# High-Fidelity Flash Lidar Model Development

Glenn D. Hines[a], Diego F. Pierrottet[b], Farzin Amzajerdian[a]

[a]NASA Langley Research Center, Hampton, VA, USA 23681; [b]Coherent Applications Inc., 20 Research Drive, Suite 500, Hampton, VA USA 23666

## ABSTRACT

NASA's Autonomous Landing and Hazard Avoidance Technologies (ALHAT) project is currently developing the critical technologies to safely and precisely navigate and land crew, cargo and robotic spacecraft vehicles on and around planetary bodies. One key element of this project is a high-fidelity Flash Lidar sensor that can generate three-dimensional (3-D) images of the planetary surface. These images are processed with hazard detection and avoidance and hazard relative navigation algorithms, and then are subsequently used by the Guidance, Navigation and Control subsystem to generate an optimal navigation solution. A complex, high-fidelity model of the Flash Lidar was developed in order to evaluate the performance of the sensor and its interaction with the interfacing ALHAT components on vehicles with different configurations and under different flight trajectories. The model contains a parameterized, general approach to Flash Lidar detection and reflects physical attributes such as range and electronic noise sources, and laser pulse temporal and spatial profiles. It also provides the realistic interaction of the laser pulse with terrain features that include varying albedo, boulders, craters slopes and shadows. This paper gives a description of the Flash Lidar model and presents results from the Lidar operating under different scenarios.

**Keywords:** Flash Lidar, high-fidelity, model, simulation

## 1. INTRODUCTION

Landing spacecraft vehicles on planetary bodies is a difficult task. Adding to this the prerequisite requirements of safety, which is a core requirement of all NASA missions, and precision, makes the task much more complex. NASA's ALHAT project[1] was created to help develop the technologies needed for future missions to accomplish this task seamlessly. ALHAT is currently funded under NASA's Advanced Exploration Systems (AES) program within the Human Exploration And Operations (HEO) Mission Directorate. It is a multi-center effort that includes: Johnson Spaceflight Center (JSC), which is responsible for project management, and guidance and navigation algorithm development; Jet Propulsion Lab (JPL), which is developing the system's hazard detection and hazard relative navigation algorithms; Langley Research Center (LaRC) which is developing the system's sensors and executes the simulations; and multiple contractors including FastMetrix, and Coherent Application Inc. (CAI) for laser system development.

A system like ALHAT is needed for future missions because of the inherent dangers and complexity of safely and precisely landing a vehicle on another planet. NASA has been fortunate in the past. Many of the Apollo missions were relatively close to having a mishap. Apollo 14 landed with a tilt angle of 11 degrees when its specified maximum was 12 degrees as shown in Figure 1. Apollo 15 landed on the rim of a small crater which tilted the vehicle to ~10 degrees and possible damaged the engine bell. The ALHAT system will help to significantly reduce the probability of these types of incidences. Additionally, future spaceflight missions, such as a mission to the interior of Shackleton Crater on the Moon, may have to land in an area of complete darkness. For missions to build or occupy a habitat or to explore a cave on Mars will require landing with a significantly improved precision from on the order of kilometers to meters. ALHAT will provide the technologies to enable these types of missions.



**Figure 1. Both Apollo 14 and 15 landed with the Lunar Module tilted.**

# 2. ALHAT ARCHITECTURE AND SIMULATION ENVIRONMENT

## 2.1 ALHAT Functional Architecture

A baseline functional ALHAT architecture[2] is shown in Figure 2. The core of the ALHAT system is a set of guidance, navigation and control (GN&C) algorithms, which are programmed with a planned trajectory for the vehicle. These algorithms receive inputs from multiple sensors including conventional guidance sensors, such as star trackers, accelerometers and inertial measurement units; a laser altimeter for altitude information, a navigation Doppler Lidar[3] for speed, altitude and partial attitude; and terrain and hazard detection sensors whose output is processed with terrain relative and hazard detection/relative navigation algorithms[4,5]. The output of the GN&C algorithms guides the thrusters and rocket motors to control the flight of the vehicle. The sensors interact with the terrain of the planetary body to produce their respective measurements.
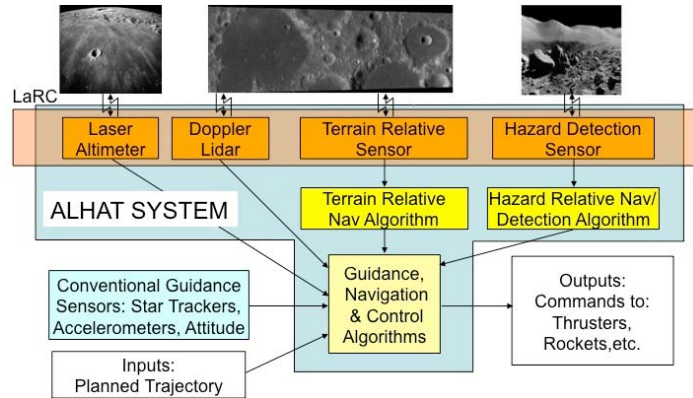


**Figure 2: Basic functional block diagram of the ALHAT architecture.**

## 2.2 POST2

Because of the broad design pallet and the complexity of component interaction, early on the project decided to perform a detailed simulation of the system and its operational environment. The project primarily used a LaRC developed tool called Program to Optimize Simulated Trajectories II (POST2)[6] to perform a complete end-to-end descent and landing trajectory simulation. POST2 provides the capability to include models of the vehicle mass properties, propulsion systems, guidance, navigation and control algorithms, planet including gravity and surface/terrains, and sensors as depicted in Figure 3. Various vehicle entry conditions and approaches were simulated over thousands of runs to study and determine suitable designs, integration methods of the components, and system performance[6]. Engineering analyses and system trades on trajectories, vehicles, and sensors were performed using the data generated within POST2.
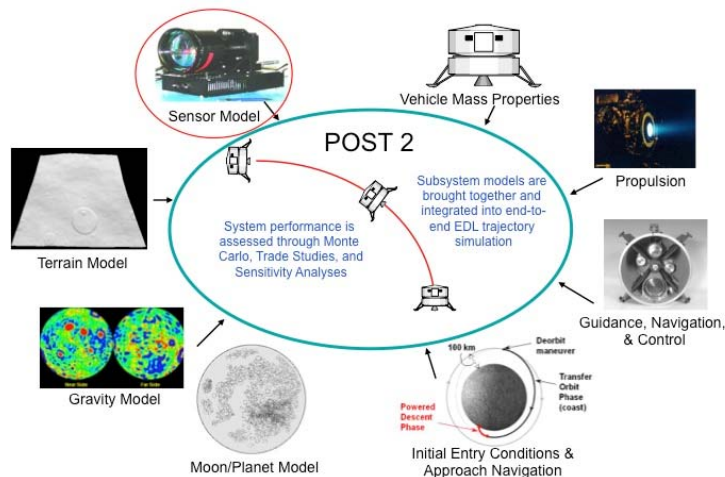


**Figure 3: Program to Optimize Simulated Trajectories II (POST2) end-to-end simulation engine.**

# 3. FLASH LIDAR

## 3.1 Description

The ALHAT team produced high-fidelity models for all of the functional components of the architecture. One of the primary models is of the hazard detection and avoidance (HDA)[5] and hazard relative navigation (HRN) sensor. For ALHAT this is a Flash Lidar sensor[7]. A Flash Lidar generates a range image based on the transmit and return time-of-flight of a laser pulse reflected from a surface. This range image is interpreted as a 3-D image of the surface, thus hazards such as rocks, craters and slopes can be imaged. The range image is generated simultaneously for all pixels on the focal plane array of the sensor so the images are not subject to motion distortion from a moving platform. Because of the high frame rate and multiple images can be easily obtained during descent in a timely manner. ALHAT is currently using a 128 x 128 array that generates images at 30 Hz.

## 3.2 Sample image

An example of a Flash Lidar range image is shown in Figure 4[8]. The 2-D view shows a sensor test range target board with various slopes, rectangular boxes of different size and depth, and areas of different reflectivity. The board is located at approximately 250 m from where the sensor is normally mounted for testing. A range image generated by the Flash Lidar is located in the foreground of the image. A depth chart is located on the right side of the Figure. The slopes and most of the rectangular targets can be resolved in the image. The objects in the background are trees that are located behind and above the test board.
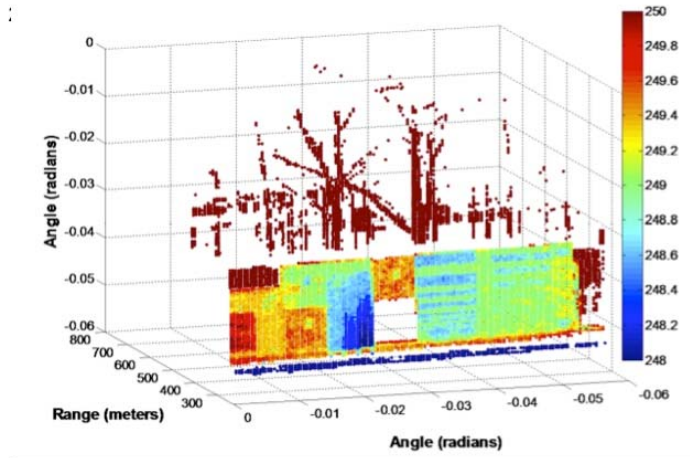


**Figure 4: Flash Lidar image of sensor test range target board.**

# 4. FLASH LIDAR MODELING

## 4.1 Physics

Our high-fidelity model is based on the physics of how the sensor works. The imaging equation for the sensor, $s(x, y, t)$, is described by the following equation:

$$s(x, y, t) = p(x, y, t) \otimes o(x, y, t) + n(t)$$

where $p(x, y, t)$, point spread function, is obtained from the lidar temporal pulse profile; $o(x, y, t)$, the object function, is obtained from the surface/terrain map and $n(t)$ is the sum of various random processes. A map is made up of a number of finite elements and a pixel is composed of the number of finite elements that fill a single detector's instantaneous field of view (IFOV). The object function describes how the transmitter energy is scattered from a map in amplitude and time.
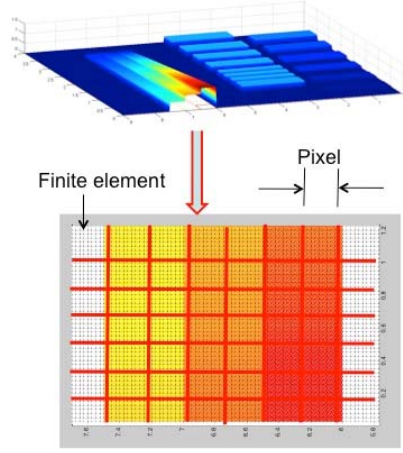
**Figure 5: Mapping of finite elements on a map to pixels within a focal plane array.**

The mapping of the finite elements used to compose a map to pixels within a focal plane array is depicted in Figure 5. To construct the IFOV for each pixel we use the following two equations for the point spread function and object function respectively:

$$p(x,y,t) = \frac{2E_T}{\tau\pi(R\Theta_{FW})^2} \exp\left[-\frac{(x-x_o)^2 + (y-y_o)^2}{\left(\frac{R\Theta_{FW}}{2}\right)^2}\right] \exp\left[-\left(\frac{(t_o - t_N)^2}{\tau^2}\right)\right]$$

and

$$o(x,y,t \rightarrow t) = \rho(x,y)\delta\left(t - \frac{2z}{c}\right).$$

The signal per finite element is the convolution of the point spread function and the object function, plus any noise:

$$s(x,y,t) = p(x,y,t)\otimes o(x,y,t) + n(t).$$

Solving this discretely, the signal per pixel is then the integration of the finite element data:

$$s_{IFOV}(x,y,t) = \rho(x,y)\sum_{i=1}^{n}\sum_{j=1}^{m} p(x_{i,j}, y_{i,j}, t - \frac{2z_{i,j}}{c}) + n(t)$$

There are various noise sources that should be accounted for in the calculation of the signal. The background thermal radiation photon flux is described by:

$$M(\lambda) = \frac{\pi\left(\frac{D_r}{2}\right)^2}{(IFOV)^2} \int_{\lambda_b}^{\lambda_a} 2\pi c/\lambda^4 \left[e^{h\frac{c}{\lambda kT}} - 1\right] d\lambda.$$

The electronic noise, total noise current, is described by the root-sum-square of detector dark current noise, Johnson noise and shot noise:

$$i_N = \sqrt{i_d^2 + i_j^2 + i_{Sh}^2}.$$

Then using radiometric link analysis we can determine a signal photon flux at the receiver focal plane array image plane:

$$F_S = \frac{\frac{\lambda}{hv}E_t(x,y)}{IFOV * R}(A_R/R^2)\beta K_o e^{-2\int \alpha(z)dz} \, ,$$

so the total current is described by the sum of the signal, background flux and noise:

$$i_T = q\eta_Q F_S + q\eta_Q M(\lambda) + i_N$$

thus giving a signal in noise. This current is converted to a voltage for processing. Different sensors will have different mechanisms for calculating a range. Our current version of our model first passes the signal through a differentiating circuit. A threshold value triggers a peak detection algorithm for which the peak is detected at the zero crossing of the differentiated Gaussian pulse in noise, as shown in Figure 6. Range is then calculated based on the time of the zero crossing mark. This is performed for each pixel and a 3-D image is reconstructed.
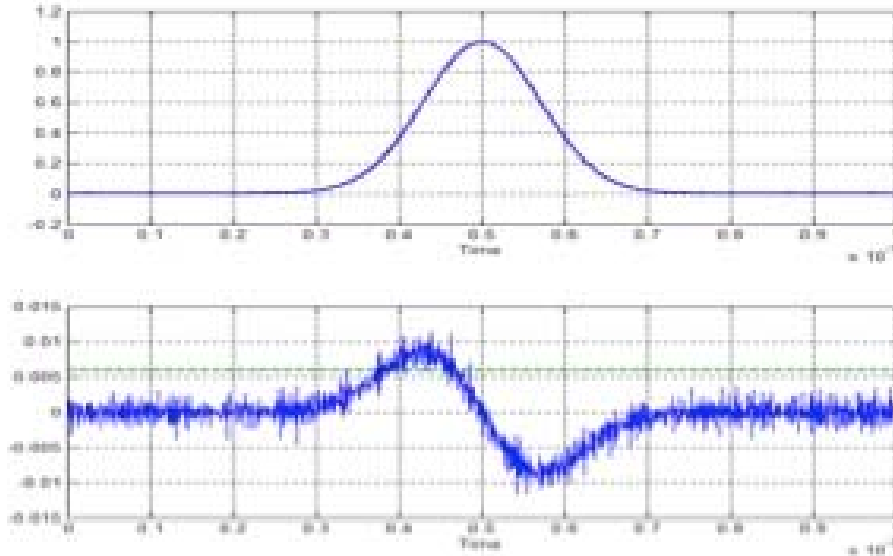


**Figure 6: Top: Gaussian pulse in noise. Bottom: differentiated pulse. The horizontal line is the threshold set point.**

## 4.2  POST2 compatible model

This model was initially developed within the Matlab environment. To be fully compatible with POST2, the code was converted from Matlab into C++. Figure 7 shows a functional block diagram of how the Flash Lidar model fits within the POST2 context. The model is first configured with a set of sensor input parameters, see Table 1 for a sample, and standard constants. These parameters enable the model to be configured either at compile time or run-time, thus allowing adjustments to be made during multiple, long Monte Carlo simulation runs. Looking back at Figure 7, the model must interact with a terrain and provide images (range samples) to the HDA/HRN algorithms. The output of these algorithms is fed to the GN&C algorithms for flight adjustments, and to data visualization tools so that the output can be viewed and analyzed. Spacecraft (S/C) trajectory data is fed back into the model so that it can determine pointing angles for the sensor to obtain the next set of samples from the terrain.

Table 1. Sample parameters for the Flash Lidar model.

| Parameter | Value | Description |
|---|---|---|
| Fpa_rows | 128 | Focal plane array rows |
| Fpa_cols | 128 | Focal |
| IFOV | 4.0e-4 | IFOV of the receiver |
| R_efficiency | 0.8 | Receiver optical efficiency |
| Dr | 0.12 | Receiver aperture diameter |
| Fno | 2.4 | Receiver optical f number |
| Lambda | 1.5e-6 | Laser wavelength |
| E | 5.0e-2 | Transmitter laser energy per pulse |
| BeamDiv | 0.0512 | Transmitter beam divergence |
| M | 40 | Detector Gain |
| QE | 0.7 | Quantum efficiency |
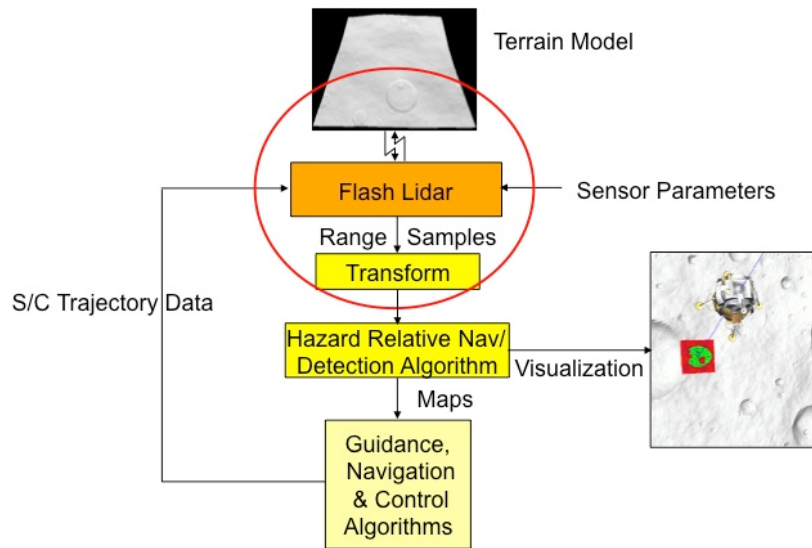
**Figure 7: Block diagram of the C++ version of the model in the POST2 environment**

Moving into the Flash Lidar model box is the layer of functionality shown in Figure 8. The S/C trajectory is fed into the routine ResamplePoses that calculates the position and orientation, or pose, of the sensor along the S/C trajectory. The S/C trajectory is considered to be truth and it is in a lunar surface lunar fixed coordinate frame (all simulations were for a lunar mission). The routine RayDTMIntersction performs the projection function for the sensor by determining the patch of map that corresponds to where the field of view (FOV) and IFOVs of the sensor falls. The transmitter beam is modeled as a bundle of rays. The routine ComputePolyFiniteElements returns the finite elements intercepted by the rays on the map via interpolation. Truth ranges, surface albedo and incidence angles are returned and sent to the routine MeasureRanges. The MeasureRanges routine performs the imaging equation as discussed earlier for the each pixel within the array. The calculated ranges, which are now in the sensor frame, are then sent to subsequent HDA/HRN algorithms for processing.
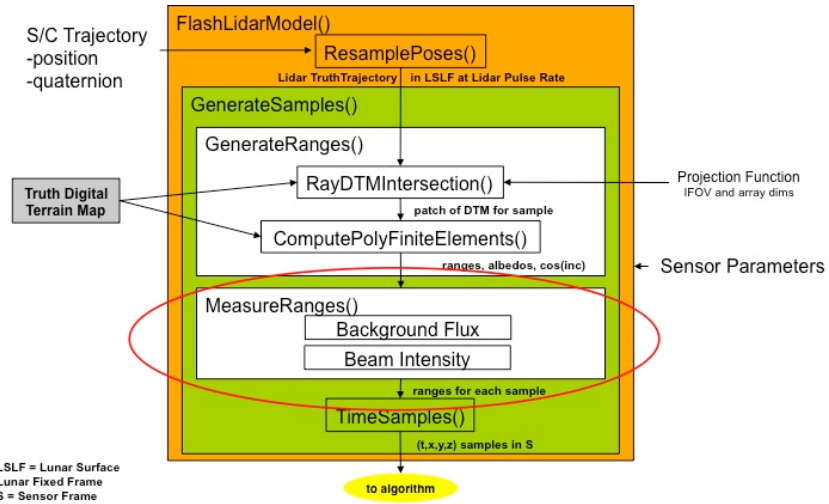
**Figure 8: Block diagram of the C routines that implement the model of the Flash Lidar and its interaction with a map.**

The following figures are examples of the processing generated by the model. Figure 9 is a model map of the sensor test board discussed earlier. This "terrain" was fed into the model starting with the parameter settings of a pulse width of 4 nsec, IFOV of 0.2 mrad and a precision of 5 cm. The output of the model is shown in the upper left corner of Figure 10. Then, only the IFOV was adjusted to 0.4 mrad, 0.8 mrad and 1.0 mrad. The resulting output images are shown in the upper right, lower left and lower right corners, respectively, in Figure 10. It is easy to see the rapid degradation of the image as we make this parameter change. At an IFOV of 1.0 mrad only nearly none of the rectangular targets that are decreasing in depth can be resolved, only the larger rectangular objects at the top of the board can be resolved, and the differences in the change in slopes is hard to discern.
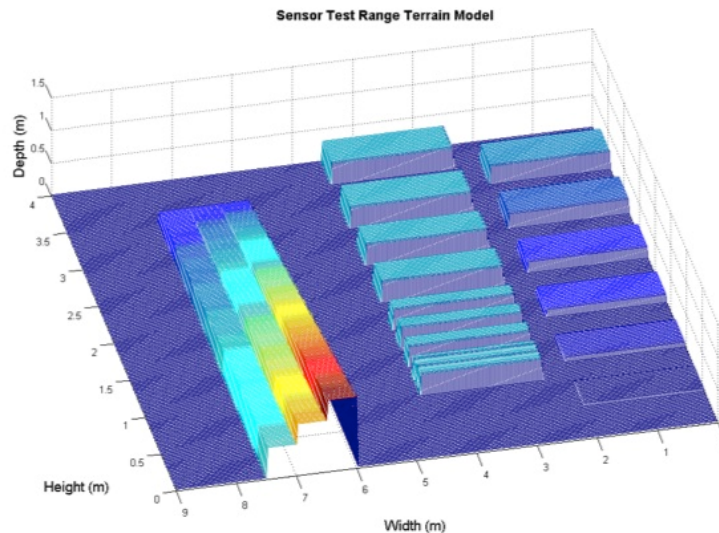


**Figure 9: Model of the sensor test range used to test the Flash Lidar model.**
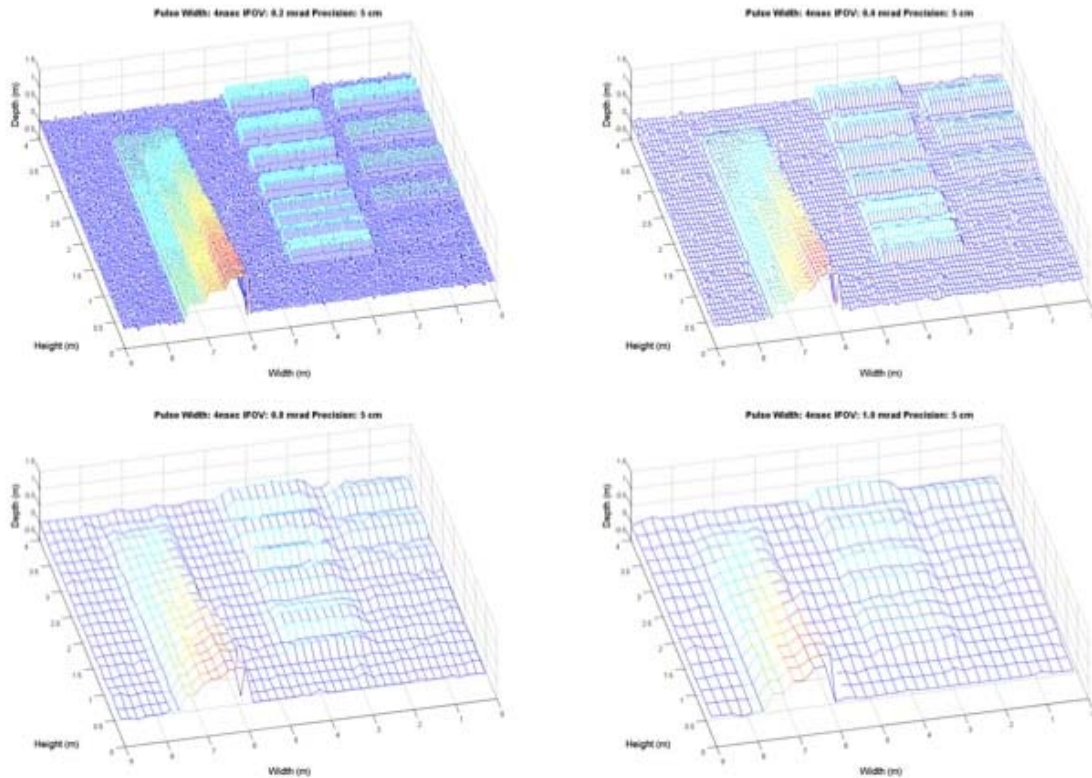
**Figure 10: Output model data with increasing IFOV from 0.2, 0.4, 0.8 and 1.0 starting from the upper right.**

An example of the processing of a synthetic terrain is shown in Figure 11. On the left is the input (truth) scene from synthetic terrain data for the Flash Lidar model pointed nadir towards the surface from 1 km. On the right is the processed output data of the model. Hazards in the upper left corner can be seen very easily while those on the bottom right are not resolved. Performing this type of analysis under different configurations and surface properties gives excellent insight on the operational boundaries of the system.
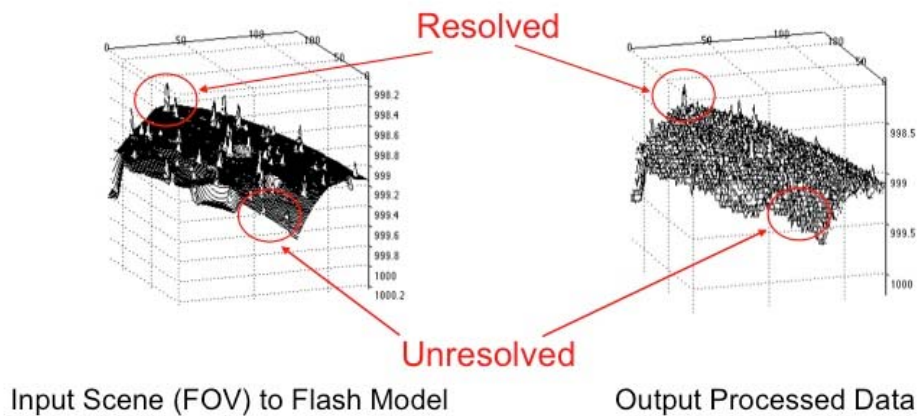


**Figure 11: Model processing of synthetic terrain data.**

# 5. CONCLUSIONS

In conclusion, we have developed a high fidelity Flash Lidar model that has been used by the ALHAT project to evaluate its operational performance under various configurations and under different flight trajectories. This provides a significant improvement over the standard low fidelity, statistics-based noise and bias model used by many missions for simulation and analysis. The ALHAT project has already benefitted from the use of the model. It has helped with sensor acquisition and development, determining trajectory profiles for flight tests, HDA/HRN algorithm development and with creating actual hazard fields for test vehicles to land within.

# 6. ACKNOWLEDGEMENTS

## REFERENCES

[1] Epp. C. Robinson, E., and Brady, T., "Autonomous landing and hazard avoidance technology (ALHAT)," IEEE Aerospace Conference, 1-7, (2008).

[2] T. Brady, S. Paschall, D. Zimpfer, C. Epp, and E. Robertson, "Hazard detection methods for lunar landing," Proc. IEEE Aerospace Conference (2009).

[3] Farzin Amzajerdian, Diego Pierrottet, Larry Petway, Glenn Hines, and Bruce Barnes, "Doppler lidar sensor for precision navigation in GPS-deprived environment," Proc. of SPIE ,Vol. 8731, SPIE Defense Security, and Sensing Conference (2013).

[4] A. Johnson and J. Montgomery, "An overview of terrain relative navigation for precise lunar landing," Proc. IEEE Aerospace Conference (2008).

[5] A.E. Johnson, A. Huertas, R.A. Werner, and J.F. Montgomery, "Analysis of on-board hazard detection and avoidance for safe lunar landing," Proc. IEEE Aerospace Conference (2008).

[6] Davis, J., Stripe S., Maddox R., Hines, G., Paschall S., Cohanim, B., Fill, T., Johnson, M., Bishop, R., MeMars, K., Sostaric, R., and Johnson, A., "Advances in POST2 End-to-End Descent and Landing Simulation for the ALHAT Project," AIAA/AAS Astrodynamics Specialist Conference, AIAA-2008-6938, (2008).

[7] Amzajerdian, F., Vanek, M. D., Petway, L. B., Pierrottet, D. F., Busch,G. E., Bulyshev, A., "Utilization of 3-D Imaging Flash Lidar Technology for Autonomous Safe Landing on Planetary Bodies," SPIE Proceeding Vol. 7608, paper no 80 (2010).

[8] Pierrottet, D., Amzajerdian, F., Meadows, B., Estes, R., and Noe, A., "Characterization of 3-D imaging lidar for hazard avoidance and autonomous landing on the moon," Proc. SPIE 6650, (2007).

Filename:              lf99-17657_final-high_fidelity_flash_lidar_model_development
Directory:             C:\Users\wbaize\AppData\Local\Temp\1
Template:

                       C:\Users\wbaize\AppData\Roaming\Microsoft\Templates\Normal.
   dotm
Title:                 Sample manuscript showing specifications and style
Subject:
Author:                mattn
Keywords:
Comments:
Creation Date:         5/23/2014 5:19:00 PM
Change Number:         2
Last Saved On:         5/23/2014 5:19:00 PM
Last Saved By:         Amzajerdian, Farzin (LARC-D208)
Total Editing Time:    0 Minutes
Last Printed On:       6/24/2014 8:09:00 AM
As of Last Complete Printing
   Number of Pages: 9
   Number of Words:        2,717 (approx.)
   Number of Characters:   15,489 (approx.)