cases were created such that each primitive is used at least once. The following test cases were developed and tested:

1. Cube and Cone,

2. Cylinder and Cone,

3. Generic Upper Stage,

4. Generic Spacecraft, and

5. Simplified Deimos 2 satellite

Figure 7 shows the voxelized representation of the test cases developed for the validation of the method. Figures 7a and 7b represent simple combinations of primitives. These are probable combinations that can be expected after the demise of the spacecraft in object oriented tools. Figures 7c and 7d represent generic representations of upper stage and spacecraft, respectively. The voxelized Deimos 2 shown in figure 7e is a simplified derivation of a real spacecraft shown in figure 6.

We perform validation using drag coefficient and make the assumption that the method provides a realistic approach for computing heat transfer. In terms of heat transfer, the approach is expected to work well for cases with only single primitives since the visibility factor will always be unity and the heat transfer will essentially be interpolation of the database. Computing heat transfer for complex objects is high non-trivial because the heat transfer is highly sensitive to mesh generation resolution of the boundary and shock layers. Improper resolution can cause errors of more than 100%. Also, creating a mesh for objects like the upper stage, generic spacecraft, or Deimos 2 with current simulation tools is close to impossible. Validation can be attempted, however, the existing tools need significant advances to have confidence in the validation results for heat transfer.

**Fig. 6**: The Deimos 2 spacecraft

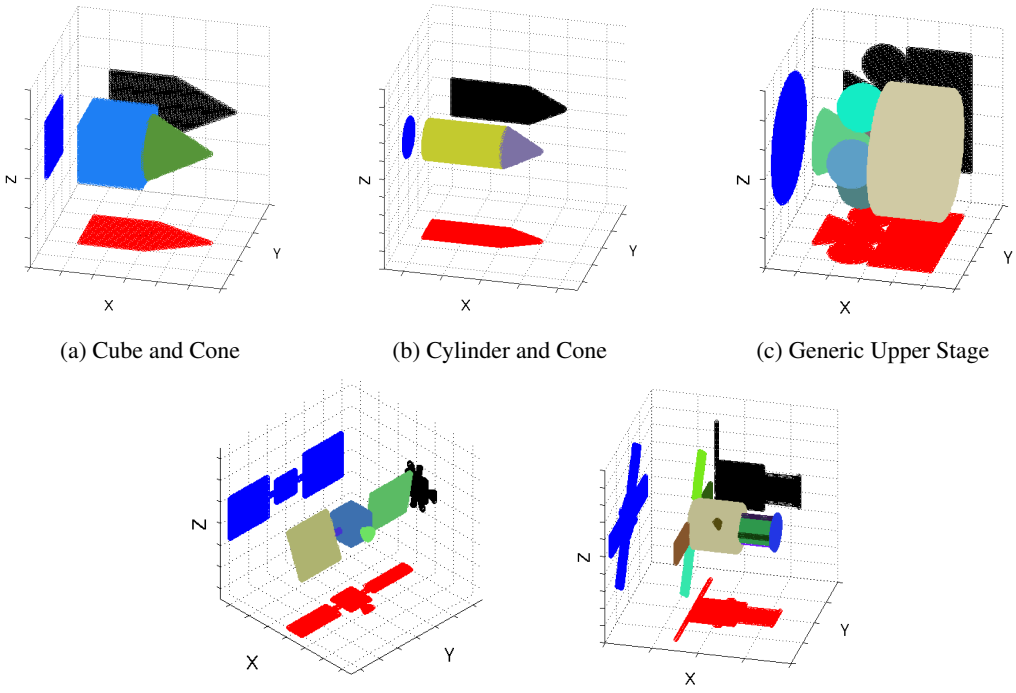
## 6. VOXELATOR: RESULTS

We perform a two-step validation of the method: i) the first step validates the ability of the method to accurately compute visibility factors, and ii) the second step validates the ability to compute parameters of interest. We perform aerodynamic validation with drag coefficients ($C_D$), the 'true' values for which are computed using the standard ray-tracer approach available as part of Deimos' proprietary tool HYDRA [15].

Step 1 validation is performed simultaneously with a resolution study, the purpose of which is the to get a feel for the kind of resolution ratios required for the different primitives for optimization of computational time and accuracy. The resolution study provides insights into what is achievable and should not be taken as recommendation or rule-of-thumb.

After the quantitative validation of the computed visibility factors, we perform a qualitative validation by computing the visibility factors on the grid for the pitch angle varying from -90 to +90 degrees and the yaw angle varying from 0 to 360 degrees with a 10 degree resolution. The visibility factors are expected to exhibit features due to the symmetric nature of the geometries.

Step 2 validation is performed by computing the $C_D$ on the grid with the *Voxelator* and comparing to $C_D$ computed




(a) Cube and Cone (b) Cylinder and Cone (c) Generic Upper Stage

(d) Generic Spacecraft (e) Deimos 2

**Fig. 7**: Test cases developed for the validation of Voxelator

with the standard ray-tracer approach, the difference of which is defined as $C_D$ error. Since, the databases for the $C_D$ of primitives used do not currently exist, the $C_D$ for the primitives are also computed with the standard ray-tracer method.

### 6.1. Cube and Cone

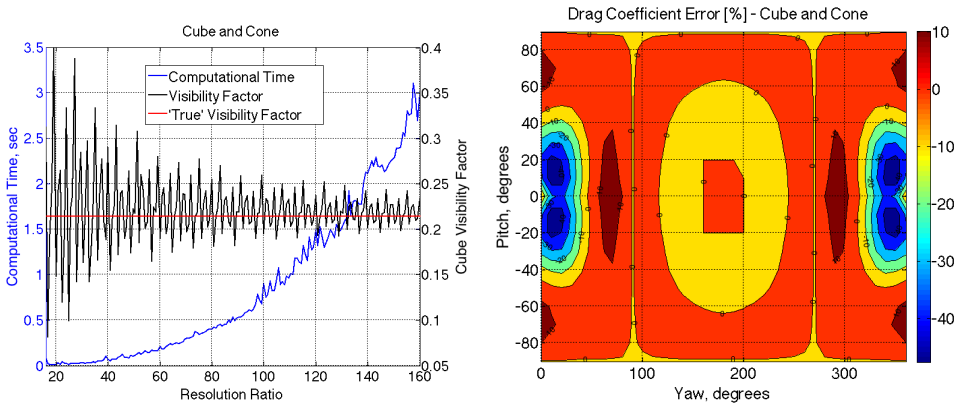
Figure 8a shows the resolution study for the Cube and Cone geometry with a flow direction along -X in the Cartesian coordinates (right to left along X in figure 7a). It is seen that the visibility factor converges onto the 'true' analytic value. The computational time varies between a small fraction of second to a few seconds depending on the resolution ratio. The choice of the resolution ratio is chosen by the user and can be case-specific, however, a quickly conducted (few tens of seconds to few minutes) resolution study can greatly assist the user.

Figure 8b shows the $C_D$ error computed on the grid using the visibility factors given in figure 9. Symmetry is observed in the visibility factors of figure 9 due to the symmetry in the cube-cone geometry. At an attitude (pitch,yaw) [0,0], the cone has a visibility factor of unity whereas the cone has a finite visibility factor equal to the analytic value in figure 8a. At [0,180], the cube has a visibility factor of unity while the cone is completely hidden. At [0,90] both the cube and cone have a visibility factor of unity.

The symmetry in visibility factors in translated to the errors in $C_D$ computed on the grid. For most orientations, the errors in $C_D$ are low except for the region of small changes in pitch and yaw from [0,0]. This is because, for e.g., a small increase in pitch exposes the side face of the cube that increases the visibility factor significantly which providing little contribution to the $C_D$. The error at specific orientations can be high, however, the mean error is about 6.22%, which makes the method applicable under the assumption of random tumbling.

### 6.2. Cylinder and Cone

Figure 10a shows the resolution study for the Cube and Cone geometry with a flow direction along -X in the Cartesian coordinates (right to left along X in figure 7b). It is seen that the visibility factor converges onto the 'true' analytic value. Symmetric patterns in visibility factors (figure 11) and $C_D$ error (10b) similar to those for the cube-cone geometry can be observed again due to symmetry in geometry. The absolute maximum error is almost three times as much as that for the cube-cone geometry and is the largest observed among all the test cases because of a longer side surface leading to a much larger increase in visibility factor at small pitch and yaw angles while making very small contributions to $C_D$. However, even for the cylinder-cone geometry, the mean error is only about 8.28%.




(a) Resolution study

(b) $C_D$ error

**Fig. 8**: Aerodynamic validation for Cube and Cone

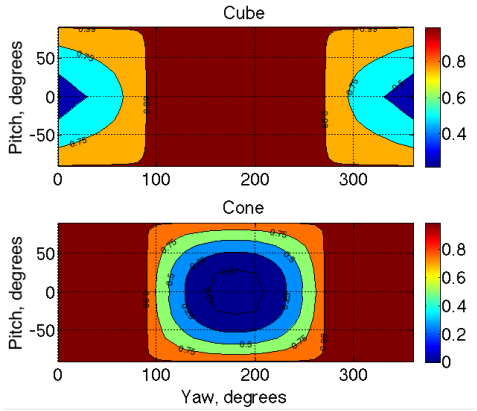



**Fig. 9**: Visibility factors for cube-cone geometry

### 6.3. Upper Stage

Figure 12a shows the resolution study for the upper stage with a flow direction along +X in the Cartesian co-ordinates (left to right along X in figure 7c). It is seen again that the visibility factor converges onto the 'true' analytic value. Figure 13 shows the visibility factors for the primitives that make up the upper stage. The visibility factors for spheres along the Y and Z axes (+ve and -ve) are mirror images of each other. Because the geometry is axially symmetric, the visibility factors for cylinder and cone and the $C_D$ error (figure 12b) are symmetric about 0 degrees in pitch and 180 degrees in yaw. The absolute maximum error in $C_D$ is less than 15% with the mean error only about 3.47% (figure 12b).

### 6.4. Spacecraft

Figure 14a shows the resolution study for the upper stage with a flow direction along -X in the Cartesian co-ordinates (right to left along X in figure 7d) with the visibility factor converging onto the 'true' analytic value. Figure 15 shows the visibility factors for the primitives that make up the generic spacecraft. None of the visibility factors are symmetric in this case because the two solar panels have different relative orientations as can be seen in figure 7d. The absolute maximum error in $C_D$ is about 40% with the mean error only about 4.96% (figure 14b).

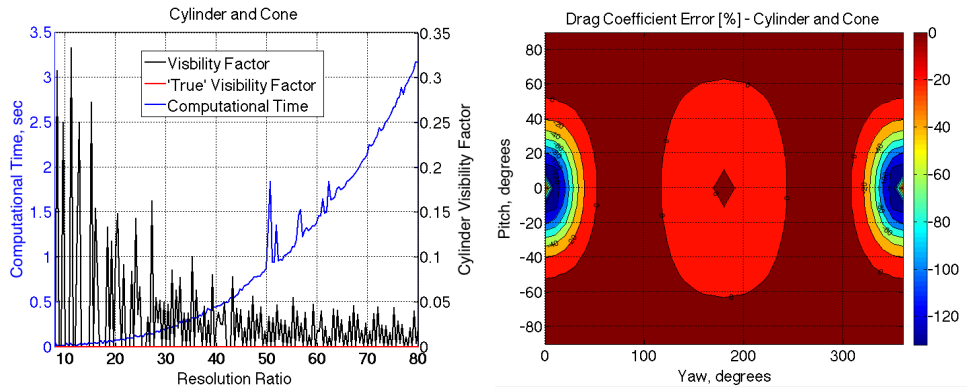
### 6.5. Deimos 2

Figure 16a shows the resolution study for the upper stage with a flow direction along -X in the Cartesian co-ordinates (right to left along X in figure 7e) with the visibility factor converging onto the 'true' analytic value. Comparing figures 6 and 7e shows that because of the limited number of primitives currently modeled in the *Voxelator*, simplifying assumptions were made about the shape of the Deimos 2 geometry. Also, high resolution ratios were used to properly resolve the RCS nozzles and the connectors, however, results showed that the nozzles and connectors play a negligible part in the overall value of the drag coefficients and hence were not taken into account. Therefore, figure 17 does not show the visibility factors for the connectors and nozzles. The absolute maximum error in $C_D$ is about 23% with the mean error only about 4.46% (figure 16b).

## 7. CONCLUSIONS AND FUTURE WORK

The uncertainties involved in the modeling and simulation of the re-entry warrants a probabilistic approach. Therefore, methods that can provide quick and accurate predictions for aerodynamic and aerothermodynamic properties are desired. The goal of this work is to develop and present computer-graphics inspired tools that can be applied towards aerodynamic and aerothermodynamic modeling for object during atmospheric re-entry.

Closed for solutions for aerodynamics and aerothermodynamics for a one-sided flat plate can be applied to the re-




(a) Resolution study (b) $C_D$ error

**Fig. 10**: Aerodynamic validation for cylinder-cone geometry

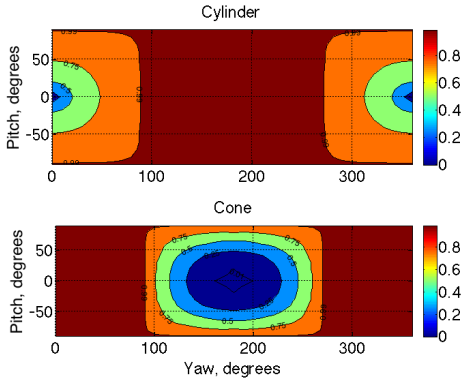



**Fig. 11**: Visibility factors for cylinder-cone geometry

entry object if the object is modeled as a mesh with triangular facets. However, these solutions have assumptions and limitations, especially for aerothermodynamics. To date, aerothermodynamics poses a great challenge in the flow regimes encountered by an object undergoing re-entry.

As part of this work, we first develop a new ray-tracing approach called *Pixelator*. The approach is much more efficient than the traditional ray-tracing approach because it takes away the need for computing expensive ray-facet intersections. Implemented in MATLAB®, the *Pixelator* is close to an order of magnitude faster than the traditional ray-tracer implemented in the language *C*. The *Pixelator* efficiency can only improve if implemented in *C*, we expect and improvement of at least an order of magnitude.

Next, we develop and new and novel method for modeling aerodynamics and aerothermodynamics called *Voxelator*. The method uses voxelized primitives to create the complex object of interest. The approach uses aerodynamic and aerothermodynamic databases for the primitives for contribution of each primitive and weights them with the visibility factors computed by the *Voxelator* to give the solution for the overall shape as observed from the direction of the flow.

In this work, we validate the *Voxelator* with several different test cases. Validation is performed only for the aerodynamics as computing the aerothermodynamic solutions accurately for the complex shapes used in the validation is almost impossible with the current simulation techniques. Hence, we assume that the proposed methods gives the best and most physical estimate for the heat transfer. We use the aerodynamic values obtained with the traditional ray-tracer as the 'true' solution.

The validation for the test cases show that the error can be quite high for limited geometries at certain attitudes, however, in all cases the mean error as a function of attitude is always less than 10%. This combined with the assumption of random tumbling of the re-entering object provide and fast and accurate method to the aerodynamic and aerothermodynamic solutions.

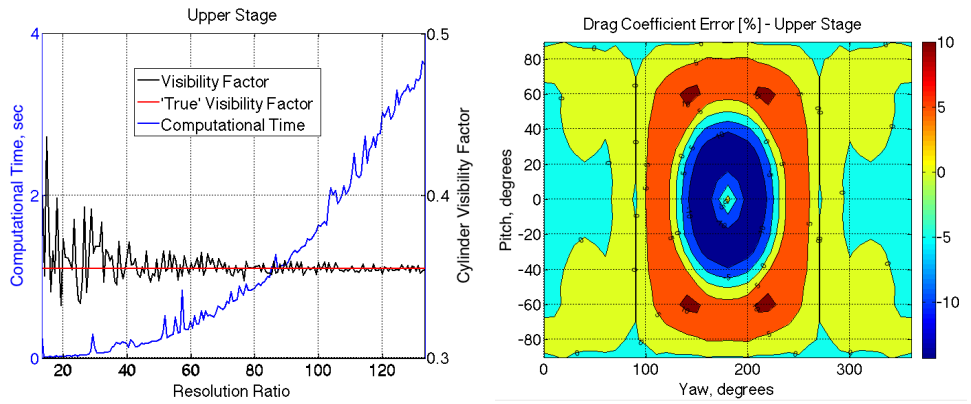
In its present version, the tool can model a limited number of primitives. Future work includes adding the capability to model more primitives for more realistic representations of the complex shapes. Future work also includes creating the aerodynamic and aerothermodynamic databases for the primitives used in the tool. Also, further improvement in performance can be attained if the tool is ported over to a much faster language like *C*.

# References

[1] D. J. Kessler and B. G. Cour-Palais, "Collision frequency of artificial satellites: The creation of a debris belt," *Journal of Geophysical Research*, vol. 83, no. A6, pp. 2637–2646, June 1978.

[2] NASA-STD-8719.14., "Process for limiting orbital debris," May 2012, Revision A Change 1.

[3] "Requirements on space debris mitigation for esa projects," April 2008, European Space Agency.




(a) Resolution Study (b) $C_D$ error

**Fig. 12**: Aerodynamic validation for Upper Stage

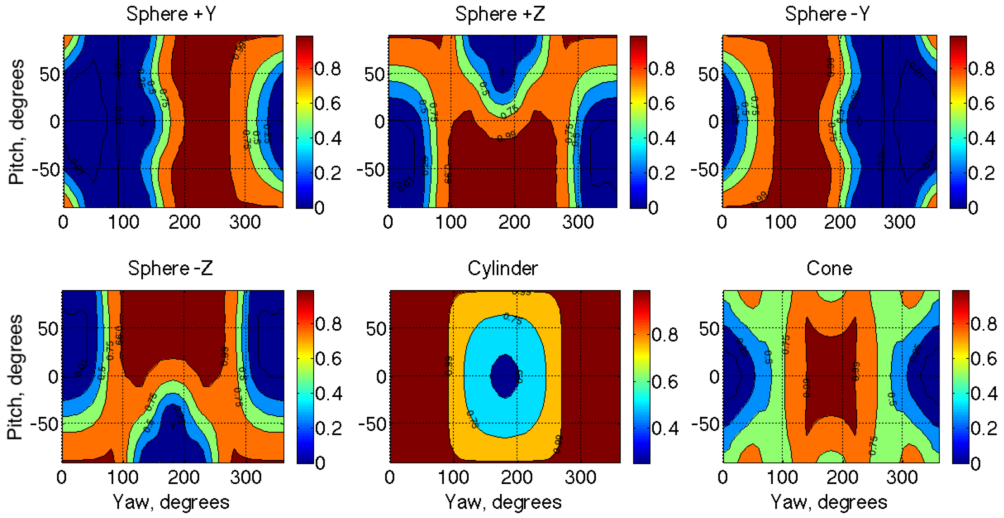



**Fig. 13**: Visibility factors for Upper Stage


[4] G. Koppenwallner, B. Fritsche, T. Lips, and H. Klinkrad, "Scarab a multi-disciplinary code for destruction analysis of spacecraft during re-entry," in *5th European Symposium on Aerothermodynamics of Space Vehicles*, Cologne, Germany, November 2005.

[5] W. C. Rochelle, B. S. Kirk, and B. C. Ting, "Users guide for object reentry analysis tool (orsat)," Tech. Rep. Version 5.0, NASA Lyndon B. Johnson Space Center, 1999, JSC-28742.

[6] C. Martin, C. Brandmueller, and K. Bunte et. al., "A debris risk assessment tool supporting mitigation guidelines," in *4th European Conference on Space Debris*, ESA/ESOC, Darmstadt, Germany, April 2005, ESA SP-587.

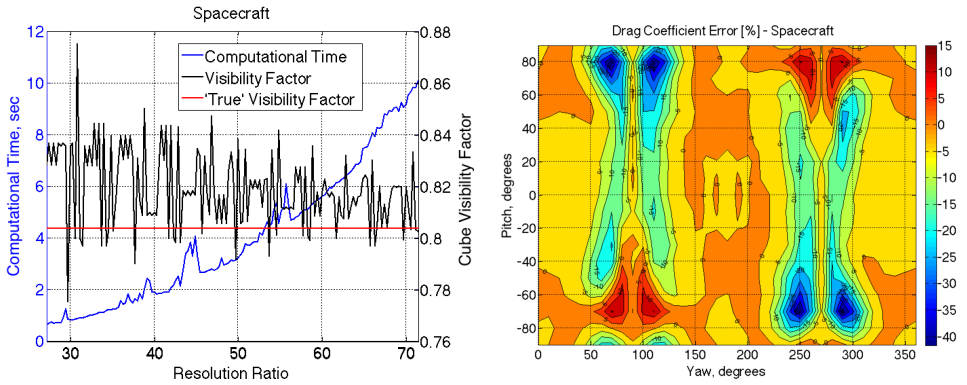
[7] Cristina Parigini, Irene Pontijas Fuentes, Rodrigo Haya Ramos, and Stefania Cornara, "Debris tool and its use in mission analysis activities," in *8th ESA symposium on aerothermodynamics of space vehicles*, Lisbon, Portugal, March 2015.

[8] I. Newton, *Principia - Motes Translation Revised*, University of California Press, 1946.

[9] S. A. Schaaf and P. L. Chambre, *Flow of Rarefied Gases, High Speed Aerodynamics and Jet Propulsion*, pp. 1–55, Princeton Univ. Press, 1958.

[10] R. G. Wilmoth, R. A. Mitcheltree, and J. N. Moss, "Low density aerothermodynamics of the stardust sample return capsule," *Journal of Spacecraft and Rockets*, vol. 36, no. 3, pp. 436–441, 1999.

[11] Piyush M. Mehta, Edmondo Minisci, Massimiliano Vasile, Andrew Walker, and Melrose Brown, "An open source hypersonic aerodynamic and aerothermodynamic modeling tool," in *8th ESA symposium on*




(a) Resolution Study (b) $C_D$ error

**Fig. 14**: Aerodynamic validation for Generic Spacecraft

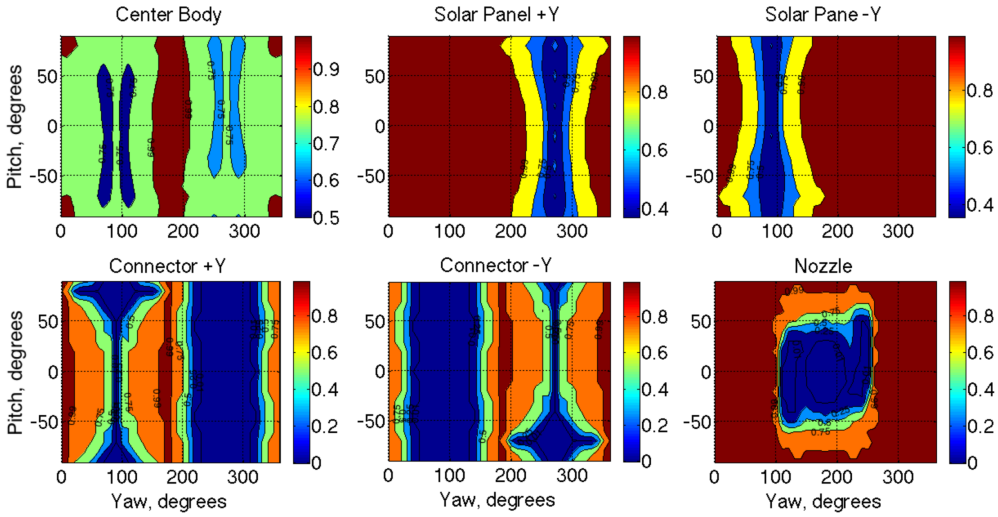



**Fig. 15**: Visibility factors for Generic Spacecraft

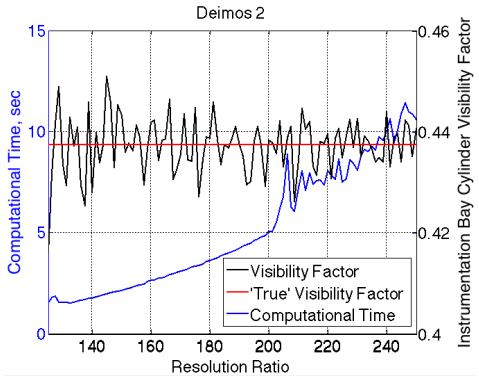
*aerothermodynamics of space vehicles*, Lisbon, Portugal, March 2015.

[12] J. A. Fay and F. R. Riddell, "Theory of stagnation point heat transfer in dissociated air," *Journal of the Aeronautical Sciences*, vol. 25, no. 2, pp. 73–85, 1958.

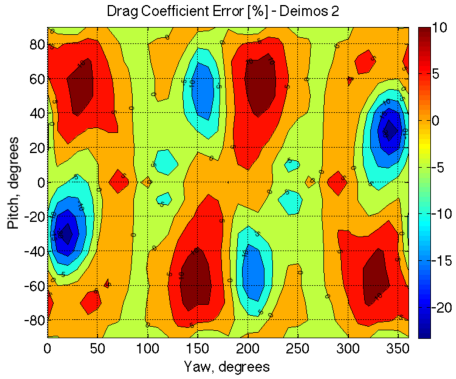
[13] F. J. Regan and S. M. AnandaKrishnan, *Dynamics of Atmospheric Re-Entry*, AIAA Education Series, 1993.

[14] H. Legge, "Hypersonic approximations for heat transfer and shear stress applied to continuum and rarefied plume impingement," Tech. Rep., 1987, DFVLR-IB 222-87 A23.

[15] B. Davide, C. Parigini, and G. Zaiacomo et. al., "Petbox: Flight qualified tools for atmospheric flight," in *6th International Conference on Astrodynamics Tools and Techniques (ICATT)*, ESOC, Darmstadt, Germany, March 2016.




(a) Resolution Study




(b) $C_D$ error

**Fig. 16**: Aerodynamic validation for Deimos 2

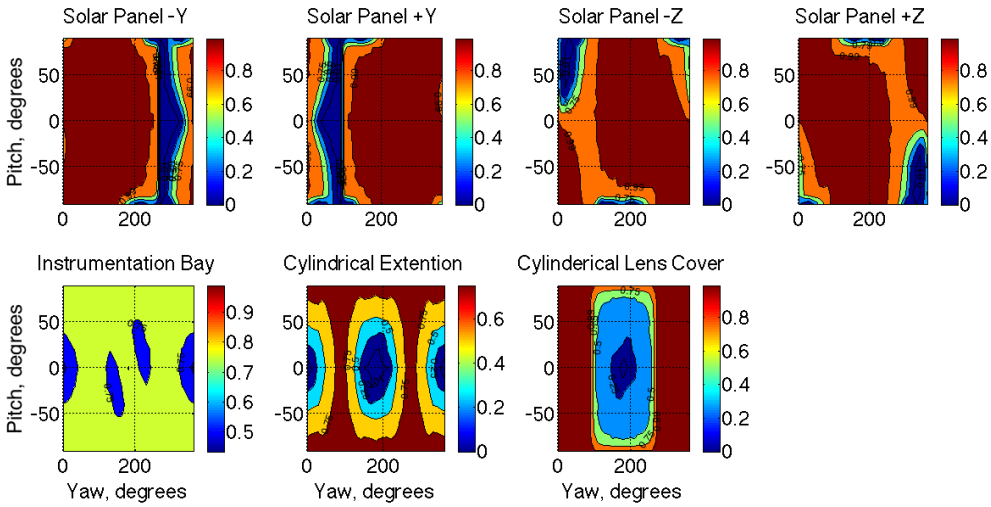



**Fig. 17**: Visibility factors for Deimos 2