An Efficient Ray-Tracing Method for Determining Terrain Intercepts in EDL Simulations

Jeremy Shidner
Analytical Mechanics Associates, Inc.
21 Enterprise Parkway, Suite 300
Hampton, VA 23666-6413
757-865-0000
jeremy.d.shidner@nasa.gov

Abstract—**The calculation of a ray's intercept from an arbitrary point in space to a prescribed surface is a common task in computer simulations. The arbitrary point often represents an object that is moving according to the simulation, while the prescribed surface is fixed in a defined frame. For detailed simulations, this surface becomes complex, taking the form of real-world objects such as mountains, craters or valleys which require more advanced methods to accurately calculate a ray's intercept location. Incorporation of these complex surfaces has commonly been implemented in graphics systems that utilize highly optimized graphics processing units to analyze such features. This paper proposes a simplified method that does not require computationally intensive graphics solutions, but rather an optimized ray-tracing method for an assumed terrain dataset. This approach was developed for the Mars Science Laboratory mission which landed on the complex terrain of Gale Crater. First, this paper begins with a discussion of the simulation used to implement the model and the applicability of finding surface intercepts with respect to atmosphere modeling, altitude determination, radar modeling, and contact forces influencing vehicle dynamics. Next, the derivation and assumptions of the intercept finding method are presented. Key assumptions are noted making the routines specific to only certain types of surface data sets that are equidistantly spaced in longitude and latitude. The derivation of the method relies on ray-tracing, requiring discussion on the formulation of the ray with respect to the terrain datasets. Further discussion includes techniques for ray initialization in order to optimize the intercept search. Then, the model implementation for various new applications in the simulation are demonstrated. Finally, a validation of the accuracy is presented along with the corresponding data sets used in the validation. A performance summary of the method will be shown using the analysis from the Mars Science Laboratory's terminal descent sensing model. Alternate uses will also be shown for determining horizon maps and orbiter set times.**

TABLE OF CONTENTS

978-1-4673-7676-1/16/$31.00 ©2016 IEEE

1. INTRODUCTION

The calculation of range from an arbitrary point to a surface is a common task in computer simulations. For simple surfaces, such as spheres, the intercept point is easily calculated from geometric relationships between point and sphere. Such geometric relationships are used by computer simulations to inform models (atmosphere, radar, etc...) of the shape of the terrain. However, finding an intercept from an arbitrary point to a surface cannot always rely on simple geometric relationships. The intercept must be determined by searching the arbitrary surface. For the case presented, the intersection point is assumed to be on a Digital Elevation Model (DEM), that defines a planetary surface for use in a computer trajectory simulation. Under this assumption, the search to find the intercept must therefore adhere to the following rules:

1. The intercept must be bounded by the closest data points.
2. The surface normal is greater than 90 degrees opposite to the pointing vector.

The ray-tracing method presented in this paper has been implemented and validated using the Program to Optimize Simulated Trajectories II (POST2 [1]). The various surfaces used come from NASA missions designed to map the surface of various planets including Venus [[2]], Mars [[3]], Earth [[4]], and the Moon [[5]] (Figure 1).

2. POST2 DESCRIPTION

The POST2 simulation is a generalized point mass simulation that allows definition of multiple phases, using discrete events, for trajectory analysis. For Entry, Descent and Landing (EDL) applications, the versatility and heritage of POST2 is leveraged by adding routines specific to each EDL mission such as atmosphere, aerodynamics, thrust, terrain, planet and vehicle models to multiple vehicles in a single simulation. These models are important for understanding end-to-end EDL performance related to such items as parachute deploy, inflation, heatshield jettison, powered descent, and touchdown detection. The POST2 simulation can be used to support Monte Carlo analysis by simulating thousands of different initial conditions with random spacecraft and environmental perturbations, generating statistics, and evaluating trends and performance.

The ray-tracing method discussed in this paper is an integral piece of the POST2 terrain model. Terrain is often delivered as a binary file that represents the surface of a planetary body that the simulation can use to support various analyses. The POST2 software utilizes the terrain to determine atmosphere look-up altitude, vehicle altitude, terminal descent sensor range, and surface definition for vehicle ground-contact in-

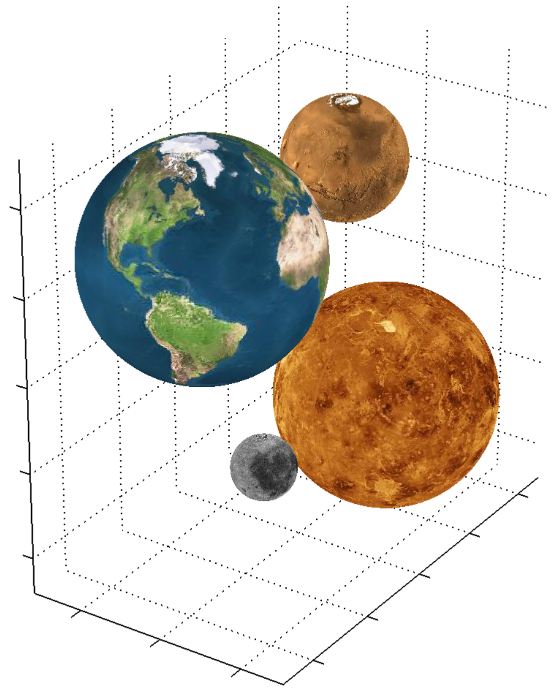

Figure 1. Surfaces Used in Ray-Tracing Intercept Method

teraction modeling. An example of these aspects of terrain usage is shown in Figure 2 of the Mars Science Laboratory touchdown animation. The dials around the edge of the image represent quantities such as vehicle altitude, radar derived altitude, vertical speed relative to the terrain surface and Mach number as derived from the atmosphere model and vehicle velocity. All of these quantities are dependent on results from the terrain model.

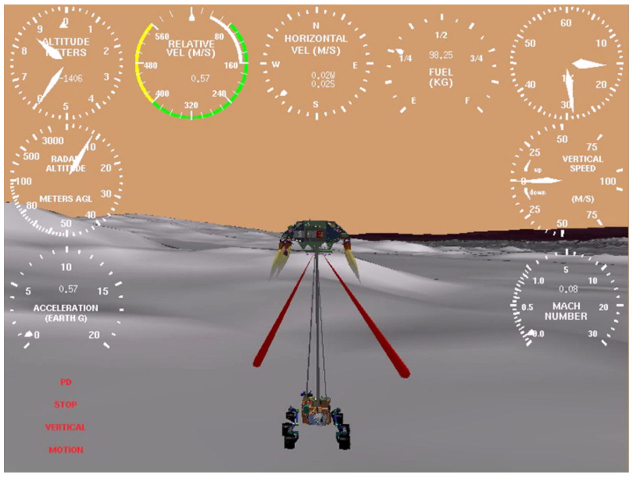

Figure 2. Terrain Interactions in POST2

The terrain provides varying levels of fidelity to challenge the various models by including more variability among data posting over shorter length scales. The POST2 terrain model accomplishes variable terrain fidelity by allowing multiple layers of terrain to be loaded into the simulation. Each layer can be searched according to a hierarchy indicated by the user. Therefore, highly detailed terrain models can be loaded when the vehicle is near the ground while less detailed terrain models can be utilized when the vehicle is at higher altitudes, as shown in Figure 3, where parachute deploy is indicated by ”PD” and backshell separation is indicated by ”BS”. Prior to parachute deploy, the simulation only needs a coarsely defined terrain model (450 m spacing between postings), indicated by the layer that is colored in red, yellow and blue. After parachute deploy, the simulation requires more definition to support radar sensor modeling (10 m spacing between postings) and the brown colored layer is utilized. After backshell separation, the simulation requires the highest terrain definition (2 m spacing between postings) to support the touchdown interaction with variable surface slopes, using the grey colored layer.

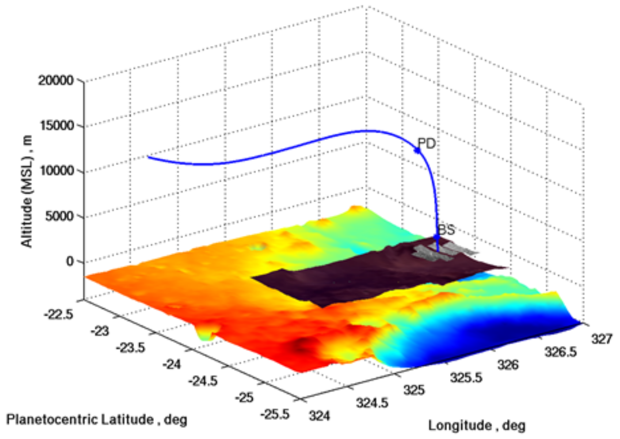

Figure 3. Terrain Layering in POST2

All of these quantities rely on a ray-trace algorithm that can determine the range from any point on the vehicle (center of gravity, vehicle touchdown supports, etc...) to the arbitrary terrain surface along a specified pointing vector. The determination of that range needs to be as efficient as possible since the simulation utilizes terrain lookups millions of times in a single EDL simulation and billions of times across a typical EDL Monte Carlo simulation. The formulation of this calculation will be discussed next.

3. FORMULATION

The ray-tracing algorithm relies on two inputs, a planet relative position ($\vec{S}$) and pointing vector ($\vec{P}$). From these two vectors, the cross product yields the normal to the plane along which the ray travels:

$$\vec{n} = \vec{S} \times \vec{P} \tag{1}$$

The normal defines the ray-tracing plane. The traversal of this plane is handled by moving from one triangle to the next that are intersected by the plane. Figure 4 shows the triangles that are intersected by the plane where the red and dashed green triangles represent the searchable surfaces from the terrain dataset. The search continues along the red and dashed green triangles until an intersection of the pointing vector on the surface is found.

The following section will describe the derivation of this algorithm by first describing the assumed data format and

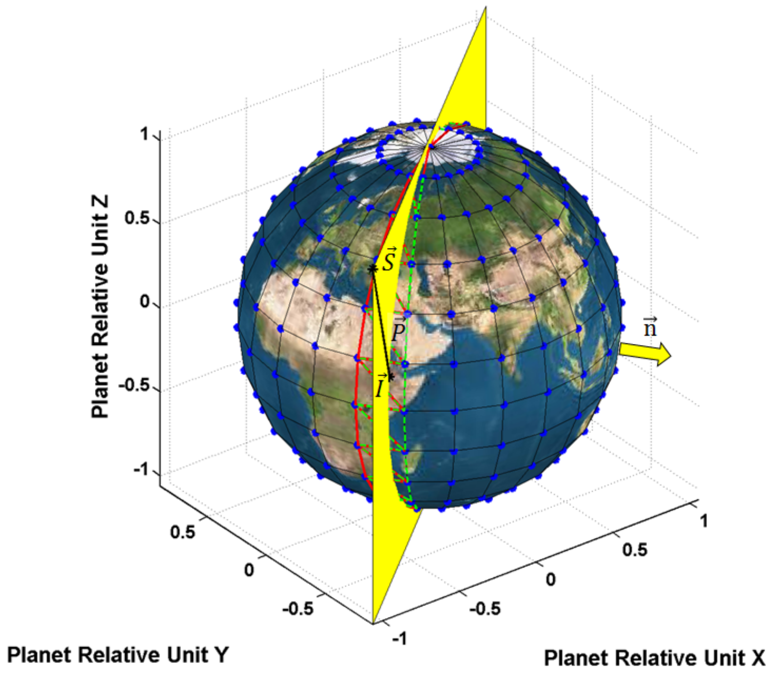

Figure 4. **Ray-Tracing Plane**

the respective conversions to a planet-fixed cartesian frame. Second, the initialization of the ray-trace search will be presented by discussing various methods that help to optimize the efficiency of the search. Third, the determination of whether or not the ray intersects the triangle of interest will be presented. Finally, the progression of the search along the ray-trace plane will be discussed.

Data Usage

To understand the derivation of the ray-tracing algorithm, the data format of the surface must first be presented (Figure 5). The DEM data sets used are typically stored in binary format relative to an equidistant grid in planetocentric longitude (λ) and planetodetic latitude (ϕ). The grid values (h) are typically in units of meters and offset from a reference ellipsoid as defined by the data producers in terms of the semi-major axis (a) and semi-minor axis (b). To calculate the radius vector from the grid values to planet centered coordinates, the planetocentric latitude ($\phi\prime$) and radius (r) of the specified grid point are first determined by the following equations

$$f = b^2/a^2 \tag{2}$$

$$C = \frac{1}{\sqrt{\cos^2\phi + f\sin^2\phi}} \tag{3}$$

$$x = \cos\phi(aC + h) \tag{4}$$

$$z = \sin\phi(afC + h) \tag{5}$$

$$r = \sqrt{x^2 + z^2} \tag{6}$$

$$\phi' = \arcsin(\frac{z}{r}) \tag{7}$$

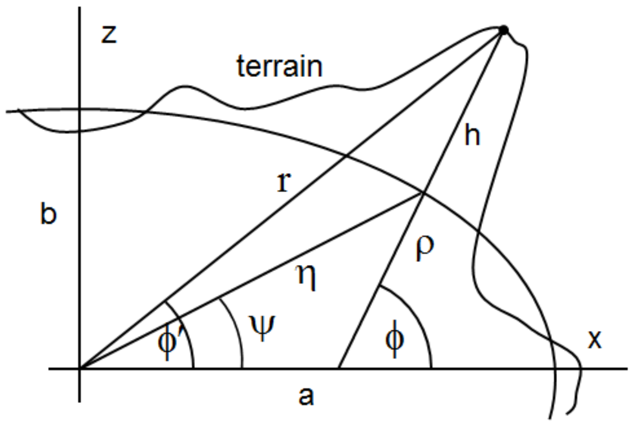

Figure 5. Assumed DEM Data Format

Using the planetocentric latitude, longitude, and radius of the grid point, the vector to a DEM data point is determined

$$\vec{r} = \begin{bmatrix} \cos\lambda\cos\phi' \\ \sin\lambda\cos\phi' \\ \sin\phi' \end{bmatrix} \tag{8}$$

From the above equations, three vectors can be determined within the DEM grid to assemble the triangles used in the ray-tracing search.

Initializing the Ray-Trace

Before defining the ray-tracing plane, the search method needs to be well defined. The normal vector to the plane is only well defined if it is not parallel to the position vector. Therefore, the dot product of the unit normal vector and position vector can not equal negative one within a tolerance of machine precision (ε).

$$-1 - \varepsilon < \vec{S} \cdot \vec{P} < -1 + \varepsilon \tag{9}$$

If the dot-product is equal to negative one, the solution can be determined without any search necessary, since the pointing vector will pass through the triangle of the starting point.

Once the ray-tracing plane is well defined, the ray-tracing search needs to be initialized. There are three initialization options that all have applicability depending on the problem being analyzed. The first option is to use the current planetocentric latitude and longitude of the planet relative position $\vec{S}$. This option is useful when determining line of sight to terrain features. The second option is to determine the intercept of the ray on an ellipsoid that best fits the surface, using the planetocentric latitude and longitude of the intercept. The third option is to start the search at the last intercept determined by the algorithm for the previous location and pointing vector using the closest valid planetocentric latitude and longitude on the ray-trace plane. If there is no previous intercept for the third option to use, the algorithm reverts to the second option. If the spacecraft is enclosed by the best-fit ellipsoid, then the algorithm reverts to the first option.

The intercept ($\vec{I}$) on the best-fit ellipsoid, depicted in Figure 6 and described using equations 10-13, is determined from the position vector ($\vec{S}$), pointing vector ($\vec{P}$), semi-major axis (a) of the best-fit ellipsoid, and semi-minor axis (b) of the best-fit ellipsoid.

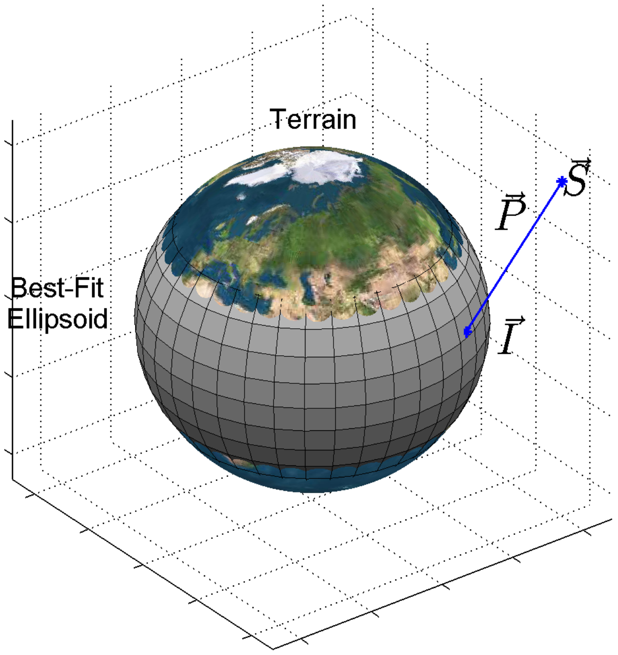

Figure 6. Ellipse Intercept

$$A = P_x^2 + P_y^2 + \frac{a^2}{b^2} P_z^2 \tag{10}$$

$$B = 2(S_x P_x + S_y P_y + \frac{a^2}{b^2} S_z P_z) \tag{11}$$

$$C = S_x^2 + S_y^2 + \frac{a^2}{b^2} S_z^2 - a^2 \tag{12}$$

$$\vec{I} = \vec{S} + \frac{-B - \sqrt{B^2 - 4AC}}{2A} \vec{P} \tag{13}$$

The ellipse intercept function will not return an intercept ($\vec{I}$) if either of two conditions are true. First, no intercept is returned if the dot-product of the position vector and pointing vector is positive. This assumes that the spacecraft is pointing away from the surface, and therefore, there is no intercept. Note that this may not be desirable if the spacecraft is traveling near a pronounced surface where an intercept exists on the side of the surface, just above the local horizontal. In this case, the first or third starting option should be used to find the intercept. The second condition is that if $B^2 \leq 4AC$, no intercept is returned because, mathematically, there is no real intersection with the ellipsoid.

The third option starts the search based on the last intersection point. Care needs to be taken in selecting this starting point, since the last intersection likely does not lie on the ray-trace plane (a requirement for the ray-trace search). Therefore, the starting point on the ray-trace plane is the point that is the least distance from the last intercept to the ray-trace plane, Figure 7. The equation for this distance is a function of the longitude of the ascending node of the ray-trace plane (Ω), inclination of the ray-trace plane (i), planetocentric latitude of the last intercept (ϕ'), longitude of the last intercept (λ) and angular distance (θ) from the x-y plane to the new starting point on the ray-trace plane.

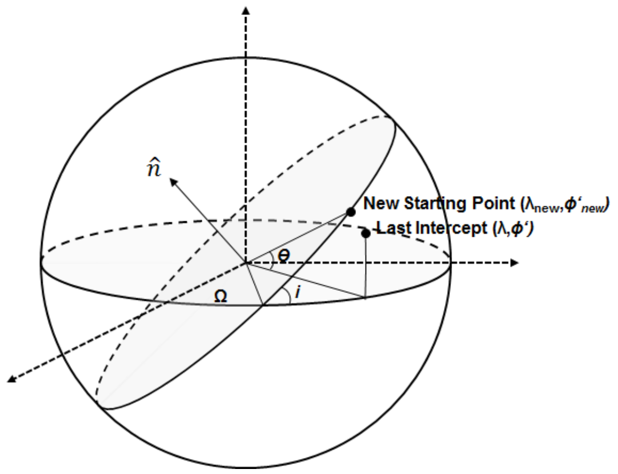
$\hat{n}$
New Starting Point (λ_{new},ϕ'_{new})
Last Intercept (λ,ϕ')
θ
Ω
i

Figure 7. Starting Point

$$\Omega = atan2(n_z, -n_y) \tag{14}$$

$$i = \arccos n_z \tag{15}$$

$$\tan\theta = \frac{\sin i \sin\phi' + \cos\phi' \cos i \sin(\lambda - \Omega)}{\cos\phi' \cos(\lambda - \Omega)} \tag{16}$$

$$\sin\phi'_{new} = (\sin\theta \sin i) \tag{17}$$

$$\tan\lambda_{new} = \left(\frac{\sin\Omega\cos\theta + \cos\Omega\sin\theta\cos i}{\cos\Omega\cos\theta - \sin\Omega\sin\theta\cos i}\right) \tag{18}$$

The resulting planetocentric latitude and longitude can be used as the starting point to start the ray-trace.

Triangle Intersection Test

The triangle intersection test is used at each point in the search to test for convergence of the ray on a surface. The intersection test is taken from [6] and illustrated in Figure 8.

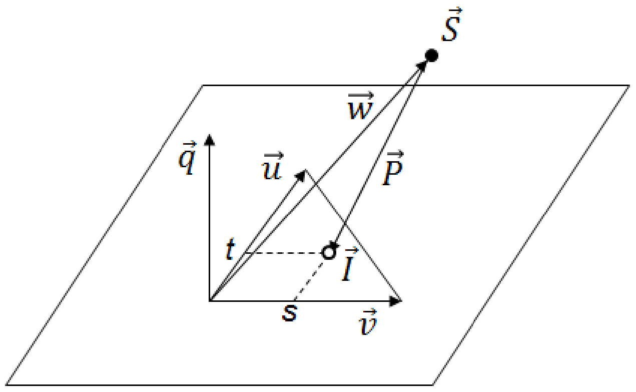

Figure 8. Ray-Triangle Geometry

The intersection test starts by calculating the two vectors $\vec{u}$ and $\vec{v}$ from the three ground point vectors $\vec{V_0}$, $\vec{V_1}$, and $\vec{V_2}$. The surface normal to the triangle plane, $\vec{q}$ is determined from the cross product of $\vec{u}$ and $\vec{v}$. Note that if the dot product of the surface normal and ray is zero, the ray is parallel to the triangle plane. The intercept on the plane of the triangle is calculated.

$$\vec{I} = \vec{S} + \frac{\vec{q}\cdot\vec{w}}{\vec{q}\cdot\vec{P}}\vec{P} \tag{19}$$

The parametric coordinates, s and t, of the intercept are then computed. If the values of s and t are both greater than or equal to 0, and their sum is less than or equal to 1, then the intersection is contained by the triangle.

$$s = \frac{(\vec{u}\cdot\vec{v})(\vec{w}\cdot\vec{v}) - (\vec{v}\cdot\vec{v})(\vec{w}\cdot\vec{u})}{(\vec{u}\cdot\vec{v})^2 - (\vec{u}\cdot\vec{u})(\vec{v}\cdot\vec{v})} \tag{20}$$

$$t = \frac{(\vec{u}\cdot\vec{v})(\vec{w}\cdot\vec{u}) - (\vec{u}\cdot\vec{u})(\vec{w}\cdot\vec{v})}{(\vec{u}\cdot\vec{v})^2 - (\vec{u}\cdot\vec{u})(\vec{v}\cdot\vec{v})} \tag{21}$$

$$Intersection = \begin{cases} True & \text{if } s \geq 0 \text{ and } t \geq 0 \text{ and } s+t \leq 1 \\ False & \text{if } s < 0 \text{ or } t < 0 \text{ or } s+t > 1 \end{cases} \tag{22}$$

Following the Ray-Trace Path

The search for an intersect is performed by evaluating first if the current triangle at the starting planetocentric latitude and longitude contains the intersection point. If not, the search first evaluates the dot product of the three triangle points with the ray-trace plane normal vector. If the dot product is positive, the triangle point is on one side of the ray-trace plane, while if the dot product is negative, the triangle point is on the opposite side. The dot product results of the three triangle points with the ray-trace plane normal vector are used to logically inform the progression of the ray-trace search. When no intersection is found, the search is appropriately incremented such that the next increment in the three triangle points bound the ray-trace plane.

4. IMPLEMENTATION

The ray-tracing algorithm has been coded in the C programming language. Special care has been given to creating a common data structure that can be replicated for multiple terrain layers. Each grid can be correlated to one another in relation to values such as height, roughness, reflectivity, absorption, etc. These data structures are independent of each other in the simulation so that varying levels of fidelity can be loaded per structure. This design, combined with the speed of C, provides an easily portable implementation for use across a variety of simulation models.

The gridded data products are typically given in a planet relative frame as defined by the International Astronomical Union ([7]). The IAU fits the planet motion to a set of coefficients that describe planet pole rotation, precession, and nutation. To date, POST2 only allows definition of polar rotation, so latitudinal effects are not accurately captured in the grid heights relative to the simulated state. However, the effects of precession and nutation are almost negligible on short time scales (analyses that are not long duration orbital missions). Therefore, comparison of the POST2 planet relative frame and the grid's reference frame is acceptable for use on the scales of short simulation run times.

The inputs expected by the algorithm are planet relative position and pointing vector. With respect to the intersection point, the outputs are range, planet relative position, and surface normal. Intersection options are available so that the intersection is on one of two possible triangular planes given the four points of a gridded dataset that bound the intersection.

In order to optimize the calls to the ray-tracing algorithm, the data structures are made available to other routines so that the last intersect point can be loaded as an initial guess. However, the same type of logic applies internal to the ray-tracing algorithm such that if the algorithm is called twice in a row with the same inputs, an internal data structure records the first intersection so that the second is updated with the same answer without calling the algorithm.

Run time is also reduced by using the Network Common Data Form (NetCDF) to store and read the data. "NetCDF is a set of software libraries and machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data ([8])". The NetCDF format allows loading of subsets of terrain such that only relevant data is loaded.

Data Sets

The datasets presented within this paper were provided from the Mars Orbiter Laser Altimiter (MOLA) Mission Experiment Gridded Data Records (MEGDR) at 128 degree resolution [3] for evaluating planet scale terrain. The USGS Astrogeology Center provided the Gale Crater landing site

DEM [9]. The NetCDF format allows storage of identifying information that is listed in tables 1 and 2 below.

Table 1. MOLA 1/128 Degree Resolution Dataset

Parameter	Value
FILE	Mars128degr.nc
TITLE	Mars MOLA Data Set / 128 Degree Resolution / Radius
FRAME	Reference Coordinate Frame: Mars IAU 2000
CREATION	Created on Mon Oct 29 15:55:25 EDT 2007 by Jeremy Shidner (Jeremy.D.Shidner@nasa.gov).
SOURCE	Downloaded from http://pds-geosciences.wustl.edu/geo/mgs-m-mola-5-megdr-l3-v1/mgsl_300x/meg128/
FORMAT	shorti
RESOLUTION	1/128 deg
SCALE FACTOR	1.0
NULL VALUE	32767
REFERENCE SEMI-MAJOR AXIS	3396000 meters
REFERENCE SEMI-MINOR AXIS	3396000 meters
BEST FIT SEMI-MAJOR AXIS	3395959.679951 meters
BEST FIT SEMI-MINOR AXIS	3376846.911013 meters
MAXIMUM LATITUDE	87.996094 deg
MINIMUM LATITUDE	-87.996094 deg
MAXIMUM LONGITUDE	359.996094 deg
MINIMUM LONGITUDE	0.003906 deg

The FRAME parameter identifies the planet-fixed reference frame of the dataset's reference ellipsoid. The REFERENCE SEMI-[MAJOR,MINOR] AXIS defines the reference ellipsoid used to define the assumed dataset format as described in section 3. The BEST FIT SEMI-[MAJOR,MINOR] AXIS defines the ellipsoid that most closely fits the dataset and is used to determine a starting point in the ray-trace search algorithm.

The [MAXIMUM,MINIMUM] [LATITUDE,LONGITUDE] represents the bounds of the dataset that are available. The RESOLUTION defines the angular spacing between data postings. The NULL VALUE is used to represent dataset values where no data is available.

The FORMAT parameter defines the variable size of the DEM. Possible values for FORMAT include shorti (16 bit - short int), float (32 bit), and double (64 bit). By utilizing different FORMATs, the DEM size can be reduced at the cost of accuracy. The SCALE FACTOR is the multiplier on the dataset values and can be used to recover some accuracy for FORMATs such as shorti.

Table 2. Gale Crater Dataset

Parameter	Value
FILE	Gale CTX_HiRISE Mosaic at 2 meters resolution v7.2.nc
TITLE	Gale HiRISE/CTX Mosaicked Digital Elevation Model 4/4/12 Update
FRAME	Mars IAU 2000
CREATION	04-Apr-2012
SOURCE	Edited and Converted (Devin Kipp) from USGS HiRISE/CTX Mosaic (Randy Kirk and Fred Calef)
FORMAT	shorti
RESOLUTION	1/29635.881752 deg
SCALE FACTOR	0.1
NULL VALUE	32767
REFERENCE SEMI-MAJOR AXIS	3393000 meters
REFERENCE SEMI-MINOR AXIS	3393000 meters
BEST FIT SEMI-MAJOR AXIS	3386018.031334 meters
BEST FIT SEMI-MINOR AXIS	4952639.365362 meters
MAXIMUM LATITUDE	-4.248966 deg
MINIMUM LATITUDE	-4.925477 deg
MAXIMUM LONGITUDE	137.729815 deg
MINIMUM LONGITUDE	137.122477 deg

Optimization

As described in the formulation section, the initialization of the ray-trace search is important in optimizing the algorithm efficiency. A series of tests were run to determine the number of search iterations the algorithm performed across a single EDL run of the Mars Science Laboratory POST2 simulation. The simulation started from cruise stage separation and ended at the completion of the fly-away maneuver of the descent stage. The simulation takes about 5.5 minutes to run for a 970 second trajectory. The tests varied by:

1. Beginning the ray-trace search at the origin $\vec{S}$.
2. Begin the ray-trace search at the intersection $\vec{I}$ on the best-fit ellipsoid.
3. Begin the ray-trace search at the last intersection $\vec{I}$ to occur.

The tests were run on an Intel(R) Core(TM) i7-3720QM CPU @ 2.60 GHz with 8 GB of RAM. The results of the test for the test cases listed above are shown in Table 3 with the total runtime shown in column 2. Note that these are single runs and do not represent an average of the expected performance. The number of times the ray-trace algorithm was called is shown in column 3. The GNU gprof utility program was utilized to statistically determine the percent runtime consumed by the ray-trace algorithm with respect to other models executed in POST2 as shown in column 4.

Table 3. **Algorithm Performance in POST2**

Test Case	Runtime	Iterations	% of Run
1. Origin	1598 sec	265e6	70.86 %
2. Best Fit Intercept	593 sec	86.1e6	36.21 %
3. Last Intercept	333 sec	17.2e6	6.62 %

The results show that without the optimization of the routine, the simulation would be spending over 70% of its runtime in just the terrain model alone. When running Monte Carlo simulations, this type of inefficiency is amplified by the larger number of cases to run, making the model unsustainable for large-scale analysis. Overall, the optimizations of the ray-trace starting location is shown to have a dramatic improvement on the runtime, and number of ray-trace search iterations required.

A second test was run by turning off the feature that retains the last terrain request. This feature reduces redundancy by not evaluating the same inputs twice. By turning this feature off, the efficiency of programs calling the terrain model in POST2 (atmosphere, vehicle altitude, radar, etc...) can be determined. The results are shown in Table 4.

Table 4. **Terrain Redundancy Performance in POST2**

Test Case	Runtime	Iterations	% of Run
1. Origin	1614 sec	276e6	70.64 %
2. Best Fit Intercept	595 sec	97.4e6	37.40 %
3. Last Intercept	346 sec	28.6e6	7.92 %

The results show a small increase in runtime, but for the fastest case, using the last intercept as the starting point, the number of search iterations increased nearly 66% from 17.2 million to 28.6 million. This shows that many models in the POST2 simulation make redundant calls while executing. Including the feature that retains the answer from the last call is beneficial in terms of reducing the number of iterations the algorithm must execute.

5. VALIDATION

The presented algorithm and datasets were utilized in conjunction with the Mars Science Laboratory EDL flight data to verify correct operation and validation of the methodology. The radar slant range measurements were taken from the flight software telemetry, provided by Brian Schratz at JPL. The Gale crater terrain datasets were provided by Devin Kipp at JPL, Randy Kirk at USGS and Fred Calef from USGS. The terrain datasets are produced from orbiter stereographic images of the landing site with no map-tie rectification provided from ground assets at the MSL landing site. For slant ranges that exceed the Gale crater terrain dataset, the MOLA dataset is used. The reconstructed position and orientation was provided by Fred Serrichio at JPL. The reconstructed position gives the Descent Stage IMU location in the J2000 coordinate frame. The position and orientation were first rotated to the IAU Mars relative frame. The position then had to be modified by adding the respective TDS beam locations relative to the IMU location. The beam locations and look directions were the pre-launch estimates with no calibration performed. The modified beam positions and look directions were then evaluated using the ray-tracing algorithm to determine the slant range to the terrain. The reconstructed slant range is shown in Figure 9, where the solid lines represent the ray-trace slant range solution while the circles represent the TDS range measurements of the individual 6 beams. The data is reported after heatshield separation which is where the TDS system becomes active.

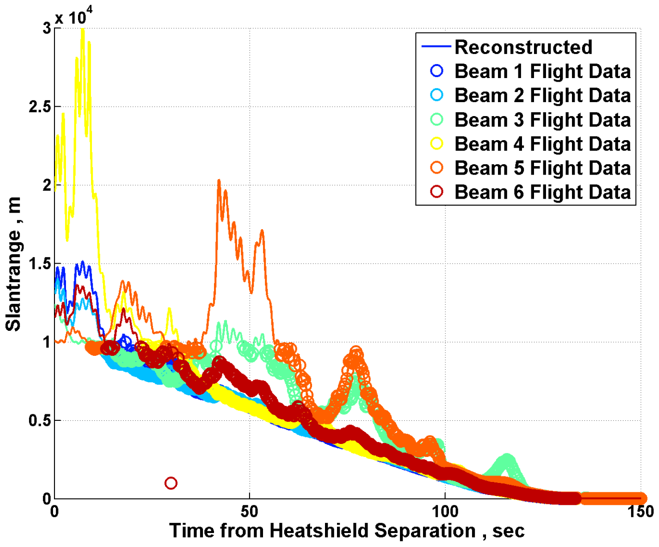

Figure 9. MSL TDS Range Measurements

The flight data shows that the radar returned valid data up to approximately 10 km slant ranges. The limit of 10 km slant ranges was expected from the physics based radar models that MSL used to determine performance, but can not be predicted by the purely geometric ray-tracing model. The rogue data point at approximately 30 seconds after heatshield separation is the TDS locking up on the heatshield. The trends are well captured by the ray-trace algorithm as indicated by the solid lines following the flight data circles well. The error in the ray-trace slant range measurements from the TDS slant range measurements are shown in Figure 10.

The errors are shown to jump at approximately 20 seconds after heatshield separation which is where the NAV filter incorporates the TDS measurements and updates the state. From that point forward, the errors are roughly zero mean with larger values seen during regions of high dynamic motion (parachute wrist mode, powered descent). The errors eventually converge towards zero near touchdown which occurred approximately 150 seconds after heatshield separation.

The good agreement in flight data verses the ray-trace slant range estimates helps to validate the terrain algorithms used in the POST2 simulation.

6. DATA PRODUCTS

Alternate uses of the terrain model have been shown to be useful in real world situations. One such application used on the Mars Science Laboratory operations was the generation of horizon masks. Horizon masks define line of sight viewing capability from a given surface location. Once the Curiosity

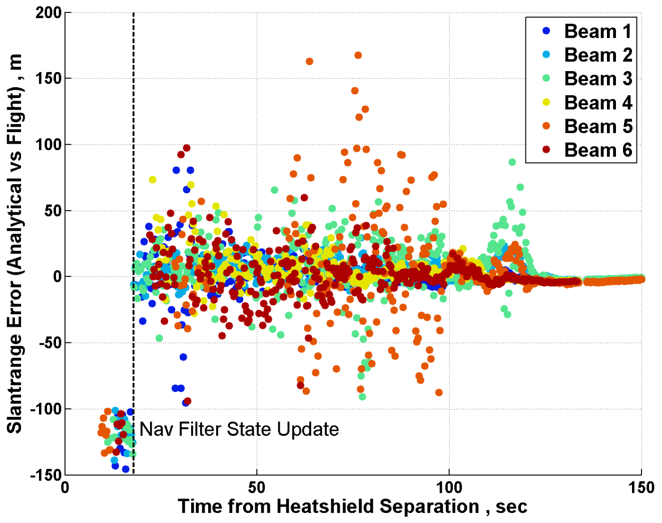

Figure 10. Error in Range Measurement Estimates

rover was safely on the ground, the horizon is not flat due to terrain features such as Mount Sharpe in Gale Crater. Orbiter relay communications would potentially be blocked depending on the occultation of the orbiter by terrain. For mission planning purposes, the rover operations team would require horizon masks to properly plan for communication relays. Such horizon masks are easily computed using the ray-tracing algorithm and datasets presented.

To do so, first the location of the rover UHF antenna needs to be known in the planet-relative frame. A sweep is then made in azimuth about the rover antenna. Along each azimuth, the ray-tracing algorithm, with the origin as the starting point, search for terrain along the pointing vector created by a descending elevation until a terrain intersect is found. The result is a dataset of azimuth verses elevation that can be utilized by rover operators in planning relay communications between orbiters and the rover. The paths of Mars Odyssey and the Mars Reconnaissance Orbiter are shown in Figure 11 to highlight the impact of the horizon mask on orbiter occultation.

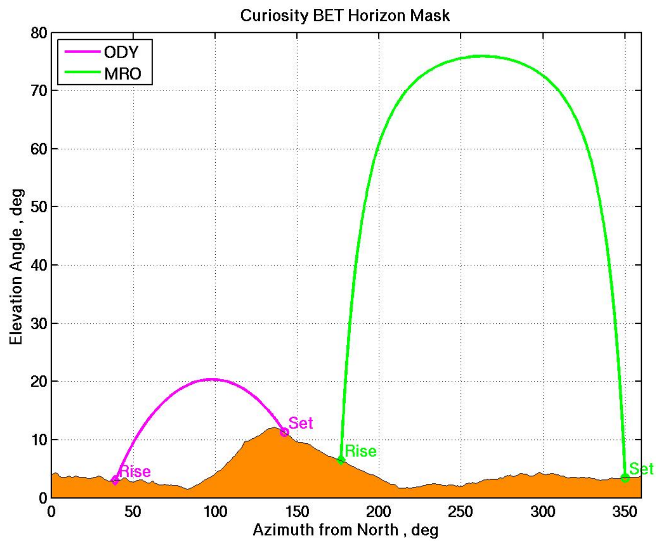

Figure 11. Curiosity Landing Location Horizon Mask

Another use identified during MSL operations is the determination of landing location by using the signal-lost time from the orbiter. In this case, the Mars Odyssey orbiter was relaying MSL's real time data transmissions during EDL on August 5, 2012 using a bent-pipe transmission. The orbiter lost the signal as it set behind Mount Sharpe at 05:37:41 UTC. As seen in the horizon mask image, the planet-relative location corresponds to a specific time that the estimated trajectory of Mars Odyssey would set behind Mount Sharpe. The inverse problem can then be analyzed where the vehicle location is not known, but the orbiter planet-relative location is known.

The landing location determination is done by sweeping through the expected longitude and latitudes of the landing footprint, and determining what time the orbiter would set behind the terrain of interest. By knowing the actual loss of signal time, the data can then be interpolated relative to the various computed set times to determine a line of possible locations the vehicle could be at loss of signal. An example of this is shown in Figure 12 where the MSL onboard estimates are shown in detail as described in Davis, et al [10], in addition to the cyan line where the vehicle is estimated to be due to the loss of signal time from Mars Odyssey.

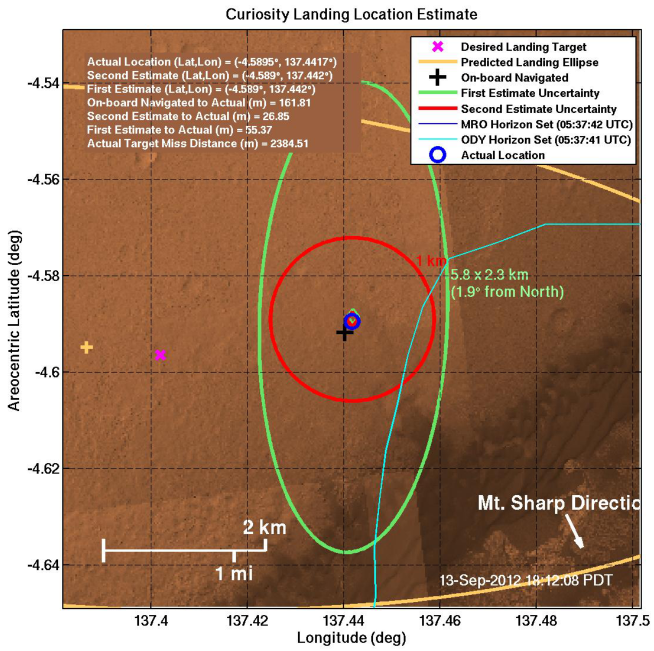
Curiosity Landing Location Estimate
Actual Location (Lat,Lon) = (-4.5895°, 137.4417°)
Second Estimate (Lat,Lon) = (-4.589°, 137.442°)
First Estimate (Lat,Lon) = (-4.589°, 137.442°)
On-board Navigated to Actual (m) = 161.81
Second Estimate to Actual (m) = 26.85
First Estimate to Actual (m) = 55.37
Actual Target Miss Distance (m) = 2384.51
Desired Landing Target
Predicted Landing Ellipse
On-board Navigated
First Estimate Uncertainty
Second Estimate Uncertainty
MRO Horizon Set (05:37:42 UTC)
ODY Horizon Set (05:37:41 UTC)
Actual Location
1 km
5.8 x 2.3 km
(1.9° from North)
Mt. Sharp Directic
2 km
1 mi
13-Sep-2012 18:12:08 PDT
Areocentric Latitude (deg)
Longitude (deg)

Figure 12. Curiosity Landing Location Estimate Using Orbiter Set Time

The results show that the estimate is within 1 kilometer of the actual landing location. This method could be useful in cases where onboard state estimated telemetry is lost from the vehicle, but contact is maintained up until signal loss. Using this method would help orbital assets to know where to look to locate the ground asset.

7. CONCLUSION

A simplified ray-tracing algorithm has been presented with focus on determining range to terrain intercepts. The method has been shown to operate efficiently in the POST2 simulation, only consuming 7% of the total runtime for the Mars Science Laboratory EDL simulation. The model used terrain datasets from the MSL Gale Crater landing site and MSL EDL flight data to verify its operation and interpretation of the datasets in computing slant range estimates for a radar system. Various uses of the model were presented for determining horizon masks and vehicle location due to orbiter set times. The model is currently being used on the Mars InSight mission to support terrain modeling. Future use can include any missions that require terrain interaction modeling on such planetary bodies as Earth, Venus and the moon.

ACKNOWLEDGMENTS

The author would like to thank NASA and the Mars Program Office for funding this project.

REFERENCES

[1] R.W. Powell, S.A. Striepe, P.N. Desai, E.M. Queen, G.L. Brauer, D.E. Cornick, D.W. Olson, F.M. Petersen, R. Stevenson, M.C. Engel, S.M. Marsh. ”Program to Optimize Simulated Trajectories: Volume II, Utilization Manual”. Lockheed Martin Corporation, Version 1.1.6.G, May 2004.

[2] Venus Magellan: Magellan Spherical Harmonics, Topography, and Gravity Data. http://pds-geosciences.wustl.edu/missions/magellan/shadr_topo_grav/index.htm PDS Geosciences Node. October 5 , 2007. Web.

[3] Mars Global Surveyor: MOLA MEGDRs. http://pds-geosciences.wustl.edu/missions/mgs/megdr.html PDS Geosciences Node. April 19 , 2007. Web.

[4] Shuttle Radar Topography Mission. http://srtm.usgs.gov/ United States Geological Survey. October 25 , 2007. Web.

[5] Clementine Gravity and Topography Data. http://pds-geosciences.wustl.edu/missions/clementine/gravtopo.html PDS Geosciences Node. October 5 , 2007. Web.

[6] Sunday, Dan. http://geomalgorithms.com/a06-_intersect-2.html ”Intersections of Rays and Triangles (3D).” Geometry Algorithms. June 08 , 2007. Web.

[7] S.E. Urban and P. Kenneth Seidelmann (eds), ”Explanatory Supplement to the Astronomical Almanac: Third Edition” Mill Valley [CA]: University Science Books, 2013.

[8] NetCDF. http://www.unidata.ucar.edu Network Common Data Form (NetCDF), 2009. Web.

[9] R.L. Kirk, E. Howington-Kraus, D. Galuszka, B. Redding, J. Antonsen, K. Coker, E. Foster, M. Hopkins, A. Licht, A. Fennema, F. Calef III, S. Nuti, T.J. Parker, and M.P. Golombek. ”Near-complete 1-m topographic models of the MSL candidate landing sites: Site safety and quality evaluation.” EPSC-DPS Joint Meeting 2011.

[10] J. L. Davis, J. D. Shidner, and D.W.Way, ”Mars Science Laboratory Post-Landing Location Estimation Using POST2 Trajectory Simulation,” AAS 13-313, AAS/AIAA 23rd Space Flight Mechanics Meeting, Kauai, HI, Feb. 2013.

BIOGRAPHY

Jeremy Shidner *received his B.S. in aerospace engineering from Embry-Riddle Aeronautical University in 2004 and M.S. degree in aerospace engineering in 2006 from the University of Maryland. He is currently the director of Flight Dynamics and Control at Analytical Mechanics Associates and is a member of the Atmospheric Flight and Entry Systems Branch at NASA Langley. His current activities include work on the NASA Space Launch System, Commercial Crew Program and POST2 development.*