Pinpoint Landing through Landscape Matching and Terrain Evaluation

Andrew Sheng
Department of Computer Science
Carnegie-Mellon University
Pittsburgh, PA 15213
asheng@andrew.cmu.edu

Ina Fiterau
Machine Learning Department
Carnegie-Mellon University
Pittsburgh, PA 15213
mfiterau@cs.cmu.edu

Abstract

A critical part of a lunar surface mission is pinpoint landing. To land in a pre-designated location, the craft must be aware of its position, which this paper shows to be accomplishable through visual registering. The proposed Landmark Recognition System distinguishes lunar landmarks in images captured from orbit and estimates the coordinates at which the images were taken. The Surface Assessment Module chooses an unobstructed landing site which ensures the successful accomplishment of surface operations. Tests on photos from the Apollo missions and imagery taken by the Lunar Reconnaissance Orbiter demonstrate the system's effectiveness. Performance is highlighted through simulated landings. Reinforcing landing through Landscape Analysis has the potential to change the outlook of unmanned missions to charted celestial bodies.

1 Introduction

Pinpoint landing is a prerequisite to any lunar surface mission. Not only must the lander reach the moon unharmed, but it must also land as close as possible to a desired location. Moreover, the exact landing site must be chosen so that surface operations, such as rover egress, are easily performed – this is to be accomplished without human input. Thus, two issues arise. First, the lander must have a good estimate of its present position above the moon. Second, it is imperative that when close to the ground (50-200 meters), the lander seeks a location with a smooth, even surface devoid of rocks, craters and steep slopes.

The first problem is addressed through landscape matching, which achieves localization by establishing correlations between the observed landscapes and known lunar features. The reasoning is that only a very compact representation of the lunar surface is required rather than vast amounts of imagery. Also, the sheer number and diversity of lunar craters ensure that it is very unlikely two configurations will look

exactly alike. The latter issue, choosing a landing spot, is solved by evaluating the terrain up close and determining which portions of it are unsuitable for landing. As there are many ways to assess the suitability of a landing spot, several image processing methods are used for this purpose.

The system discussed here contains three key components: a database to record lunar surface features, a "Landmark Recognition System" (LRS) to determine lander position, and a "Surface Assessment Module" (SAM) to determine a desirable landing site.

The database of lunar features is organized into three parts: first, a set of non-processed orbital lunar photos displaying the moon's large structures, such as the Sea of Tranquility, from different angles; second, a KD-tree containing the position and radius of lunar craters extracted from LIDAR data and inserted in order of size; third, a series of low-altitude images gathered from various moon landing missions.

The Landmark Recognition System works in three stages. First, the LRS detects craters in an image of the lunar surface. Next, it estimates the scale and relative orientation of the moonscape based on the lander's attitude and altitude. Finally, it uses a graph matching algorithm to search the database for lunar terrain resembling the landscape beneath the lander. The LRS is shown to effectively distinguish landmarks such as the Apollo landing sites from orbital images and achieve effective localization using the known coordinates of the landmark.

The Surface Assessment Module is designed to pick a landing spot when the lander is close to the lunar surface. To do so, the SAM employs three mechanisms. First, a landing heat map generator uses a heuristic algorithm based on moving window filters to seek smooth terrain. Secondly, a landing spot classifier uses features extracted from pixel values trained on labeled segments of low-altitude images. The third is a surface model builder that estimates the shape of the terrain based on shading and stereo techniques. An estimation of the azimuth based on the shading is also derived at this level. This is significant because the lander must settle in an east-west direction so that the solar panels receive the maximum amount of sunlight.

The performance of the Landmark Detection System has been illustrated through simulation.

2 Related work

Yang Cheng and Adnan Ansar propose a scheme that solves landmark identification, lander velocity and position estimation [1]. They use ellipse fitting for crater detection, geometric recognition to match craters detected in real time to a database containing crater locations. To get a robust estimation of the lander position, the authors use projections penalized by least mean squares error. A Kalman filter model is used for velocity estimation. The paper explores the use of observable features to obtain position and velocity information about the lander. A similar approach is presented in [7].

Mourikis et al. [5] seek to make use of a mixed vision/inertial sensor system to achieve precise landing on a planetary surface. Although this group does not explicitly focus on crater detection, it does make use of an extended Kalman filter to process visual data in order to determine present vehicle position. This exploration offers useful insight into the possibility of tracking features between different images.

Johnson et al. [6] describe an algorithm for determining the present state of a planetary lander in descent. Although this algorithm is dependent on inertial measurements, it does offer information as to possible methods of localization.

The Autonomous Landing and Hazard Avoidance Technology (ALHAT) [8] program encompasses research into technologies that allow landers to measure and analyze topography of the landing site. Their technologies are tested with Monte Carlo simulations and through Hardware-in-the-Loop testbeds.

3 Landmark Recognition System (LRS)

This section presents the Landmark Recognition System, a solution to the pinpoint landing problem that uses correspondences between observed lunar terrain and a database of features to estimate the position of the lander.

An overview of our system is displayed in Figure 1. Active components are displayed in red rectangles, while data items are represented in blue. LIDAR (Light Detection and Ranging) Data from the Lunar Reconnaissance Orbiter (LRO) and LCROSS (Lunar CRater Observation and Sensing Satellite) impact images were used as inputs to a Feature Compiler, which built a database of lunar features. This database is updated as new information becomes available and a copy of it will be placed on the lander to be accessed upon descent. The image capture happens as the lander nears the target.

At high altitude, an image from the camera is sent to the crater detection module, which extracts a list of observed lunar features characterized by their position relative to each other and their size. The Landscape Matching algorithm finds correspondences between the observed features and the landmarks stored in the database. If the observed configuration of features is similar enough to a stored feature configuration, the system can infer the position along the lunar orbit where the lander was located at the time the image was captured.

As the Lander gets closer to the surface, the landmark data becomes less informative and attention is shifted towards determining a safe landing location. The Surface Assessment module accomplishes this by estimating the shape of the terrain based on the imagery and creates a heatmap representing the estimated probability of successfully landing at a given location.

Finally, either the position information or the landing heatmap, depending on the altitude of the craft, are used to guide the movement of the lander. Specifically the output of the Surface Assessment module will be considered when the lander is close enough to the surface according to a predetermined threshold and any of the following three conditions are true:

- according to landscape matching, the lander is in the correct area
- according to landscape matching, the lander is in an incorrect area, but nothing can be done to change it so the least that can be done is to attempt safe landing anywhere
- landscape matching can no longer provide useful information because of the proximity to the lunar surface

The homing system makes the high-level decisions concerning the subsequent target positioning of the lander.

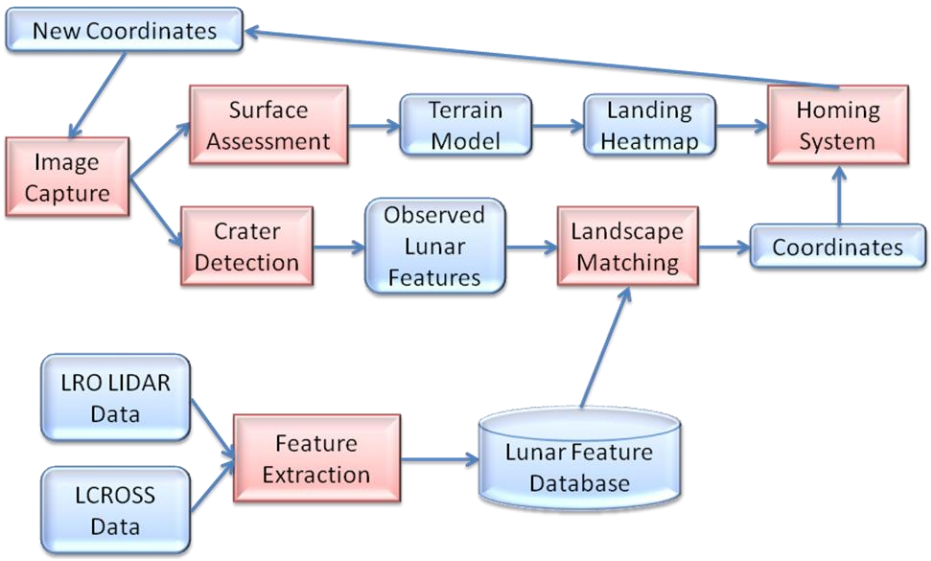


Figure 1: Overview of Landmark Recognition System

3.1 Available lunar data

LIDAR data necessary to create the lunar feature database is available from the LRO and the Clementine project. Although the Clementine data is more evenly spaced, the LRO data comes in an already-interpolated form, therefore the latter was used in feature extraction. The Lunar Reconnaissance Orbiter (LRO) has the most detailed imagery of the lunar surface, with a resolution of 100m/pixel. Images from this set were used in testing the crater identification and landmark matching modules.

Close-ups of the lunar surface were required to train and test the surface assessment module. The LCROSS probe is virtually the only source to get such images short of the Apollo photos. Most of the Apollo photos were not taken during descent, and the available ones matching this criterion are not sufficiently detailed to make training a classifier feasible. Training data was therefore limited to LCROSS photos.

One of the challenges of this project was selecting, from the vast amount of moon-related information available, the subset that is the most relevant for the intended task and processing it into usable form.

3.2 Feature extraction from LIDAR data

The LIDAR data that is used as ground truth needs to be processed so that it can be represented as a tree of features rather than a bi-dimensional array of heights. This task is performed by the Feature Compiler.

The Feature Compiler takes as input a matrix containing altitude data from a given region of the moon's surface and outputs a list of longitude and latitude coordinates and a radius for each detected features. The compiler scans along the latitudinal and longitudinal axes looking for local minima in terms of height – valleys. Once these are detected, second degree curves – parabolas and ellipses – are fitted to the surrounding points to obtain the size of the feature. The fitting error of a curve represents the mean squared difference between the actual height at each point and the estimated height at that point according to the curve. The final set of features is obtained by selecting the

fitted curve with the minimum fitting error from a region along an axis and cross validating with the features obtained along the other axis.

LIDAR data from the regions around the Apollo landing sites has been processed by the Feature Compiler and used as ground truth in testing the Landscape Matching algorithm. An example of features extracted from the interpolated LIDAR data in the region of the Apollo 11 landing site can be seen in Appendix A.

3.3 Crater Detection

In order to compare the landscape "seen" by the lander to the features in the existing database, it is necessary to detect the craters in an observed lunar landscape. This is the purpose of the Crater Detection module, which receives as input an image of the moon's surface as seen from above, the altitude at which this image was taken, the attitude of the Lander and the resolution of the camera. The module transforms the image, and finds indented circular patterns that represent craters.

To start with, the image is initially flattened to eliminate the curvature of the moon. Missing values due to the uneven data density distribution along the axes are interpolated. The crater fitting is accomplished by merging the outputs of several image processing procedures, since it was observed that not one method is superior to the rest for all test samples.

The basic algorithm for the first two methods is the same: edge detection followed by a Hough transform to find circles. The third method, developed for cell recognition [4], performs unsupervised segmentation by finding circular patterns through aggregation, not through Hough transform. It must be noted that standard edge detection methods such as canny, sobel or prewitt fail for lunar images because, despite the existing contrast obvious to the human eye, the image gradients are smooth. Using a lower threshold for the gradient does not work either, because in this case most accidental variations in contrast are incorrectly considered edges.

The first method detects crater edges by scanning each line and detecting the starting and ending points of a sequence that has descending values and then ascending values. The tolerance parameter this procedure receives represents the percentage of points that can be out of order. This method is also designed to allow for overlapping circles, which the subsequent procedures will cross-validate. The second method is based on comparing the color histogram in neighboring regions of the image to detect edges.

Both of these procedures take as a parameter a range – minimum and maximum values for the radius of the circle. Craters of various sizes are detected by running the procedures multiple times for different ranges. For a given range, the output of the procedures is combined by taking the union of the detected sets for each procedure and eliminating heavily overlapping circles by majority vote – the circle that overlaps with the most other circles is kept.

The final output of the module is a list of (x,y,r) values, where the (x,y) represent the position of the center of the crater and r is its radius. All the values are scaled to represent the actual dimensions of the landscape; the calculations are based on the resolution of the camera and the known altitude of the lander.

3.4 Landscape Matching

The Landscape Matching algorithm finds correspondences between the craters found by the Crater Detection Module and the features that exist in the database. It takes as an input the list of craters observed by the lander during descent, a value for size tolerance and one for distance tolerance. The tolerance is required since it is very

unlikely that there will be any exact matches between the observations and the features recorded in the database. From previous iterations of the algorithm, a subset of the database has been selected. The subset, which is called ‘reference’, corresponds to the response to a range query in the kd-tree. The range corresponds to an upper bound on the distance the lander had traversed since the last run of the algorithm.

First, the algorithm determines candidate reference-observation couples representing features that can be mapped to each other based on their size. For each pair of couples, it is determined if they are mutually exclusive for the given distance tolerance, that is, whether the distance between references is close enough to the distance between observations. If the distance between two observations is not within tolerance of the distance between references, it is impossible that both observation-reference mappings are correct because we are assuming that the distance between the two features must be preserved, even as the features are considered in different models – the observed image and the feature database. The result after this step is a set of mappings, some of which are mutually exclusive.

Of interest is the largest subset of mappings for which any two mappings are not exclusive. Were the mutual exclusive relationship represented in a graph, the subset corresponds exactly to the largest clique in the graph. It is a well known fact that determining the largest clique is both an NP hard problem and difficult to optimize. Fortunately, in this case it is only necessary to determine the maximum clique during the first run of the algorithm and only when sanity check is required. At all other times, it is enough to have one or several control observation-reference couples. Therefore, the algorithm can use an iterative process where the mappings that are consistent with the controls are kept and become controls, while the rest are discarded.

Three controls are sufficient to ensure that there are no ambiguous cases when two points are consistent with the controls, but not with each other. Error recovery happens when, during sanity check, there is too little overlap between the new control points and the old ones, where the overlap is measured in terms of number of pairs that are not mutually exclusive.

3.5 Surface Assessment

The Surface Assessment Module is designed to determine which portions of an observed area are fit for landing. The problem is imbalanced in the sense that a false positive – a spot that is deemed appropriate for landing but really isn’t – could have a devastating effect on the mission.

To start with, a heuristic method of generating a heatmap has been developed. For each pixel from the input image projected onto the horizontal plane, the algorithm computes the maximum distance between the initial pixel value and the values obtained by applying square moving average filters of dimensions varying from 2 to the image height. The intuition behind this is that, for even surfaces, the pixel values will change less after applying a moving average filter than for portions representing craters. Once the values are normalized, cross validation can be used to obtain a threshold so that asymptotically (i.e. as the number of samples increases), the false positive error is below a desired value.

A different way to detect landing spots is to train a classifier based on a series of features that characterize each pixel from a set of manually labeled images. The set of 10 features include information about the distribution of values in the vicinity of the pixel. The following classification methods were used: tree-augmented Naïve Bayes, logistic regression, K-nearest neighbors, decision trees, support vector machines. In the case of logistic regression, an empirical bound on the false positive error can be

obtained by learning the threshold above which a point is appropriate for landing. It is to be noted however that with the decrease of false positive error comes an increase in false negative error, up to the point where the elimination criteria will be so harsh that there will be virtually no candidate landing spots.

The method by which the lander determines the relative height of terrain features is via use of stereo imaging. The Birchfield-Tomasi stereo discontinuity algorithm is adequate given terrestrial images; however, its present effectiveness on lunar terrain is highly dependent on the placement of lander stereo cameras and lunar lighting conditions. This algorithm is applied as follows: The lander takes two pictures of the same region of the lunar surface from two different points in its orbit in order to simulate a large stereo vision system. The baseline can be estimated by comparing the size of the same feature in the two consecutive images. The two images are then fed into the Birchfield algorithm in order to generate a disparity map of the lunar surface. From this disparity map, the relative elevations of various terrain features are determined. Determining a landing spot is, at least theoretically, straightforward.

Light tracking is used to obtain the trajectory of the brighter parts of the image. From their variation, as well as from the relative positioning of the camera, we can obtain sequences of movement of the sun. From these sequences, the angle the lander has with respect to the desired east-west axis is calculated and the attitude is then adjusted accordingly.

4 Evaluation

The evaluation of the LRS was performed in two stages, the first one consisting of unit testing each of the components, and the second of a descent simulation.

4.1 Testing the Crater Detection Module

The crater detection method was tested with LRO images depicting the Apollo landing sites. A crucial aspect must be pointed out: the crater detection need not be exact. As long as an amount of landscape information is obtained that is sufficient to distinguish one landscape from the other, the landmark matching algorithm will unambiguously determine the location of the Lander – when the ellipse fitting is accurate. Appendix B shows the three methods of crater detection applied to an LRO image. Also, the crater detection module is used in testing the landmark matching module – Table 1.

4.2 Testing the Landmark Matching Module

The landmark matching module was also tested using LRO images of the Apollo sites. A set of 20 other images randomly selected from the LRO repository were used a negative examples. In each test, an image from a landing site was cropped to contain only portions of the landscape, resulting in five overlapping images. Two sets of four tests have been performed, for the Apollo 11, 14, 16, 17 locations, each test using 25 images, 5 of which were positives.

In the first set of tests, the table containing lunar features was manually created as appearing in the World Wind application [3]. For the second set of tests, the extracted features from the LRO LIDAR data were used. The performance of the Landmark Matching process is illustrated in Table 1. For the manually generated feature set, the performance is perfect. For the feature set obtained from the LIDAR data, the system made some mistakes because of the discrepancies that appear between the LIDAR data and the LRO images after processing. The reason why this does not present a problem in the simulation is because the LRS does not match a single image, but a

sequence of images and also uses the orbital images in the highest level of the database to verify the estimation.

Table 1: Results of Landmark Matching

Test Set 1	True Positives	False Positives	True Negatives	False Negatives
Apollo 11	5/5	0/20	20/20	0/5
Apollo 14	5/5	0/20	20/20	0/5
Apollo 16	5/5	0/20	20/20	0/5
Apollo 17	5/5	0/20	20/20	0/5
Test Set 2				
Apollo 11	4/5	3/20	17/20	1/5
Apollo 14	3/5	4/20	16/20	2/5
Apollo 16	5/5	3/20	17/20	0/5
Apollo 17	4/5	4/20	16/20	1/5

4.3 Testing the Surface Assessment Module

To achieve classification with the heuristic SAM, a threshold must be selected so that the probability of a false positive is below a user-specified value α. Table 2 shows the false negative rate β for a set of values for a set of limiting false positive rates, as evaluated on leave-one-image-out cross validation on a set of five images from the LCROSS repository.

Table 2: Error Rates for heuristic SAM

False Positive Rate	**0.1**	**0.01**	**0.001**	**0.0001**	**0.00001**
False Negative Rate	0.056	0.121	0.482	0.578	0.924

The trained SAM was also evaluated using LOO cross validation on LCROSS images – an image was left out, the classifier was trained on the remaining 4 images, then evaluated on the image that was left out. Table 3 shows the achieved accuracies for each type of classifier. Accuracy represents the percentage of (manually labeled) test points that are were correctly classified as acceptable/not acceptable landing spots.

Table 3: Accuracy for trained SAM

Classifier	**TAN**	**LR**	**KNN**	**DT**	**SVM**
Accuracy	0.75	0.79	0.67	0.71	0.82

Evaluation of the stereo procedure has not yielded good results since the horizon line is missing. Available packages are not specifically designed to work with landscapes taken from this angle or this type of barren, colorless terrain.

4.4 Simulation of Homing System

The experiments presented so far were aimed at evaluating individual components of the LRS. Testing the system as a whole presents some challenges, mainly due to the difficulty of performing reliable field tests. The TAM requires close-ups of the moon's surface that aren't widely available. Also, although high resolution images of

the moon's surface are available from lunar orbiters, these are all taken from a direction perpendicular to the moon's surface.

The developed simulation framework uses the World Wind SDK [reference needed], which comprises of imagery from the Clementine mission. The simulation algorithm is depicted in Algorithm 1. Appendix D presents a sequence diagram of how the simulation works. The Lander starts from a given altitude and set of coordinates. An image corresponding to those coordinates is taken from the World Wind API and fed to the LRS. The LRS runs the crater detection module and the landscape matching algorithm comparing the observed craters with reference features selected based on the previously known position of the Lander.

To determine the initial position, the observed features are compared to all the high levels (i.e. of greater size) features in the kd-tree to find the position. Also, the captured image of the landscape is compared to the images in the first partition of the database – the orbital lunar photos, in order to validate the initial assessment of the position. After the position is estimated, control points are established and the estimated position is sent back to the simulator, which checks it. The LRS also requests a direction of movement depending on the target. The simulator uses the movement model of the Lander to update the position. The algorithm is then repeated. The simulation has achieved successful targeting 5 out of 5 times for Apollo 16, 4 out of 5 for Apollo 11, 14, and 17.

Algorithm 1: Simulation of Homing System

```
def run_simulation(initial_position, initial_altitude, destination):
        #setup initial position
        actual_coor = inital_position
        altitude = initial_altitude
        while (altitude > 0):
                # accesses World Wind API for view from given location
                image = worldwind_get_image(coor,altitude)
                # feeds image to LRS in order to estimate current position
                estimated_coor = lrs_analyze_image(image,altitude)
                # compares current and desired coordinates to determine flight direction
                direction = compare_coor(estimated_coor, destination)
                # moves lander in desired direction and updates altitude
                actual_coor, altitude = move_lander(actual_coor,altitude,direction)

def lrs_analyze_image(image,altitude):
        # calls crater detection algorithm on the image
        craters = detect_craters(image)
        # scales for lander altitude
        craters_mod = adjust_for_range(craters,altitude)
        # calls landscape recognition algorithm and compares with database information
        return compare_with_database(craters_mod)
```

Conclusions

This investigation into mechanisms for solving the problem of Lander localization and trajectory planning has yielded a system capable of examining lunar terrain and comparing it to a local database. From this information, the system effectively determines its present location above the moon and the location of a landing site devoid of obstructions such as craters and steep slopes. It is therefore safe to conclude that the discussed system would facilitate the success of a lunar landing mission.

Acknowledgments

This paper was written for the 16-861 Mobile Robot Design class at Carnegie Mellon University.

References

[1] Yang Cheng; Ansar, A., *Landmark Based Position Estimation for Pinpoint Landing on Mars*, Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on , vol., no., pp. 4470- 4475, 18-22 April 2005

[2] S. Birchfield and C. Tomasi, *Depth Discontinuities by Pixel-to-Pixel Stereo*, International Journal of Computer Vision, 35(3): 269-293, December 1999

[3] Java World Wind SDK: worldwind.arc.nasa.gov/java/

[4] Cell segmentation: blogs.mathworks.com/steve/2006/06/02/cell-segmentation/

[5] Anastasios I. Mourikis, Nikolas Trawny, Stergios I. Roumeliotis, Andrew E. Johnson, Adnan Ansar, and Larry Matthies. 2009. *Vision-aided inertial navigation for spacecraft entry, descent, and landing.* Trans. Rob. 25, 2 (April 2009)

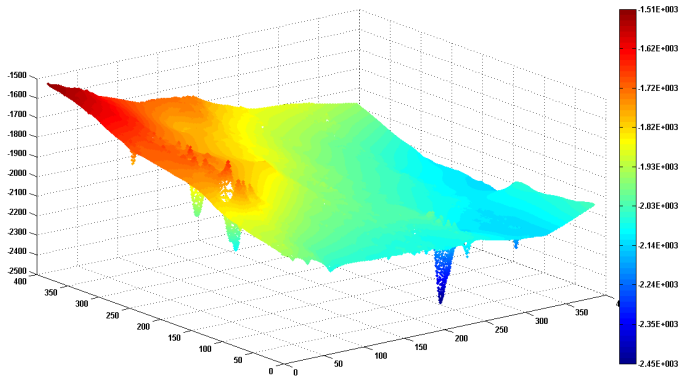
[6] Johnson A., Ansar A., L. Matthies, N. Trawny, A. I. Mourikis, and S. I. Roumeliotis, *A General Approach to Terrain Relative Navigation for Planetary Landing*, AIAA Infotech at Aerospace Conference, Rohnert Park, CA, May 2007.

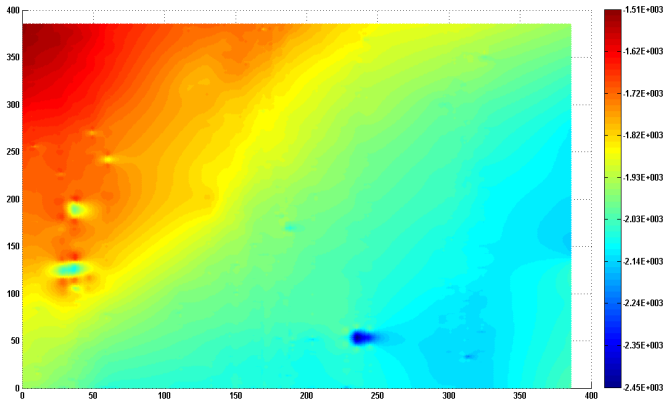
[7] Yang Cheng, Andrew E. Johnson, Larry H. Matthies, Clark F. Olson, *Optical Landmark Detection for Spacecraft Navigation*, In Proceedings of the 13th Annual AAS/AIAA Space Flight Mechanics Meeting, Ponce, Puerto Rico, February 2002.

[8] http://alhat.jpl.nasa.gov/

Appendix A – Feature Compiler Example

LIDAR Data (Apollo 11 site):



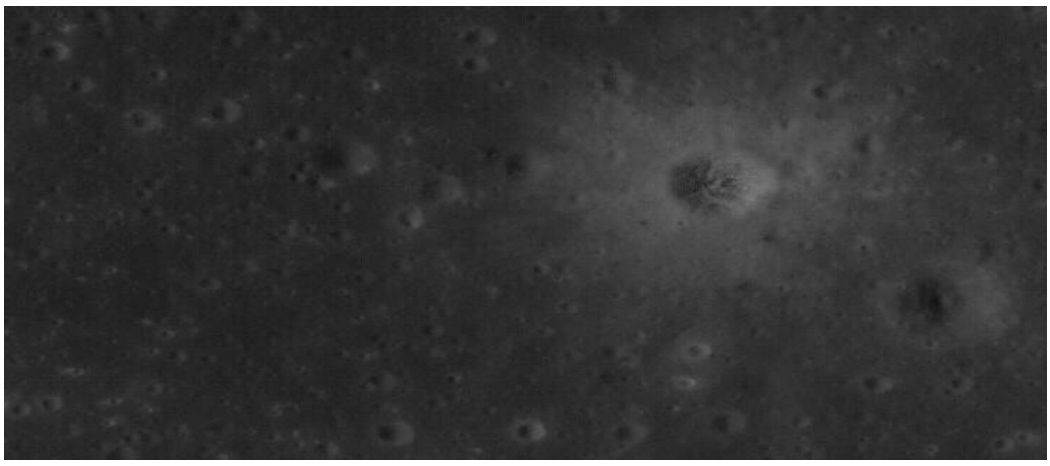


Extracted features:

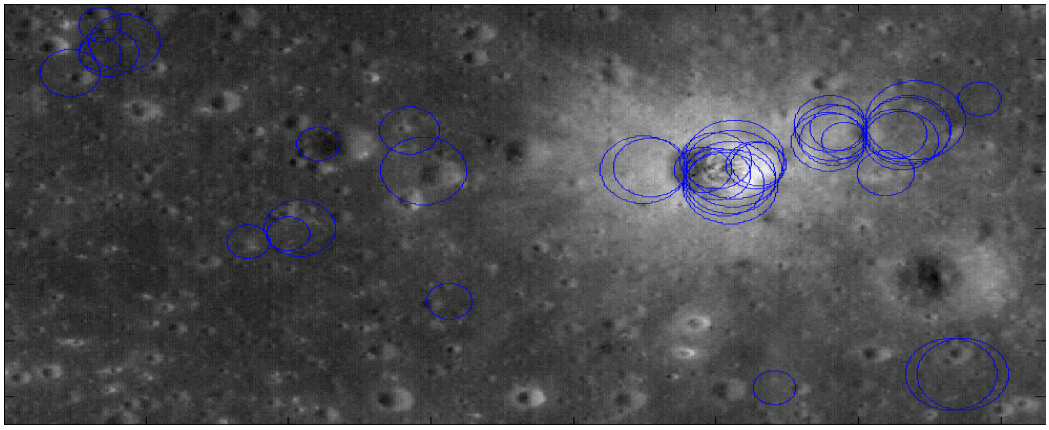
X	Y
235	54
27	126
42	189
212	179
55	242

Appendix B – Crater Detection Results on LRO image

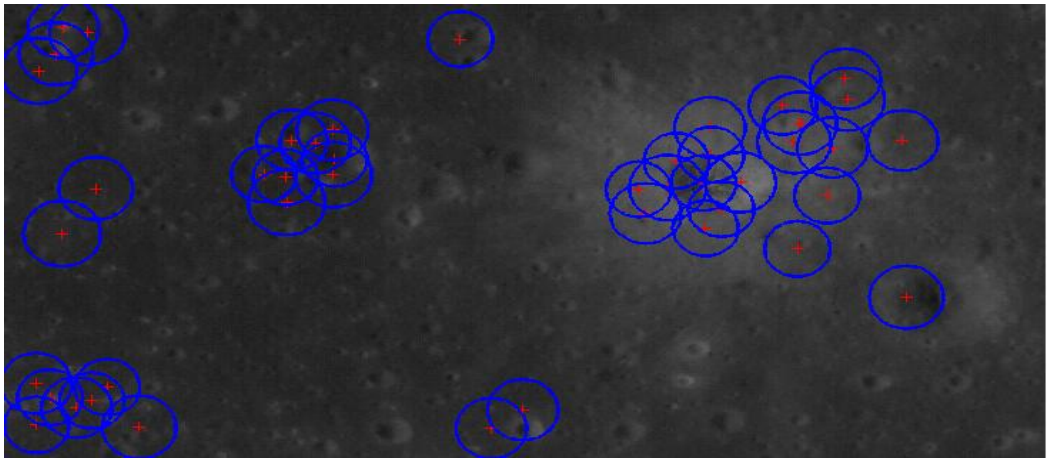
Original image:



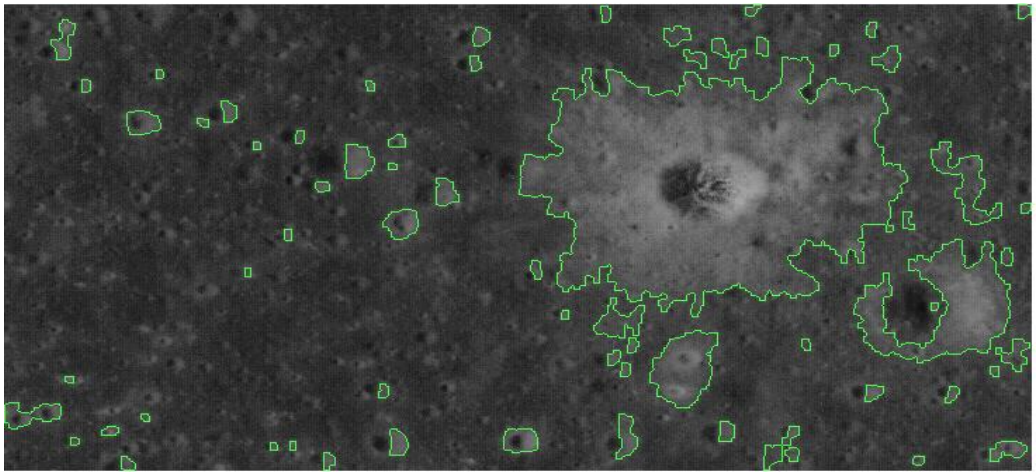
Circles detected with gradient edge detection + Hough transform:



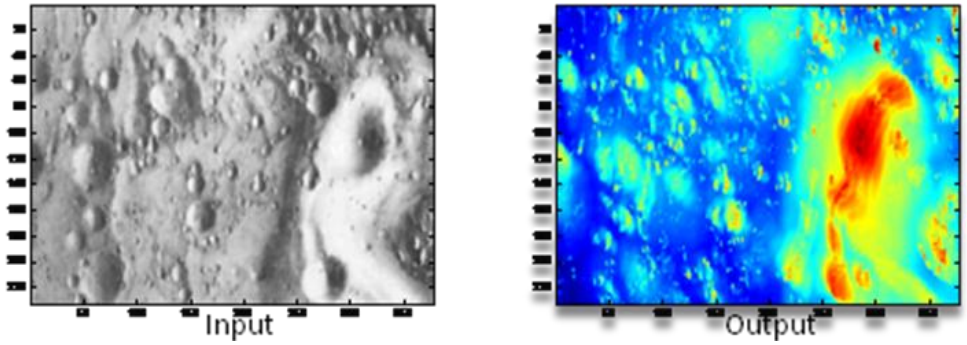
Circles detected with histogram edge detection + Hough transform:



Circles detected with segmentation for cell recognition:



Appendix C – Surface Assessment: Heat-map Example (Apollo 11 Site)



Appendix D – Sequence diagram of Simulation