Simulations of the Hazard Detection System for Approach Trajectories of the Morpheus Lunar Lander

Michael E. Luna[1], Andres Huertas[2], Nikolas Trawny[3], Carlos Y. Villalpando[4], Keith E. Martin[5], William Wilson[6]
Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, 91109, USA

Carolina I. Restrepo[7]
NASA Johnson Space Center, Houston, TX, 77058, USA

The Hazard Detection System is part of a suite of sensors and algorithms designed to autonomously land a vehicle on unknown terrain while avoiding any hazards. This paper describes the simulations built to predict the performance of the Hazard Detection System to support flight testing onboard the Morpheus Lander testbed at the Kennedy Space Center. The paper describes a hardware-in-the-loop simulation that was used to predict system performance under nominal operating conditions, and also a Monte Carlo simulation to predict command timing performance bounds under a wide range of varying conditions.

I. Introduction

The Autonomous Landing and Hazard Avoidance Technology (ALHAT) Project has developed a suite of sensors and algorithms that can sense terrain data, analyze the topography of the terrain surrounding an intended landing site, select safe landing locations, and communicate with its host vehicle to navigate relative to terrain features [1]. One part of the ALHAT system is the Hazard Detection System (HDS), developed by the Jet Propulsion Laboratory (JPL). The HDS consists of a 2-axis yoke-style gimbal with a LIDAR (light detection and ranging) sensor payload, a Compute Element (CE), a power distribution unit (PDU) and battery box, and an LN-200 inertial measurement unit (IMU) [2].

The HDS operates under strict timing constraints that were determined based on the trajectory of its host vehicle, the Johnson Space Center-built Morpheus free flight lander. In a typical free flight, the Morpheus vehicle lifts-off from a ground point adjacent to the Kennedy Space Center's Shuttle Landing Facility (SLF) runway and flies toward a hazard field constructed for flight tests. Prior to lift-off, the HDS is powered on and initialized on the ground. After lift-off, at a desired altitude, the HDS prepares the system internally for the hazard detection algorithms and commands the LIDAR to start lasing. Once the host vehicle reaches a certain range from its intended landing site, located on the hazard field, it sends HDS an OPERATE command. At this point, the *Mosaic Planner* flight software module takes in the latest navigation estimates from the vehicle and plans a mosaic, or scanning pattern, of a square area of the ground around the intended landing site. This mosaic plan consists of a list of points on the ground where the HDS LIDAR must be pointed to, which is then sent to the *Gimbal Manager* flight software module. *Gimbal Manager* applies all necessary coordinate transformations to calculate gimbal angle commands to correctly point the LIDAR boresight as the vehicle continues to fly towards its intended landing site. While the gimbal/LIDAR are scanning the planned square area on the ground, the *Annotator* flight software module annotates each LIDAR range image with position and attitude information, so that they can subsequently be assembled into a Digital Elevation Model (DEM) of the terrain. Once HDS has a complete DEM, the Hazard Detection and Avoidance (HDA) algorithm [3] selects the 5 safest landing sites along with a prominent feature in the DEM for subsequent tracking. HDS then sends the coordinates of the safest landing site to the host vehicle and points the gimbal/LIDAR at a selected feature during the Hazard Relative Navigation (HRN) phase. During HRN, the gimbal actively tracks the HRN feature so that it is kept within the LIDAR view as the vehicle approaches its landing site.

This paper describes how each of these phases was simulated in order to predict the HDS hardware and flight software performance both before a given flight test and to enable post-flight data analysis. There are two separate

[1] Staff Engineer, Guidance & Control Section, Mail Stop 198-235, AIAA Senior Member.
[2] Senior Engineer, Mobility & Robotics Systems Section, Mail Stop 198-235.
[3] Senior Engineer, GN&C Hardware & Testbed Development, Mail Stop 198-235, AIAA Member.
[4] Senior Engineer, Advanced Computer Systems & Technologies, Mail Stop 198-138
[5] Senior Engineer, Simulation & Support Equipment
[6] Staff Engineer, GN&C Hardware & Testbed Development, Mail Stop 198-235, AIAA Member
[7] Aerospace Engineer, Integrated GN&C Analysis Branch, AIAA Member.

simulations: a hardware-in-the-loop simulation that operates with actual flight software and a lab-based setup of flight hardware, and a separate Monte Carlo simulation. The "hardware-in-the-loop" simulation establishes the behavior of HDS software and hardware under nominal operating conditions. The Monte Carlo simulation predicts the timing performance of the mosaicking and HDA algorithm. The following sections include detailed descriptions of each of these simulations as well as simulation prediction results compared with actual flight test data.

II. Hardware-in-the-loop Simulation

The hardware-in-the-loop simulation has several components that include actual flight software [2], flight hardware, and separate pieces of C and MATLAB code. Its main purpose is to predict as accurately as possible the HDS hardware and flight software behavior and safe landing performance for an upcoming flight test given a simulated trajectory provided by the Morpheus vehicle team.

The simulation has the components shown in Figure 1. The vehicle's position and attitude are taken from the simulated trajectory at the time of the OPERATE command. Using this information, the *Mosaic Planner* flight software calculates the necessary points on the ground to point the gimbal to. A separate piece of C code is then used to compile the list of coordinates and gimbal angles for each of these points into a single message that is sent to the *Gimbal Manager* flight software as it would be done in flight.

Once the Mosaic Plan message has been assembled, the Gimbal Manager message along with trajectory data can be run through the lab setup for a full simulation. The lab setup consists of the compute element, power distribution unit, flight gimbal, a mass simulator of the flight LIDAR mounted onto the gimbal, and a small LIDAR on the side to interface with the *Lidar Manager* flight software. The lab gimbal goes through the mosaic scanning motions as if it were flying the simulated trajectory. At the same time, the *Lidar Manager* and *Annotator* flight software modules are running in order to stamp the LIDAR images with information about the sensor-to-map position and sensor pointing at the time of each flash.

However, since the lab LIDAR does not flash the real terrain but instead a target board in the lab, a separate piece of C code that generates synthetic LIDAR images is used. These synthetic images contain the surveyed terrain data from the hazard field at KSC. The images are then run through the hazard detection flight software, which assembles them into a DEM similar to what would be seen during a flight at KSC. The DEM is then processed for hazard detection and safe site selection, and an HRN feature is selected.

Once the coordinates of the selected HRN are determined, another piece of C code is used to generate the appropriate *Gimbal Manager* message necessary to point the gimbal to the HRN feature during the remainder of the flight until the SHUTDOWN command.

Once all Gimbal Manager commands are calculated for a given simulated trajectory, rather than running the simulation up until HRN feature selection, pausing to construct the DEM, and resuming the simulation for the HRN phase, the simplest setup was to compile both the mosaic plan and HRN phase Gimbal Manager commands into a single file. This allowed the entire simulation to run seamlessly from OPERATE to the SHUTDOWN command.

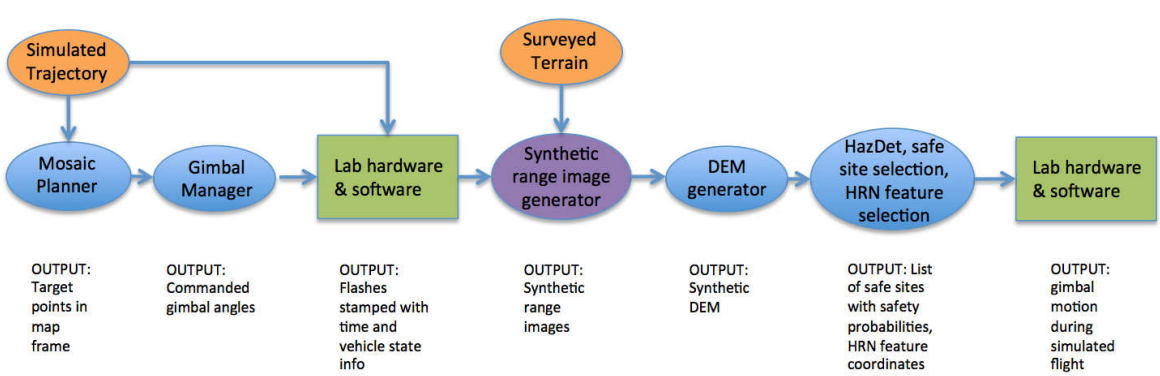


Figure 1. Hardware-in-the-loop simulation.

The following sections describe each component of the hardware-in-the-loop simulation, as depicted in Figure 1 above.

A. Simulated Trajectory

The simulated trajectory data was provided by the Morpheus Lander team as a time history of all flight data including commands that are sent from the vehicle to the HDS, such as PREPARE, OPERATE and SHUTDOWN. With this information, the vehicle's position and attitude data was used to produce accurate simulations in the JPL lab.

B. Mosaic Planner

The *Mosaic Planner* software module receives a command to plan a mosaic pattern for scanning a 60x60 meter area on the ground. At the time that the command is received, *Mosaic Planner* uses the latest estimated vehicle position and attitude as a starting point, and it calculates a Boustrophedon pattern (scanning from left to right and right to left in alternating rows) on the ground that guarantees 50% overlap between LIDAR flashes in the cross-track direction and 20% overlap in the downtrack direction. The output of *Mosaic Planner* is a list of target points on the ground in map frame[8]. This list gets sent directly to *Gimbal Manager*, so that it can command the gimbal to point sequentially to each of these points as the vehicle flies. The actual *Mosaic Planner* flight software is used here.

C. Gimbal Manager

The *Gimbal Manager* software is able to command the gimbal in three different ways: point relative to the vehicle, point to a set of coordinates on the ground, and point according to a set of azimuth and elevation angles. The mosaic points are a list of coordinates on the ground in map frame. *Gimbal Manager* takes the vehicle's current estimated position and attitude and it calculates the azimuth and elevation angles required to point the gimbal to each mosaic point. For the purposes of this simulation, a file was created that contained all *Gimbal Manager* commands throughout the simulated flight. This file was used in the lab to command the gimbal.

D. Lab Hardware and Software

The trajectory data and the file containing all gimbal commands are run through the lab setup. In the lab there is a flight gimbal with a mounted LIDAR mass simulator in order to appropriately simulate gimbal dynamics, and a small LIDAR flashing on the side in order to be able to stamp the flashes with the correct time and vehicle state information. These flashes are later replaced by synthetic LIDAR images, since the lab images reflect a target board in the lab rather than the terrain at KSC. Additionally, the compute element runs the following flight software modules: *Gimbal Manager, Data Manager, Annotator, Lidar Manager*, which then provide output files that are similar to what comes out of the real flight tests and can be analyzed in much the same way.

III. Terrain Generation, Surface Characterization, and Landing Safety Assessment

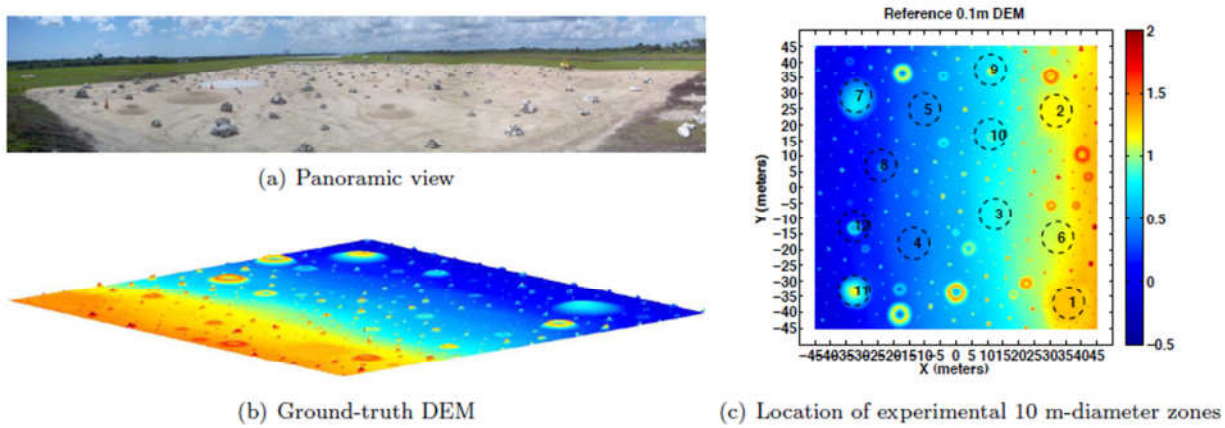
The ALHAT systems for automatic surface reconstruction and for hazard mapping and assessment have become mature. Before they can be put to practical use, however, it is essential to be able to characterize their performance for the purposes of scientific evaluation and their utility to engineers planning and designing landed missions. It is also important to be able to predict performance for a variety of scenarios. The evaluation metrics need to be simple enough to be readily comprehensible but still to capture the important relevant performance parameters [6, 7].

The goal of our simulations and evaluations is to show that the probability of landing failure is within acceptable limits. The probability estimates [3] derived and predicted by simulation (this paper) have been validated experimentally through the free flights test campaign described above. The performance predictions and results depend on the quality of the DEM representing the terrain. To accomplish that, our simulations use a high fidelity LIDAR sensor model developed for the ALHAT Project. A LIDAR is attractive because it is a direct ranging sensor applicable at relatively high altitudes [8, 9]. Planetary and small-body terrain can be constructed by fractal modeling of the underlying terrain to generate slopes that follow size-frequency models at different scales. The surface craters and rocks are added by generating populations based on size-frequency distribution models for those features. For our simulations, however, we rely on a DEM constructed from a high-density LIDAR survey of a one-hectare hazard field constructed at KSC. Figure 2 illustrates the hazard field (a) and the 0.1 meter resolution DEM (b) and (c) constructed from the 10-million point laser surface survey. This DEM constitutes the reference or ground truth terrain used for simulation predictions and subsequent flight test result evaluations. The ALHAT system makes no assumptions about the terrain. Zones 1 and 2 have been fitted with 10-m concrete landing pads, also unknown to the ALHAT systems. The pads, however, represent flat and leveled patches and thus the simulations should demonstrate

[8] Map frame is defined at the PREPARE command, where X-map is the cross-track direction, Y-map is the downtrack direction, and Z-map is up (altitude).

that when the pads are in the field of view, the safest location to land is on the visible pad. The crater and boulder populations were derived from the size frequency distributions of automatically detected and measured craters and rocks [10] from a patch of actual lunar terrain.

Next we discuss the LIDAR sensor model tool used to generate LIDAR flashes for a given trajectory, and the flight-code-based simulation tools for: DEM generation, terrain characterization, hazard assessment, safe site selection and hazard-relative navigation for guidance to the selected safe landing site.



(a) Panoramic view

(b) Ground-truth DEM

(c) Location of experimental 10 m-diameter zones

Figure 2. (a) Panoramic view of 1-hectare hazard field. (b) Corresponding 0.1m resolution ground-truth DEM. (c) Nadir view of the terrain DEM with 10m-diameter safe zones (1-5) and experimental zones (6-10).

A. Flash LIDAR model for Synthetic Range Image Generation

The ALHAT system on board Morpheus incorporates a flash LIDAR that generates range images at 20 Hz, based on the transmit and return time-of-flight of a laser pulse reflected from the surface of the hazard field. Each 128x128 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 multiple images can be easily obtained during descent within the real-time requirements of the system.

Prior to field testing, the ALHAT team at NASA Langley Research Center produced a high-fidelity model of the flash LIDAR for use in early ALHAT POST2 simulations [11, 12]. In our simulations we used a version of the model with enhancements produced by the JPL ALHAT team. The high-fidelity model is based on the physics of how the sensor works; it implements the imaging equation for the sensor (see [12] for details):

$$\boldsymbol{s(x,y,t) = p(x,y,t) \otimes o(x,y,t) + n(t)} \tag{1}$$

where *p(x,y,t)*, the point spread function, is obtained from the LIDAR temporal pulse profile; *o(x,y,t)* the object function is obtained from the hazard field surface 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. The model accounts for the various noise sources when calculating the signal, in particular, the background thermal radiation photon flux, and the electronic noise contributed by the dark current, the Johnson noise and the shot noise. Using radiometric link analysis the model determines a signal photon flux at the receiver focal plane array. The total current is described by the sum of the signal, background flux and noise thus giving the resultant signal. The model calculates range by passing 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. 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. Table 1 illustrates a few of the constants and parameters (of the 37 possible settable values) used in our simulations. Figure 3 illustrates a single flash LIDAR range image and the projection onto a frame DEM. The middle flash DEM covers about 13m x 7m of the terrain from about 400m away. The stretched frame is for a trajectory with 30 deg flight path angle. Note the missing data due to the shadow of the rock. The flash DEM resolution is 0.1m per pixel, compatible with the full DEM mosaic it contributes to.

Table 1. Sample parameters for the flash LIDAR model

Parameter	Value
Focal Plane Assembly rows (pixels)	128
Focal Plane Assembly columns (pixels)	128
Receiver IFOV (mrad)	1.364e-4
Receiver optical efficiency R_{eff} (%)	0.8
Receiver aperture diameter D_r (m)	0.125
Range precision (m)	0.12
Laser wavelength (m)	1.06e-6
Transmitter laser energy per pulse E (J)	7.0e-2
Transmitter beam divergence (rad)	0.0256
Detector Gain M (dB)	40
Quantum efficiency QE (%)	0.7

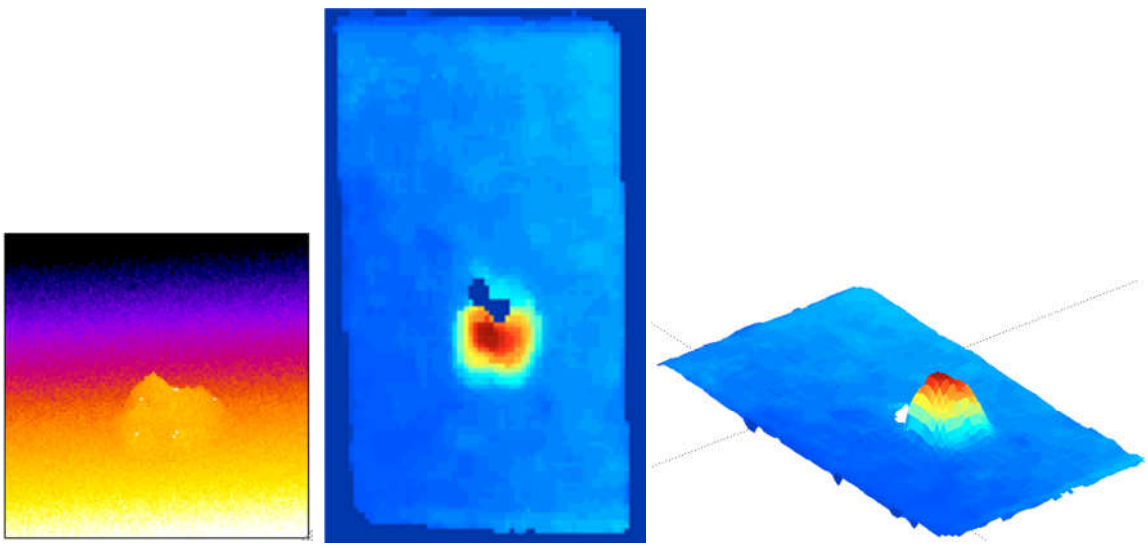


Figure 3. Left: 128x128 flash range image of a rock on the surface of the reference hazard field. White is closer. Middle: The projection of the point cloud derived from it is transformed into a flash DEM in map frame. Right: 3-D view of the rock from an arbitrary viewpoint.

B. DEM Generator

The DEM generator assembles all the relevant LIDAR range images into a single DEM suitable for terrain characterization, hazard detection, safe site selection and HRN. The incoming LIDAR frames are annotated by the *Annotator* flight software module with a coordinate system transform describing the relationship between the landing map and the sensor coordinate frames. The DEM generator converts the range image to a point cloud in map frame as illustrated above. To account for LIDAR and navigation noise, the DEM generator first aligns the point cloud to the DEM being constructed before accumulating it into the DEM to avoid discontinuous seams and/or smeared features [6].

In order to satisfy real-time requirements, i.e. the maximum allowable total duration of the OPERATE command up to safe sites reporting, the DEM mosaics are limited to an area of 60x60m that includes the concrete landing pad in safe zone 1. The sample simulation illustrations presented below are thus limited to that portion of the hazard field. Figure 4 shows the area scanned and the footprints of the range images needed to complete the 60 60m DEM mosaic. The simulated range images incorporate a range precision of 0.12m, a worst-case scenario. Note that the flash DEMs overlap by 50% cross-track and 20% downtrack.

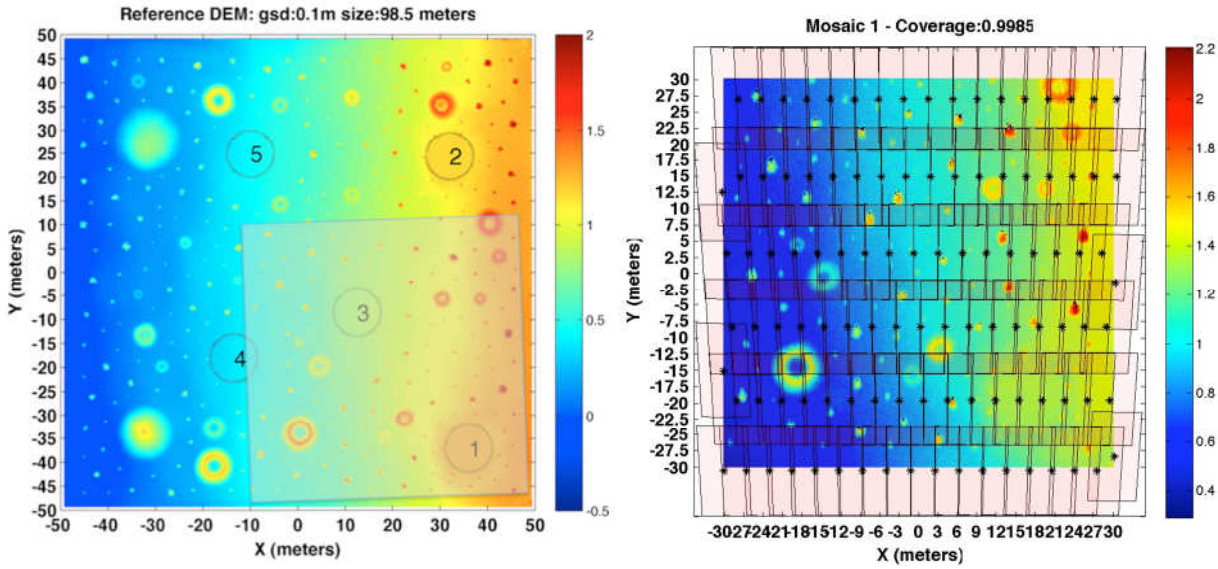


Figure 4. The 60m x 60m portion of the reference DEM scanned for the simulation and the footprints of the flash range images acquired according to the input trajectory and mosaic plan.

C. Hazard Detection, Safe Site Selection, and HRN Feature Selection

Once the DEM is complete, hazard detection and safe site analysis can commence. The HDS system uses a probabilistic analysis algorithm, integrating vehicle geometry and navigation uncertainty to evaluate the "safety probability" of a particular location in the DEM. Each pixel is evaluated for safety for a given vehicle size, configuration and mechanical tolerances, for a range of vehicle orientations as if the center of the vehicle was positioned at that spot. Regional maxima of safety probabilities are computed and safe sites are reported to the vehicle with an aim to report the probabilistically safest sites in different regions of the DEM. Figure 5 illustrates the simulated DEM mosaic and the final safety map with safe sites identified. In the left image note that the safest site is located on the concrete pad, and that the remaining safe sites are on safe locations. The safety map in the right image is produced by analyzing the terrain slopes and roughness using the full 0.1m resolution data but computed every meter for real-time application. The map is convolved with a 1-m 1-σ Gaussian filter to account for navigation uncertainty. Each pixel in the final safety map illustrated represents safety for the entire 4 m diameter lander.

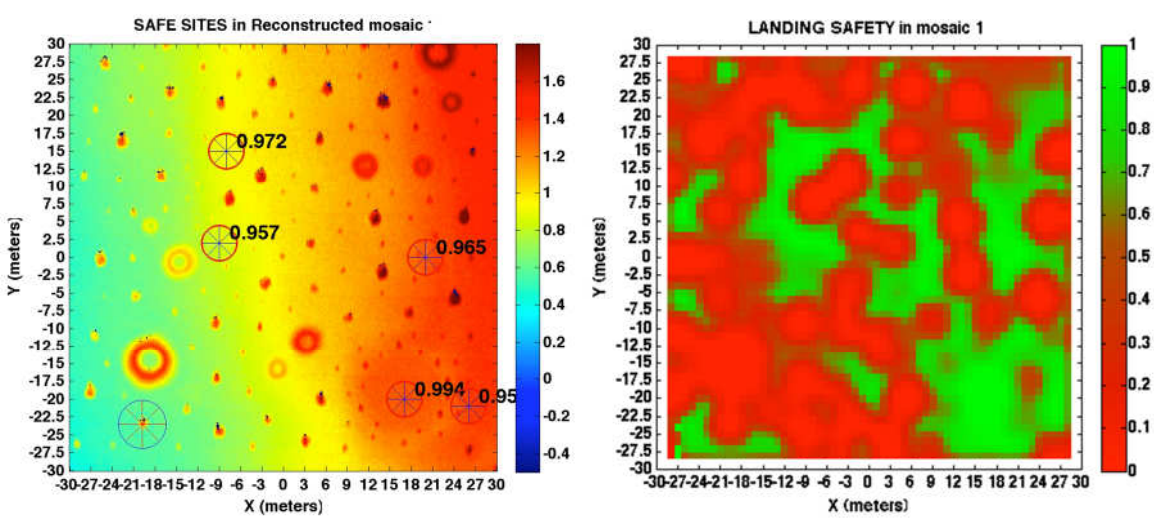


Figure 5. Left: The simulated 0.1m resolution DEM mosaic and identified top five safe landing sites represented by wheel symbols about the size of the Morpheus lander. Right: There are many safe locations to land safely as shown in the final 1-m resolution safety map.

Figure 6 below illustrates the 60x60m DEM mosaic constructed during free flight #11 of Morpheus at KSC. Compare to the prediction illustrated in Figure 5. The safest site is identified on the concrete landing pad on the bottom right. The remaining safe sites are all in safe locations. There are some differences between the ground truth illustrated earlier and the actual state of the hazard field. Also, environmental conditions, not modeled, affect the performance of the sensor and the quality of the actual flight range images.

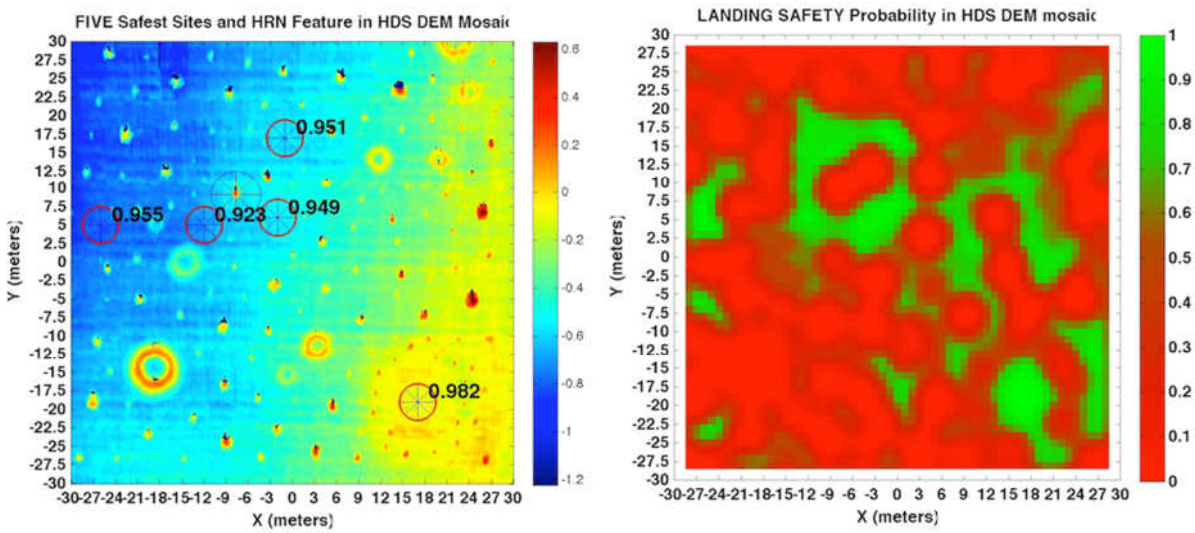


Figure 6. Left: Actual DEM mosaic and safety map constructed during Morpheus free flight #11. Right: Actual final safety map.

Once the Hazard Detection stage is complete, Hazard Relative Navigation (HRN) can commence. HRN aims to help limit the error growth of the vehicle navigation state. The HDS picks a "featureful" region of the DEM and commands the gimbal to point to that spot. The blue wheel symbol on the lower left of the DEM mosaic in Figure 5 corresponds to that spot. The HRN stage takes in a LIDAR flash annotated with the current vehicle navigation state estimate and searches for where that flash should be in the DEM constructed earlier. If the flash is found, the HRN process reports to the vehicle the measured offset of the feature found.

IV. Monte Carlo Timing Simulation

A. Software Architecture

In addition to simulating the gimbal motion with HDS flight software and lab hardware—which required simulating range image generation, DEM generation, and hazard detection—the timing performance of the OPERATE command was also simulated. The HDS is designed to function autonomously via commands sent by the host vehicle. There are two key commands in this process: PREPARE, triggered by altitude, and OPERATE, which is triggered shortly after the start of the host vehicle's initial descent guidance profile.

The PREPARE command accomplishes the following: it computes a transformation from the Earth-Surface-Earth-Fixed (ESEF) frame to the HDS internal map frame, enables the *Annotator* flight software module so that it can tag incoming LIDAR range images with the vehicle navigation state, initiates LIDAR flashing, and lastly, pre-points the LIDAR (mounted to the gimbal) to the lower left corner of the map for the "reconnoiter" stage that comes later on.

At the desired time the host vehicle commands OPERATE, which consists of the following major component activities:

1) Reconnoiter: If the host vehicle is within the desired operational range, the HDS takes a single incoming range image, obtained while the lidar is pre-pointed at the reconnoiter point, and computes the average elevation of the terrain. This elevation bias is then used as the Z-coordinate in map frame of each planned mosaic point computed by the *Mosaic Planner*.

2) Mosaic Planning: In this step, the *Mosaic Planner* software module takes the DEM center coordinates, DEM extents, desired flash overlap fractions, navigation pose (vehicle position and attitude), and other parameters to compute a mosaic plan.
3) Mosaic Execution: the gimbal physically points to the planned mosaic points while compensating for vehicle dynamics
4) Digital Elevation Model generation: the incoming LIDAR range images are added to an accumulating DEM while the gimbal is still slewing
5) DEM finalization: Once the gimbal has completed the mosaic, and the last LIDAR range image has been accumulated, the final hazard detection DEM points are computed at the desired resolution and saved to disk
6) Hazard Detection: the hazard detection DEM is analyzed for safe landing regions to identify 5 distinct safe landing sites
7) HRN Feature Selection: the Hazard Detection DEM is analyzed for a high-contrast feature that the LIDAR can be pointed to
8) Hazard Relative Navigation (HRN) updates: the gimbal points the LIDAR boresight at the selected feature, tracking it while the incoming range images are analyzed at a rate of 1 Hz to find the navigation error to that feature. Each HRN update is then passed onto the host vehicle for ingestion into the ALHAT navigation filter.

B. Timing Prediction Methods

As part of the timeline constraints, the host vehicle allocated a specific duration from the start of OPERATE to the completion of reporting of safe sites to the host vehicle. The total OPERATE-to-safe sites duration was known to be a function of both vehicle navigation state (geometric parameters) and OPERATE component activity timing parameters. The definition of the various geometric parameters is depicted in Figure 7. Slant range R, flight path angle (FPA), and cross-track angle (CTA) all define the position of the HDS IMU in map frame. The roll angle (about the HDS IMU X_b-axis), yaw angle (about HDS IMU Y_b-axis), and pitch angle (about HDS IMU Z_b-axis) are standard aircraft-type roll, pitch, and yaw angles. These angles all define the HDS IMU attitude relative to the map frame.

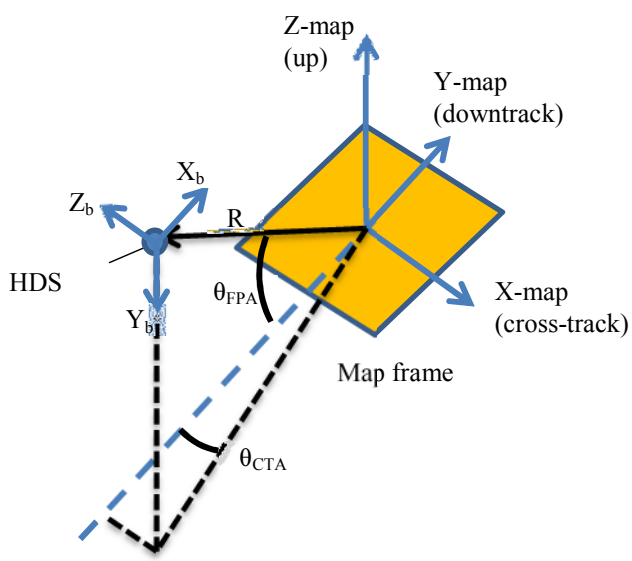


Figure 7. Vehicle (HDS IMU) positional and attitude geometry

To determine whether the OPERATE activities could be accomplished within the allocated duration, the timing sensitivity to geometry and expected variation in timing parameters was studied using 3 prediction methods:

1) A coarse "Approximate Partials" method in Excel. This method used timing parameters based on November 2012 helicopter flight data and predictions of *Mosaic Planner* planned points vs. geometric parameters (slant range, flight path angle, cross-track angle, roll angle) [4]. The uncertainty in mosaic

points was computed by taking partial derivatives of the number of mosaic points with respect to various geometric parameters using a central difference method.

2) A coarse "Curve Fit Partials" method, similar to the approximate method except that the partials were computed from MATLAB curve fits of the *Mosaic Planner* points variation with geometry.
3) A more robust Monte Carlo approach in which both geometric and timing parameter inputs were sampled randomly at 1000 points each according to a Gaussian distribution. The expected mean and 1-sigma uncertainty for each geometric parameter was based on host vehicle Monte Carlo trajectory dispersion simulation data. The timing parameter mean and 1-sigma uncertainty estimates were based partially on November 2012 helicopter test flight data [4] and hardware replays at JPL of past tether flights and nominal KSC simulated trajectories.

The key outputs of the timing prediction methods were an overall estimate of the OPERATE-to-safe sites duration, gimbal mosaic execution duration, DEM generation duration, duration to 1st HRN update, etc. The OPERATE timing performance was studied for the following conditions: several map sizes (ranging from 30x30m to 90x90m); mean slant ranges of 450m, 475m, and 500m; flight path angles of 30 deg (for the coarse methods); and mean flight path angles of 25, 30, and 35 deg (for the Monte Carlo method).

The results of the coarse methods were used to determine a suitable map size for flight tests. The Monte Carlo results were used initially to refine the coarse estimates under more realistic timing/geometry conditions, and then to determine the mean, minimum and maximum OPERATE-to-safe sites durations as final predictions prior to the free flights on the Morpheus vehicle.

1. Approximate Partials Timing Method

In the Approximate Partials method, the OPERATE activity timing for map sizes ranging from 30x30m to 90x90m with fixed geometry and attitude were investigated. To provide data for this method, the *Mosaic Planner* software was run numerous times given assumptions on map size, vehicle position (slant range, flight path angle, cross-track angle), and vehicle attitude (HDS IMU roll, pitch, and yaw angles). For a given map size, curves of the number of mosaic points versus a given geometric parameter were generated, while holding the other geometric parameters at nominal values.

In order to estimate the total OPERATE-to-safe sites duration, timing parameters for all the component activities were identified. An OPERATE timing parameter is either a direct estimate of the duration of component activity, or an interval (inverse rate) to determine the duration of a component activity. The timing parameters used in the Monte Carlo simulation method are listed below.

- OPERATE command startup duration
- Primary OPERATE slew to reconnoiter trackpoint duration (redundant since the slew is done by PREPARE)
- Range gate trigger duration
- Secondary OPERATE slew to reconnoiter trackpoint duration (redundant)
- Reconnoiter elevation bias computation duration
- Number of mosaic points per flash, $k_{pts_per_flash}$
- Mosaic planning interval (sec/mosaic point)
- DEM generation/accumulation interval (sec/flash)
- DEM finalization duration
- Hazard detection duration
- HRN Feature Selection duration
- Duration to 1st HRN update

The number of mosaic points per flash and mosaic planning interval timing parameters were estimated based on flight test data obtained in December 2012 with the HDS mounted underneath a helicopter at the Kennedy Space Center's Shuttle Landing Facility [4]. In this test campaign, there were 8 flights, several of which had multiple approaches to the hazard field in an attempt to test the OPERATE command performance. Estimates of the other timing parameters were based on JPL timing studies conducted in November 2013 using standalone Tilera processors.

Given estimates of all timing parameters above, the OPERATE timing performance was investigated for map sizes ranging from 30x30m to 90x90m, at slant ranges of 450m, 475m, and 500m and flight path angle of 30deg. For each map size investigated, the number of mosaic points was read from the plots of the number of mosaic points

versus slant range for the desired map size. The number of mosaic points, in turn, was used to estimate the number of flashes needed to generate the DEM and to estimate the mosaic planning duration. The durations for the other major OPERATE activities (DEM finalization, Hazard Detection, Feature Selection, and time to 1st HRN update) were taken from timing study results conducted at JPL in November 2013. The total OPERATE-to-safe sites reporting duration was then computed by summing the contributions from all component activities.

In order to estimate the uncertainty in the predicted OPERATE-to-safe sites duration, the uncertainty in each component activity was estimated. The only uncertainty estimates that required explicit calculation were mosaic planning and DEM generation, which involved estimating the uncertainties in the number of mosaic points, number of flashes, and DEM generation interval. The uncertainty in number of mosaic points required estimating the partial derivatives of the number of mosaic points with respect to several geometric parameters. The partial derivatives can be estimated from plots of the predicted number of mosaic points as a function of slant range, flight path angle, cross-track angle, and roll angle for each DEM size, at nominal values of the other parameters. The other duration uncertainties were taken from JPL timing study data.

Given formulas and test-data-derived averages and standard deviations for estimating the nominal OPERATE component activity durations and their uncertainties, the total duration of OPERATE-to-safe sites and its uncertainty was determined. Table 2 below shows the predicted number of mosaic points, predicted number of flashes, predicted nominal OPERATE-to-safe sites duration and uncertainty, and corresponding minimum and maximum OPERATE-to-safe sites durations for several map sizes and slant ranges.[9] At the time of completion of the Approximate Partials method, the allocated OPERATE-to-safe sites duration was set at 13.5 sec. This table clearly shows that the 75x75m and 90x90m map cases exceeded the allocated OPERATE-to-safe sites duration (denoted by the red cells). As a result, the Approximate Partials method revealed the largest feasible map size to be 60x60m.

Table 2. OPERATE-to-safe sites reporting Timing Predictions for Approximate Partials Method

Case	Map Size (m)	Slant Range (m)	Flight Path Angle (deg)	# Mosaic Points	# Flashes	OPERATE dur (sec)	OPERATE unc (sec)	OPERATE min dur (sec)	OPERATE max dur (sec)
1	30x30	450	30	120	27.0	2.484	0.487	1.997	2.971
2	30x30	475	30	115	26.0	2.413	0.406	2.007	2.819
3	30x30	500	30	105	23.0	2.203	0.375	1.828	2.578
4	45x45	450	30	271	60.0	6.152	0.796	5.355	6.948
5	45x45	475	30	257	56.0	5.828	0.838	4.990	6.667
6	45x45	500	30	250	55.0	5.746	0.583	5.163	6.329
7	60x60	450	30	500	109.0	11.220	1.661	9.560	12.881
8	60x60	475	30	430	94.0	9.930	1.226	8.703	11.156
9	60x60	500	30	375	82.0	8.898	1.129	7.769	10.027
10	75x75	450	30	730	159.0	18.381	2.005	16.377	20.386
11	75x75	475	30	688	150.0	17.501	2.141	15.360	19.642
12	75x75	500	30	564	123.0	14.861	1.515	13.346	16.377
13	90x90	450	30	1021	223.0	27.201	2.842	24.359	30.043
14	90x90	475	30	959	209.0	25.742	2.302	23.439	28.044
15	90x90	500	30	904	197.0	24.490	2.479	22.011	26.968

2. Curve Fit Partials Timing Method

The method for estimating the number of mosaic points, number of flashes, and other related timing parameters in the Curve Fit Partials method was the same as in the Approximate Partials method, except that the partial derivatives of the number of mosaic points with respect to geometry was computed from curve fits in MATLAB of the plots of the predicted number of mosaic points versus the various geometric parameters. The curve fits were obtained in order to provide a more accurate estimate of the partial derivatives, and hence the uncertainties in number of mosaic points.

The predictions of the OPERATE-to-safe sites reporting duration using the Curve Fit Partials method is shown in Table 3 below. The only difference between the Curve Fit Partials method and the Approximate Fit Partials method is in the uncertainties of the number of mosaic points. Thus, all timing estimates in Table 3 are the same as those in Table 2, except for the uncertainty in OPERATE duration, and hence the minimum and maximum OPERATE durations. These results confirm that maps above 60x60m cannot be constructed within the allocated time of 13.5 sec, as indicated by the red cells. The overall differences between the Curve Fit Partials method and

[9] In this table, the cross-track angle and HDS IMU roll, pitch, yaw angles were all assumed to be 0 deg.

Approximate Partials method are minor (less than 1 sec) for all cases. This indicates that the Approximate Partials method did not have significant error in the uncertainty estimation.

Table 3. OPERATE-to-safe sites reporting Timing Predictions for Curve Fit Partials Method

Case	Map Size (m)	Slant Range (m)	Flight Path Angle (deg)	# Mosaic Points	# Flashes	OPERATE dur (sec)	OPERATE unc (sec)	OPERATE min dur (sec)	OPERATE max dur (sec)	Uncertainty Diff (Curve Fit – Approx.) (sec)
1	30x30	450	30	120	27.0	2.484	0.420	2.064	2.904	-0.068
2	30x30	475	30	115	26.0	2.413	0.401	2.012	2.814	-0.005
3	30x30	500	30	105	23.0	2.203	0.374	1.829	2.577	-0.002
4	45x45	450	30	271	60.0	6.152	0.835	5.317	6.987	0.038
5	45x45	475	30	257	56.0	5.828	0.663	5.166	6.491	-0.176
6	45x45	500	30	250	55.0	5.746	0.623	5.123	6.369	0.040
7	60x60	450	30	500	109.0	11.220	1.425	9.796	12.645	-0.236
8	60x60	475	30	430	94.0	9.930	1.221	8.708	11.151	-0.005
9	60x60	500	30	375	82.0	8.898	1.201	7.696	10.099	0.073
10	75x75	450	30	730	159.0	18.381	2.143	16.238	20.525	0.139
11	75x75	475	30	688	150.0	17.501	1.698	15.803	19.200	-0.443
12	75x75	500	30	564	123.0	14.861	1.544	13.317	16.406	0.029
13	90x90	450	30	1021	223.0	27.201	2.346	24.856	29.547	-0.496
14	90x90	475	30	959	209.0	25.742	2.355	23.387	28.096	0.052
15	90x90	500	30	904	197.0	24.490	1.848	22.642	26.338	-0.631

3. Monte Carlo Timing Method

Although the Approximate and Curve Fit timing methods had similar results in terms of uncertainty, the timing uncertainty calculations were based on educated guesses of the uncertainty in slant range, flight path angle, cross-track angle and map roll angle. To obtain a more accurate estimate of the total OPERATE-to-safe sites duration under expected variation in vehicle position and attitude, as well as variation in timing parameters, a Monte Carlo timing simulation was conducted.

The important feature of the Monte Carlo timing method is that the key geometric and timing parameters identified in the coarse timing methods are treated as random variables with normal (Gaussian) distributions, each with specified mean and 1-σ (standard deviation) values. Several different cases were simulated, each with 1000 samples. The first step in building the Monte Carlo simulation tool was to identify the specific parameters to be sampled and decide whether their mean and 1-σ depend on map size or not. The key geometric and timing parameters relevant to the Monte Carlo simulation are listed in Table 4 below. All of the geometric parameters in this table are obviously independent of map size, whereas most of the timing parameters are a function of desired map size. Note the inclusion of the velocity magnitude as an additional "geometric" parameter, which was not considered in the coarse timing methods. The main reason velocity must be included is that the *Mosaic Planner* software has a mode where it can take the instantaneous velocity of the vehicle at the time of mosaic planning and propagate it forward to predict the changing vehicle position throughout the mosaicking process. Without velocity propagation in the mosaic planning, the software would assume the vehicle is farther away than it actually is, potentially leading to additional mosaic rows and hence increased time.

Table 4. Key Gaussian Parameters for Monte Carlo Timing Simulation

Parameter	Comment
Geometric Parameters	
Slant Range	3 values (450, 475, 500m) mean per DEM size
Flight Path Angle	3 values (25, 30, 35deg) mean for some cases, otherwise ~30deg mean
Cross-track Angle	Mean/1-sigma constant for all DEM sizes
Roll Angle (about HDS x-axis)	Mean/1-sigma constant for all DEM sizes
Pitch Angle (about HDS y-axis)	Mean/1-sigma constant for all DEM sizes
Yaw Angle (about HDS z-axis)	Mean/1-sigma constant for all DEM sizes

Velocity Magnitude	Mean/1-sigma constant for all DEM sizes
Timing Parameters	
OPERATE startup duration	Constant 2 ms for all map sizes
Primary slew to reconnoiter trackpoint duration	Constant 0 sec assumed since the PREPARE command pre-points the gimbal/LIDAR
Range check duration	Constant 1 ms for all map sizes
Reconnoiter duration	Constant 0.058 sec for all map sizes
k_{ppf} (points per flash)	Mean/1-sigma constant for all DEM sizes (capped at 100 Hz/20 Hz = 5 points per flash)
Mosaic Planning Interval (sec/mos pt)	Mean/1-sigma constant for all DEM sizes
DEM Accum Interval (sec/flash)	Mean/1-sigma a function of DEM size
DEM Finalization duration	Mean/1-sigma a function of DEM size
Hazard detection duration	Mean/1-sigma a function of DEM size
Feature Selection duration	Mean/1-sigma a function of DEM size
1st HRN update duration	Mean/1-sigma constant for all DEM sizes

An initial set of Monte Carlo timing simulations was planned to verify the correctness of the results from the coarse timing methods. In this initial set of Monte Carlo runs, the mean and 1-σ values for the timing parameters were based on flight software timing studies on a Tilera processor conducted at JPL and December 2012 KSC helicopter flight test data. The range of map sizes to study was expanded to include 50x50m and 55x55m. It was expected that sampling all Gaussian geometric and timing parameters 1000 times each per case would be a time-consuming process. To facilitate the Monte Carlo timing simulations, Excel was used to store the mean and 1-σ values for each Gaussian parameter and the constants for non-Gaussian parameters to use for each case. Excel's built-in Visual Basic for Applications (VBA) scripting language was used to write the large table of values into a single comma-separated value (CSV) text file that can be read into MATLAB.

A MATLAB Monte Carlo timing simulation tool was written to automatically conduct the timing simulation for each case defined in the CSV input file. The process for performing the timing simulation is depicted in Figure 8. First the CSV input file is read into MATLAB to capture all the mean, 1-σ, and constant values to use for each simulation case. Then each case is processed one at a time. For a particular case, the Gaussian parameters identified in Table 4 are randomly sampled 1000 times each and the data stored in an array. Next, each Gaussian parameter is assigned a value from the array storing the randomly generated samples. The sampled geometric parameters (slant range, flight path angle, cross-track angle, roll angle, pitch angle, and yaw angle) are then all collected into a single 'pose_input' vector. This pose vector is passed to another function that defines the input text file that is ultimately passed to the *Mosaic Planner* software. This function constructs the HDS IMU body-to-map frame transform (mXb), which consists of a position (or offset) vector (the HDS IMU position relative to the map frame origin, expressed in map frame) and a rotation matrix defining the attitude of the HDS IMU. The position vector is constructed from the slant range, flight path angle, and cross-track angle. The rotation matrix is constructed by converting the sampled HDS IMU roll, pitch, and yaw angles into a quaternion and then converting the quaternion into a rotation matrix. The velocity vector is defined assuming there is no cross-track velocity component. The vehicle acceleration and angular velocity were assumed to be zero for simplicity.

The *Mosaic Planner* software call then returns a MATLAB-formatted text file containing the number of mosaic points planned for a particular case. Given the wide sampling of the geometric parameters, the *Mosaic Planner* may occasionally attempt to build a plan that is outside the allowed operating range. When a mosaic plan is successfully created, however, the MATLAB tool computes the predicted total OPERATE-to-safe sites duration and other interesting durations.

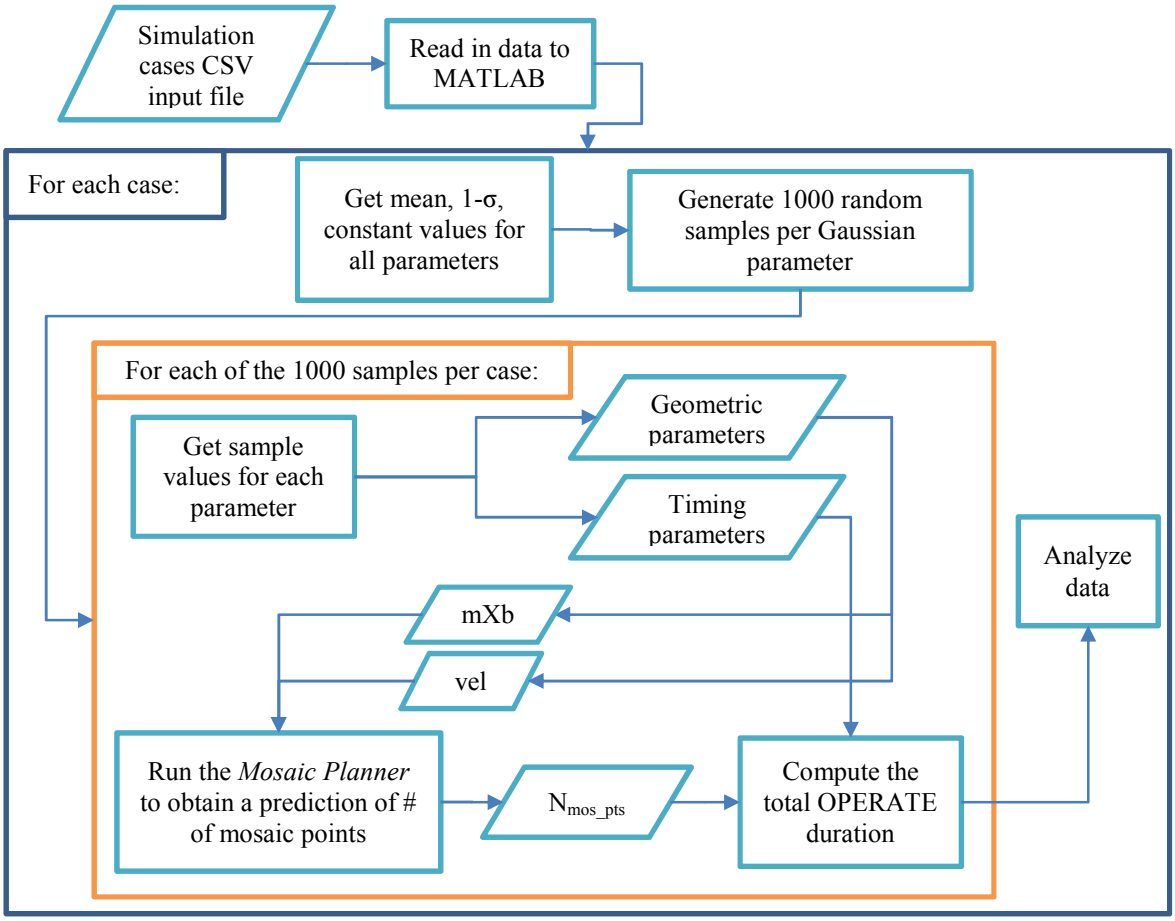


Figure 8. Data Flow Diagram for performing Monte Carlo timing simulations

Once a particular set of DEM/slant range/flight path angle cases was simulated (a process that required performing the simulations a group at a time), the resultant successful mosaic plan and timing data for each case could be analyzed statistically. An example histogram of the predicted total OPERATE-to-safe sites duration is shown in the following Figure 9. This histogram was generated prior to the first free flight (#10). At the time of the generation of this plot, the total allocated OPERATE-to-safe sites duration was stated to be 13.5. The total OPERATE duration histogram shows that the maximum predicted duration was 11.6 sec, providing a 1.9 sec margin.

The prediction results of the first Monte Carlo timing simulation runs are shown in Table 5 below for 60x60m, 75x75m, and 90x90m maps. Maps of smaller size are not shown since it was already known that at least a 60x60m map was feasible. These first run results indicated that, as expected, the 75x75m and 90x90m map cases were still not options, as the OPERATE durations exceeded the allocated time of 13.5 sec. Note that for the 90x90m map cases, a high percentage of the attempted mosaic plans failed due to exceeding the maximum of 1200 allowed points. In addition, these results showed that there were simulations where even the 60x60m map cases exceeded the allocated duration (e.g. at low slant range and high flight path angle).

Table 5. Initial Validation Run of Monte Carlo Timing Simulation Predictions

Case #	DEM size	Slant Range mean (m)	Flight Path Angle mean (deg)	# Failed Mosaic Plans	Mean # Mosaic Points	Min # Mosaic Points	Max # Mosaic Points	OPERATE mean dur (sec)	OPERATE min dur (sec)	OPERATE max dur (sec)
7	60	450	25	0	511	436	590	10.312	7.852	13.807
8	60	450	30	0	567	515	668	11.379	9.266	15.948
9	60	450	35	0	635	603	770	12.617	10.368	16.844

10	60	475	25	0	436	409	538	8.906	7.424	11.937
11	60	475	30	0	506	486	615	10.208	8.538	12.470
12	60	475	35	0	586	485	631	11.716	9.073	14.458
13	60	500	25	0	401	386	444	8.231	6.907	10.600
14	60	500	30	0	472	388	521	9.588	7.188	11.554
15	60	500	35	0	509	452	591	10.236	8.183	13.496
16	75	450	30	0	954.091	796	1170	21.449	15.873	28.277
17	75	475	30	0	831.48	733	951	18.839	15.098	24.927
18	75	500	30	0	724.598	620	908	16.572	13.987	22.567
19	90	450	25	923	1177.221	1020	1200	28.28	23.62	33.067
20	90	450	30	1000	0	0	0	0	0	0
21	90	475	30	983	1129.941	1071	1200	27.477	23.541	34.09
22	90	500	30	160	1114.682	980	1200	26.889	21.574	34.965

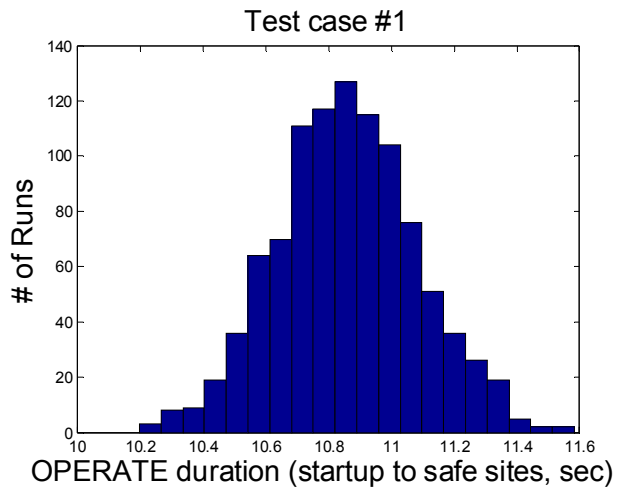


Figure 9. Distribution of total OPERATE duration for a 60x60m map at mean and 1-σ values for the 04/01/2014 Morpheus Monte Carlo trajectory simulation dispersions (prior to free flight #10)

Note that the mean and 1-σ values for the geometric parameters in the first set of Monte Carlo timing simulations were still only educated guesses. Given this, subsequent simulation runs were performed with the geometric parameter mean and 1-σ values based on separate JSC-provided Monte Carlo simulations of the Morpheus trajectory delivered to JPL. These Morpheus trajectory simulations provided dispersions in the position and attitude of the vehicle, from which the mean and 1-σ values of the geometric parameters could be extracted.

The second set of refined Monte Carlo timing simulations focused on 50x50m, 55x55m and 60x60m map sizes using the dispersions predicted from the Morpheus Monte Carlo trajectory simulations. The second set of runs was conducted prior the first free flight with HDS onboard the Morpheus vehicle (free flight #10) and the third set was completed prior to the second free flight (#11). The two separate sets of runs were conducted because the time of issuance of the OPERATE command by the Morpheus Autonomous Flight Manager (AFM) was changed after free flight #10. The results of the second set of Monte Carlo timing simulation runs are shown in Table 6 below.

The Monte Carlo timing predictions presented in Table 6 showed that none of the cases exceeded the allocated duration of 13.5 sec for free flight #10. In the worst case, for a 60x60m map, the maximum OPERATE duration was predicted to be 12.721 sec, providing a small margin of 0.779 sec. This gave the JPL team confidence that specifying a 60x60m map when flying on a nominal Morpheus trajectory and within expected position, velocity, and attitude dispersions would produce a total OPERATE duration within the allocated maximum.

Table 6. Predictions for Second Set of Monte Carlo Timing Simulations (for FF #10)

Case #	DEM size	Slant Range mean (m)	Flight Path Angle mean (deg)	# Failed Mosaic Plans	Mean # Mosaic Points	Min # Mosaic Points	Max # Mosaic Points	OPERATE mean dur (sec)	OPERATE min dur (sec)	OPERATE max dur (sec)
1	60	462.762	30.960	0	530	527	538	11.995	11.333	12.721
2	55	462.762	30.960	0	482	479	487	10.789	10.059	11.424
3	50	462.762	30.960	0	364	361	366	7.734	7.171	8.228

Discussions on the performance of free flight #10, however, lead to a decision to increase the allocated OPERATE-to-safe sites duration from 13.5 sec to 14.1 sec by shifting the time of issuance of the OPERATE command 0.6 sec earlier than in free flight #10. Given the increased maximum duration for the OPERATE command, another set of Monte Carlo timing simulations was performed using a new Morpheus nominal trajectory delivery and the associated JSC Monte Carlo trajectory dispersions. The results of these Monte Carlo timing simulations are shown in Table 7 below. This table shows that the maximum expected OPERATE-to-safe sites duration increased for all 3 map cases, as expected. In addition, the 60x60m map case showed that the maximum OPERATE duration did exceed the allocation duration of 14.1 sec. Looking at the individual sample runs further revealed, however, that only 1 out of the 1000 runs in the 60x60m map case exceeded the allocation, and only barely by 0.02 sec. Figure 10 below shows a histogram of the predicted total OPERATE duration prior to free flight #11. This histogram shows that while the majority of the predicted OPERATE durations were less than the maximum, only 1 sample actually exceeded the requirement.

Table 7. Predictions for Third Set of Monte Carlo Timing Simulations (for FF #11-14)

Case #	DEM size	Slant Range mean (m)	Flight Path Angle mean (deg)	# Failed Mosaic Plans	Mean # Mosaic Points	Min # Mosaic Points	Max # Mosaic Points	OPERATE mean dur (sec)	OPERATE min dur (sec)	OPERATE max dur (sec)
1	60	462.530	31.191	0	532	528	631	11.947	11.205	14.120
2	55	462.530	31.191	0	483	479	492	10.756	10.189	11.658
3	50	462.530	31.191	0	367	361	442	7.774	7.205	9.319

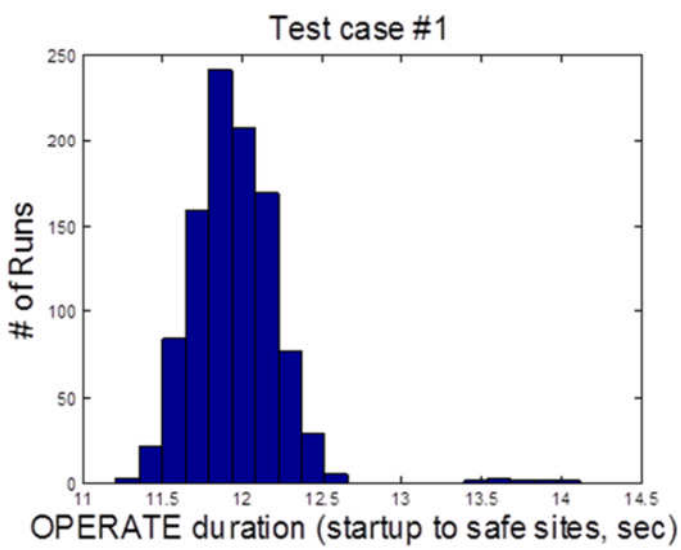


Figure 10. Distribution of OPERATE durations for a 60x60m map at mean and 1-σ values for the 04/21/2014 Morpheus Monte Carlo trajectory simulation dispersions and 04/15/2014 trajectory (prior to free flight #11)

V. Results

A. Hardware-in-the-loop Simulation Results

The hardware-in-the-loop simulation proved to be very accurate when compared to flight test data. For example, in preparation for the Morpheus free flight # 11 test, the simulation was used to predict the number of points in the mosaic plan and the time that it would take the gimbal to execute that plan. The simulation predicted 530 mosaic points, which coincided with the actual mosaic plan from the flight. The lab gimbal executed the mosaic only about 0.1 seconds faster than the flight gimbal. Figure 11 shows the mosaic plan for both the simulation and free flight #11, and also the points where the LIDAR boresight pointed to during the flight. Figure 12 shows the performance of the Gimbal Manager commands and the measured gimbal angles for both the simulation at the JPL lab and free flight #11.

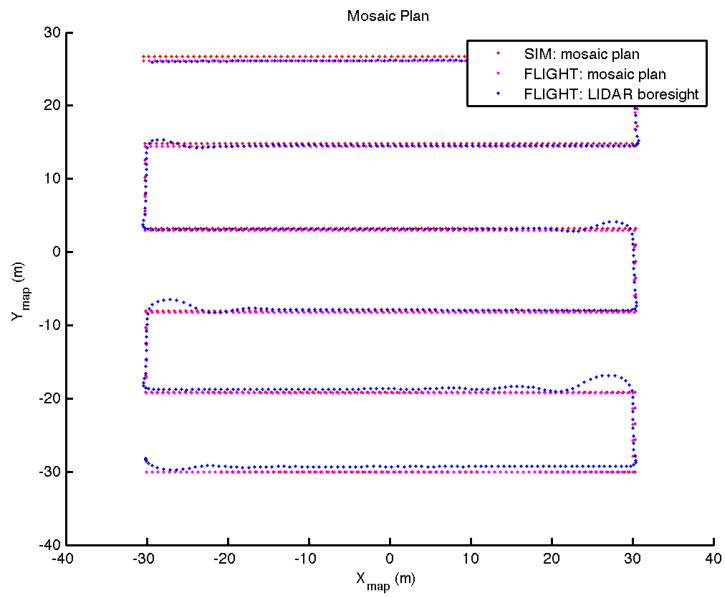


Figure 11. Comparison of Mosaic Planner Performance

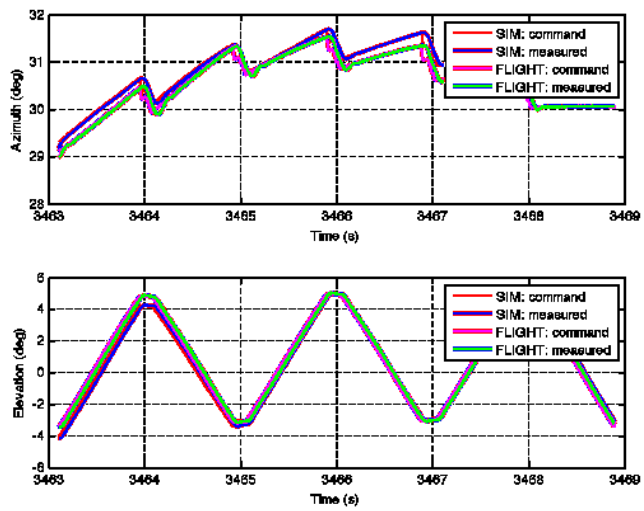


Figure 12. Comparison of Gimbal Manager/Gimbal Performance

B. Monte Carlo Simulation Results

The Monte Carlo timing simulation method was developed to accomplish two objectives: 1) validate the coarse timing method predictions and 2) provide higher fidelity predictions prior to the Morpheus vehicle free flights with the HDS hardware onboard. Three sets of Monte Carlo timing simulations were conducted to achieve these objectives prior to the five planned free flights. The first set of simulations was done to validate the coarse timing methods, the second to provide predictions prior to free flight #10, and the third to provide predictions prior to free flights #11 to 14. The first set of Monte Carlo timing simulations showed that for the 60x60m case, there were some vehicle flight geometries in which the total OPERATE-to-safe sites reporting duration exceeded the allocation. In the first set of Monte Carlo simulations, the mean and 1-σ values of the geometric parameters, however, were loose estimates. The second set of Monte Carlo timing simulations, in contrast, were based on expected vehicle position, velocity, and attitude dispersions from JSC-provided trajectory Monte Carlo simulations. This set of Monte Carlo predictions showed that for free flight #10 none of the predictions exceeded the allocated duration. The third set of

Monte Carlo timing simulations for free flights #11 to 14 showed that for the 60x60m case, only 1 of the 1000 runs exceeded the allocation.

Five Morpheus free flights with the HDS onboard were conducted in the spring of 2014, from April through May. Free flights #10 and #11 were "Open Loop" tests, free flight #12 was an "Advanced Open Loop" flight, and free flights #13 and 14 were "Closed Loop". "Open Loop" refers to the use of the vertical test-bed navigation system (VTB-Nav) onboard the Morpheus vehicle, in which the Morpheus IMU and GPS sensors provided measurements of the vehicle navigation state (position, velocity, and attitude) for use by the Autonomous Flight Manager (AFM) and the HDS, without any measurements from the HDS sensors. In the Open Loop flights, the vehicle landing site was pre-determined to be at the center of a concrete pad near the lower right corner of the hazard field, instead of being selected from the list of safe sites reported by the HDS. In the "Advanced Open Loop" flight, the vehicle landing site was allowed to be one of the safe sites returned by the HDS, while the vehicle was still being guided by VTB-Nav. In the closed loop flights, a second navigation system named ALHAT-Nav was used, in which measurements derived from the ALHAT Navigation sensors (the Doppler LIDAR, laser altimeter and HDS LIDAR) were included in the navigation filter to produce estimates of the vehicle state. In all cases, both the VTB-Nav and ALHAT-Nav state estimates were computed in real-time for comparison to one another. For the closed loop flights, the vehicle flight software (FSW) was designed to switch from ALHAT-Nav to VTB-Nav should certain tight tolerances between state estimates from the two sources be exceeded.

The total OPERATE-to-safe sites duration achieved in flight and the Monte Carlo predictions are given in Table 8 below. This table shows that the total OPERATE-to-safe sites durations for each flight were both within the allocation and within the predicted Monte Carlo bounds. Recall that for free flight #10, the duration requirement was 13.5 sec. To provide a little more margin for free flight #11, the timing of OPERATE was shifted earlier by 0.6 sec, thereby increasing the duration allocation to 14.1 sec.

Although the total OPERATE-to-safe sites duration was within the maximum allocation for all free flights, some of the component activity durations were outside the predicted bounds, as shown in Table 9. In this table, the red cells with "N" indicate the activities that had actual flight durations outside the Monte Carlo predictions. The "u" label indicates that the actual flight duration was below the minimum Monte Carlo prediction; the "o" label indicates that the actual flight duration was above or over the maximum Monte Carlo prediction; and the "r" label indicates that the Monte Carlo prediction was revised based on the previous flight's actual data.

For free flight #10, several of the OPERATE component activity durations were outside the predicted Monte Carlo bounds. In the Monte Carlo simulations for this flight, the reconnoiter point tracking duration had been assumed zero seconds given that the PREPARE command pre-pointed the gimbal at the desired point, but in flight this activity actually took a small time. The flight value was then used in the Monte Carlo predictions for the remaining free flights. The reconnoiter elevation bias duration in free flight #10 was twice the Monte Carlo prediction. To avoid this, the flight value was used in the Monte Carlo predictions for the remaining free flights. Only free flight #12 had a reconnoiter elevation bias duration slightly outside the prediction.

For the mosaic planning duration in free flight #10, the flight value was one third the minimum Monte Carlo prediction. Upon closer inspection it was found that the mosaic planning interval parameter (sec/mosaic point) had been overestimated (having been based on older KSC helicopter flight data). This parameter was then revised based on the actual mosaic planning duration observed in preparatory tests (an integration dry run and tether test 34) as well as free flight #10. After this correction, the mosaic planning duration was within predicted bounds for the remaining free flights.

Note that for free flight #12, the total OPERATE duration was within the timing requirement, but more than 1 sec larger than in free flight #11. Upon closer inspection, it was found that the flight DEM generation duration accounted for the majority of this time difference. Post flight campaign analysis revealed that the DEM generation interval per flash timing parameter had been underestimated in the Monte Carlo simulations. Future use of the DEM generation algorithm may require an improved method of estimating the DEM generation duration that accounts for variation in the DEM accumulation interval (sec/flash) parameter.

The DEM finalization flight durations were all above the maximum Monte Carlo predicted bound, except for free flight #11. Later analysis found that the timing studies on which the DEM finalization duration mean and 1-σ were based had assumed lower slant range than flown, predicted fewer flashes than seen in flight, and therefore slightly underestimated the DEM finalization duration.

The durations of the hazard detection and safe site selection process were within predicted bounds for all flights except free flight #13, where the flight duration was only slightly below the minimum Monte Carlo bound. The HRN feature selection flight durations were slightly above the maximum Monte Carlo bound for all flights. This duration, however, was not critical to flight operations because the timing requirement had been placed on the OPERATE-to-safe sites duration. Lastly, the duration after HRN feature selection to the reporting of the first HRN

update to the host vehicle was within predicted bounds for all flights except free flight #12, where the actual duration was below the minimum prediction.

Table 8. Total OPERATE-to-safe sites reporting duration results (flight vs. Monte Carlo predictions)

Parameter	FF10	FF11*	FF12*	FF13*	FF14*
Maximum OPERATE Duration Requirement (sec)	13.5	14.1	14.1	14.1	14.1
Monte Carlo Total OPERATE Min Duration (sec)	11.333	11.205	11.205	11.205	11.205
Monte Carlo Total OPERATE Mean Duration (sec)	11.995	11.947	11.947	11.947	11.947
Monte Carlo Total OPERATE Max Duration (sec)	12.721	14.120	14.120	14.120	14.120
Actual Flight Total OPERATE Duration (sec)	**12.230**	**11.498**	**12.856**	**11.332**	**11.391**

*The OPERATE command start was shifted 0.6 sec earlier than in FF #10 in these free flights

Table 9. Monte Carlo (MC) Timing Performance Table. u = flight value below min MC bound, o = flight value above max MC bound, r = MC estimate revised based on prior flight data

OPERATE Activity	FF10	FF11	FF12	FF13	FF14
Open Loop (OL) or Closed Loop (CL)	OL	OL	OL	CL	CL
Total OPERATE-to-safe sites	Y	Y	Y	Y	Y
Startup + In-range check	Y	Y	Y	Y	Y
Reconnoiter Point Tracking	**N (o)**	Y (r)	Y	Y	Y
Reconnoiter Elevation Bias Computation	**N (o)**	Y	**N (o)**	Y	Y
Mosaic Planning	**N (u)**	Y (r)	Y	Y	Y
Gimbal Mosaic Execution	Y	Y	Y	Y	Y
DEM generation (in parallel to mosaic execution)	Y	Y	Y	Y	Y
Finalize DEM	**N (o)**	Y	**N (o)**	**N (o)**	**N (o)**
Hazard Detection + Safe Site Selection	Y	Y	Y	**N (u)**	Y
HRN Feature Selection	**N (o)**	**N (o)**	**N (o)**	**N (o)**	**N (o)**
1st HRN update	Y	Y	**N (u)**	Y	Y

VI. Conclusion

Two separate simulations of the HDS were performed prior to the Morpheus free flights in spring 2014: a hardware-in-the-loop gimbal and flight software simulation and a Monte Carlo simulation of the OPERATE timing performance. For the OPERATE timing performance, three methods to estimate the total OPERATE-to-safe sites duration were employed. The two coarse timing methods revealed that a 60x60m map, or smaller, was most suitable for the expected vehicle trajectory geometry. A higher fidelity Monte Carlo timing simulation approach was also developed.

Overall, the total OPERATE-to-safe sites durations for all free flights were within the Monte Carlo prediction bounds, although some of the component activities were not. However, these outliers were not significant contributors to the total OPERATE duration. The major contribution to the total OPERATE-to-safe sites duration was DEM generation (~80%), followed by hazard detection and safe site selection (10%), then DEM finalization (5%), with the remaining 5% split among the other timing components.

In conclusion, the HDS hardware-in-the-loop gimbal simulations and OPERATE command timing analyses and Monte Carlo simulations provided excellent predictions of the number of mosaic points, gimbal performance, and OPERATE-to-safe sites duration under expected vehicle trajectory and attitude dispersions. These simulations

helped to provide confidence to the HDS team that the free flight command profile would execute within the maximum duration requirement and avoid a scenario where the host vehicle would otherwise bypass the HDS safe site list and land at a pre-designated abort landing site.

References

[1]Epp, C., and Smith, T., "Autonomous Precision Landing and Hazard Detection and Avoidance Technology (ALHAT)," *Proceedings of the IEEE Aerospace Conference*, March 2007, Big Sky, MT

[2]Villalpando, C. Y., Werner, R. A., Carson, J. M., Khanoyan, G., Stern, R.A., and Trawny, N. "A Hybrid FPGA/Tilera Compute Element for Autonomous Hazard Detection and Navigation," *Proceedings of the IEEE Aerospace Conference*, September 2013, Big Sky, MT

[3]Ivanov, T., Huertas, A., Carson, J. M., "Probabilistic Hazard Detection for Autonomous Safe Landing," *Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, August 19-22, 2013, Boston, MA.

[4]Trawny, N., Carson, J. M., Huertas, A., Luna, M. E., and Roback, V. E. "Helicopter Flight Testing of a Real-Time Hazard Detection System for Safe Lunar Landing," *Proceedings of the AIAA Space 2013 Conference and Exposition*, September 10-12, 2013, San Diego, CA.

[5]Carson, J. M., Bailey, E. S., Trawny, N., Johnson, A. E., Roback, V. E., Amzajerdian, F., and Werner, R. A., "Operations Concept, Hardware Implementation and Ground-Test Verification of a Hazard Detection System for Autonomous and Safe Precision Lunar Landing," *Proceedings of the AAS/AIAA Astrodynamics Specialist Conference*, Hilton Head Island, SC, Aug. 11–15, 2013

[6]Johnson, A.E., A. Huertas, R.A. Werner, and J.F. Montgomery, "Analysis of On-Board Hazard Detection and Avoidance for Safe Lunar Landing," *Proceedings of the IEEE Aerospace Conference*, 2008 IEEE, pp.1-9, 1-8 March 2008

[7]Huertas, A., A.E. Johnson, R.A. Werner and R.A. Maddock, "Performance Evaluation of Hazard Detection and Avoidance Algorithms for Safe Lunar Landings," *Proceedings of the IEEE Aerospace Conference*, 2010 IEEE, March 2010.

[8]A. Johnson, A. Klumpp, J. Collier and A. Wolf, "Lidar-based Hazard Avoidance for Safe Landing on Mars," *AIAA Journal of Guidance, Control and Dynamics* 25 (5), October 2002.

[9]C. Langley et al., "Recent Advancements of the Lidar-Based Autonomous Planetary Landing System." *Proc. 58th, Int'l Astronautical Conference*, Paper #IAC-07-A52, September 2007.

[10]A. Huertas and Y. Cheng, "Automatic Mapping of Lunar Craters and Boulders," *Proc. 42nd Lunar and Planetary Science Conference*, The Woodlands, TX, March 2011.

[11]D. 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).

[12]Hines, G. D. , D. F. Pierrottet and F. Amzajerdian, "High-fidelity flash lidar model development," *Proc. SPIE 9080*, Laser Radar Technology and Applications XIX; and Atmospheric Propagation XI, 90800D (June 9, 2014); doi:10.1117/12.2050677;