(Preprint) AAS 22-747

LIDAR ODOMETRY FOR LUNAR TERRAIN RELATIVE NAVIGATION

Carl De Vries[*], John Christian[†], Michael Hansen[‡], and Tim Crain[§]

Future missions to the lunar surface are expected to make use of LIDAR sensors for navigation during landing. This is especially true when the lunar landing must be accomplished under lighting conditions that are undesirable for camera-based navigation. Moreover, when local maps of the lunar terrain are also poor or unavailable to the lander in real-time, concepts from LIDAR odometry (LO) are highly relevant. This work develops an algorithmic framework for LO suitable for supporting future lunar exploration missions.

INTRODUCTION

Recent years have seen an increased interest in both lunar exploration and the development of a lunar economy. This interest has spawned numerous spacecraft development programs by government space agencies, academia, and the commercial sector. These lunar exploration activities possess a variety of technical challenges, with autonomous navigation being one of the most difficult tasks. Moreover, while the challenges of autonomous navigation are many, this is especially true for landing. This work addresses the use of LIDAR sensors for terrain relative navigation (TRN) during a lunar landing over a poorly mapped region of the Moon.

There are a variety of approaches and sensing technologies that could be used for lunar TRN, and many of these approaches have been studied for some time.[1] If solar illumination is available and reliable, there are many advantages for using conventional cameras as the primary TRN sensor. When the terrain is well-mapped, it is often possible to match observed image features to cataloged features in a map—as was done on OSIRIS-REx[2] and with the Mars Perseverance Rover's Lander Vision System (LVS).[3,4] It is also possible to observe and match explicit geological features, such as craters.[5] When the terrain is not well-mapped, much can be done with camera-based visual odometry (VO),[6] though this comes as the cost of a loss of absolute scale. Concepts from VO are the fundamental idea behind the TRN algorithms used for the Mars Exploration Rovers,[7] the Mars Ingenuity helicopter,[8] and the upcoming IM-1 lunar lander mission.[9,10] However, if a landing must be performed when lighting is poor, then camera-based navigation systems may not be able to reliably detect and match surface features.

One of the great advantages of LIDAR systems[11] is that they carry their own illumination source, and thus do not depend on solar illumination. This sometimes makes LIDARs preferable to cameras for night-time landings or for landing in lunar polar regions. LIDARs also have the advantage

[*] Graduate Student, Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30364.

[†] Associate Professor, Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30364.

[‡] Nova-C Navigation Lead, Intuitive Machines, Houston, TX 77058.

[§] Chief Technology Officer, Intuitive Machines, Houston, TX 77058.

of explicitly measuring range (whereas absolute range may only be inferred from TRN camera images by matching observed landmarks to a map). However, TRN with LIDARs is not a panacea. Indeed, there are numerous challenges associated with 3D point cloud registration—either between a measured point cloud and a topographic map (the 3D equivalent to the OSIRIS-REx and Mars Perseverance LVS mentioned above) or between two measured point clouds (the 3D equivalent to the VO methods used on MER, Mars Ingenuity, and IM-1). Since this work is interested in landing over poorly mapped terrain, the discussions that follow will focus on the latter of these two cases—the so-called LIDAR odometry (LO) problem.

This work develops an approach to estimate a spacecraft's translation using LIDAR point clouds generated at two different positions. This is achieved by matching features which exist in both of the observed point clouds, and so the vehicle's motion must be small enough that considerable overlap exists in the two LIDAR measurement sets. Moreover, since the point cloud registration is based on 3D geometry, it is assumed that the terrain has sufficient relief to generate high-quality keypoints with unique feature descriptors.

SPACECRAFT AND LUNAR SURFACE GEOMETRY

Consider a spacecraft passing above the lunar surface equipped with a LIDAR sensor as illustrated in Figure 1. The onboard LIDAR captures 3D point clouds of the surface as it moves along it's trajectory. To describe the geometry of this scenario, begin by defining $\boldsymbol{r}_{I_k}$ as the location of the LIDAR in a Moon-centered inertial frame (e.g., coordinate axes aligned with ICRF) at time t_k. This is a convenient starting point since most navigation filters estimate the vehicle state in an inertial frame. Now, since the LIDAR is observing the lunar surface, it is necessary to know the LIDAR's position relative to the Moon. Thus, defining $\boldsymbol{T}^{I}_{M_k}$ as the transformation from the inertial frame to a Moon-fixed frame at time t_k, we see that

$$\boldsymbol{r}_{(M_k)_k} = \boldsymbol{T}^{I}_{M_k} \boldsymbol{r}_{I_k} \tag{1}$$

Now, define $\boldsymbol{s}$ as the Moon-relative change in position between two successive times,

$$\boldsymbol{s}_{M_k} = \boldsymbol{r}_{(M_k)_k} - \boldsymbol{r}_{(M_{k-1})_{k-1}} = \boldsymbol{T}^{I}_{M_k} \boldsymbol{r}_{I_k} - \boldsymbol{T}^{I}_{M_{k-1}} \boldsymbol{r}_{I_{k-1}} \tag{2}$$

Let $\boldsymbol{p}^{(i)}$ be the position of an arbitrary point on the lunar surface relative to the center of the Moon. Also, let $\boldsymbol{\ell}^{(i)}$ be the position of this same point on the lunar surface relative to the LIDAR sensor. In any consistent frame, it follows that

$$\boldsymbol{p}^{(i)} = \boldsymbol{r} + \boldsymbol{\ell}^{(i)} \tag{3}$$

For TRN applications, it is helpful to express this problem in the MCMF frame. Moreover, since $\boldsymbol{\ell}^{(i)}$ comes from the LIDAR it is natively expressed in the LIDAR frame. Consequently, we may write

$$\boldsymbol{p}^{(i)}_{M_k} = \boldsymbol{T}^{I}_{M_k} \boldsymbol{r}_{I_k} + \boldsymbol{T}^{L_k}_{M_k} \boldsymbol{\ell}^{(i)}_{L_k} \tag{4}$$

or, equivalently,

$$\boldsymbol{T}^{I}_{M_k} \boldsymbol{r}_{I_k} = \boldsymbol{p}^{(i)}_{M_k} - \boldsymbol{T}^{L_k}_{M_k} \boldsymbol{\ell}^{(i)}_{L_k} \tag{5}$$

Direct use of this expression requires knowledge of the LIDAR orientation relative to the MCMF frame. Access to this attitude transformation is usually a good assumption for a lunar lander with a star tracker, where the vehicle's inertial attitude $\boldsymbol{T}^{I}_{B_k}$ is known. Then, if the LIDAR's orientation relative to the vehicle's frame $\boldsymbol{T}_{L_k}^{B_k}$ is also known, we may directly compute the transformation between the MCMF and LIDAR frames.

Substituting the result into Eq. (2) for times t_{k-1} and t_k yields

$$\boldsymbol{s}_{M_k} = \left(\boldsymbol{p}_{M_k}^{(i)} - \boldsymbol{T}_{M_k}^{L_k}\boldsymbol{\ell}_{L_k}^{(i)}\right) - \left(\boldsymbol{p}_{M_{k-1}}^{(i)} - \boldsymbol{T}_{M_{k-1}}^{L_{k-1}}\boldsymbol{\ell}_{L_{k-1}}^{(i)}\right) \tag{6}$$

Assuming the observed surface point $\boldsymbol{p}^{(i)}$ is stationary on the lunar surface (as is the case for most natural features), then its MCMF position is fixed for all time and $\boldsymbol{p}_M^{(i)} = \boldsymbol{p}_{M_k}^{(i)} = \boldsymbol{p}_{M_{k-1}}^{(i)}$. Hence,

$$\boldsymbol{s}_{M_k} = \boldsymbol{T}_{M_{k-1}}^{L_{k-1}}\boldsymbol{\ell}_{L_{k-1}}^{(i)} - \boldsymbol{T}_{M_k}^{L_k}\boldsymbol{\ell}_{L_k}^{(i)} \tag{7}$$

LIDAR DATA PRE-PROCESSING

In this work, the LIDAR measurement is assumed to be a 3D point cloud consisting of n points expressed in the LIDAR frame, $\{\boldsymbol{\ell}_{L_k}^{(i)}\}_{i=1}^{n_k}$. If these points are generated by a scanning LIDAR, it is likely that they are not in a regular grid pattern. If two such point clouds exist, $\{\boldsymbol{\ell}_{L_{k-1}}^{(i)}\}_{i=1}^{n_{k-1}}$ and $\{\boldsymbol{\ell}_{L_k}^{(i)}\}_{i=1}^{n_k}$, then our objective is to find the LIDAR sensor translation that shifts the first point cloud onto the second.

It is possible to find this translation by working directly with the 3D point cloud data, either through aligning the point clouds as a whole (e.g., with ICP[12] or TEASER[13]) or by matching 3D feature descriptors (e.g., with feature histograms[14] or other techniques[15]). However, working with an unstructured point clouds or irregular meshes constructed from these point clouds, is algorithmically complicated, computationally expensive, and not usually necessary for this application. It is usually easier to resample the terrain observations into an elevation map on a 2D grid.

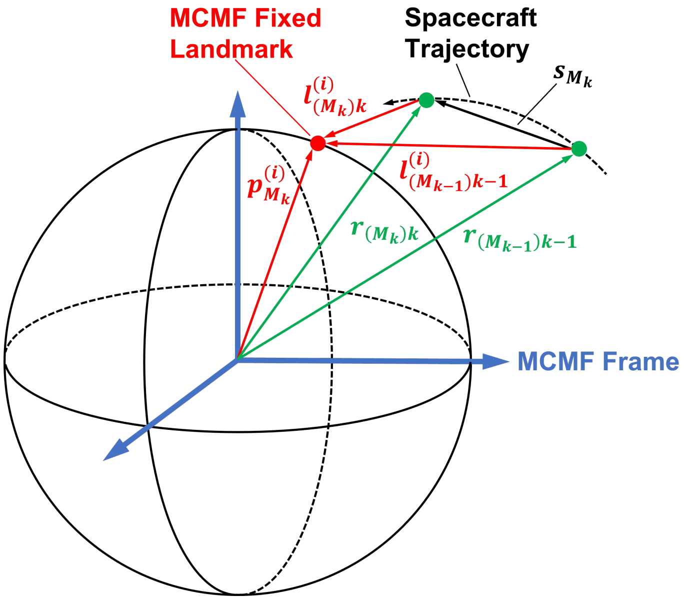

Figure 1. Multiple view geometry of a fixed landmark from a single translating spacecraft.

The LIDAR pre-processing starts by determining the projection plane and the proper reference frame to represent the point clouds in for the projection. A regular grid is then formed in the projection planes and 2D images are generated by appropriately sampling the LIDAR points in the 2D grid structures. After features are matched in the 2D image, the corresponding 3D points are constructed, and then a process to identify and reject incorrect matches is applied.

Constructing the Reference Plane for LIDAR Gridding

There are at least two reasonable ways to construct a reference plane for the gridding of measured LIDAR data. The first is to grid the data in the sensor frame, which results in a *depth map* (this happens automatically if the LIDAR happens to be a Flash LIDAR). The second is to grid the data on a reference plane tied to the local terrain (e.g., the Moon's local horizontal), which results in an *elevation map*. This work chooses to grid the data on an estimate of the terrain-anchored reference plane.

There are a few different ways one might compute the reference plane. Two such ways are discussed here. The first way is to assume the terrain relief is small as compared to the spatial extent of the point cloud, which tends to be a good assumption at large altitudes and a poor(er) assumption at low altitudes. The second way is to use knowledge of the spacecraft's global location above the Moon to construct the local horizontal. Each of these approaches are now discussed in more detail.

Best Fit Plane The best fit plane may be constructed directly from the LIDAR scan points $\{\boldsymbol{\ell}_{L_k}^{(i)}\}_{i=1}^{n_k}$. To do this, first write the point $\boldsymbol{\ell}_{L_k}^{(i)}$ in homogeneous coordinates (a point in $\mathbb{P}^3$),

$$\bar{\boldsymbol{\ell}}_{L_k}^{(i)} \propto \begin{bmatrix} \boldsymbol{\ell}_{L_k}^{(i)} \\ 1 \end{bmatrix} \tag{8}$$

Now, the point $\bar{\boldsymbol{\ell}}_{L_k}^{(i)} \in \mathbb{P}^3$ lies in the plane $\boldsymbol{\pi}_{L_k} \in \mathbb{P}^3$ when

$$\boldsymbol{\pi}_{L_k}^T \bar{\boldsymbol{\ell}}_{L_k}^{(i)} = 0 \tag{9}$$

It is therefore possible to estimate the best fit plane in the least squares sense as

$$\begin{bmatrix} \left(\boldsymbol{\ell}_{L_k}^{(1)}\right)^T & 1 \\ \left(\boldsymbol{\ell}_{L_k}^{(2)}\right)^T & 1 \\ \vdots & \vdots \\ \left(\boldsymbol{\ell}_{L_k}^{(n_k)}\right)^T & 1 \end{bmatrix} \boldsymbol{\pi}_{L_k} = \mathbf{0}_{n_k \times 1} \tag{10}$$

This is a null-space problem and the solution for $\boldsymbol{\pi}$ may be found via the singular value decomposition (SVD).

Since everything is in the LIDAR frame, the sensor boresight is along the z-axis by convention. The intersection of this direction with the plane $\boldsymbol{\pi}_{L_k}$ may be found analytically. Begin by expressing the z-axis in Plücker coordinates as

$$\boldsymbol{L}_z \propto \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{11}$$

Recall that the intersection point $\bar{\boldsymbol{x}} \in \mathbb{P}^3$ of a line described by the Plücker matrix $\boldsymbol{L}$ with a plane $\boldsymbol{\pi} \in \mathbb{P}^3$ is given by $\bar{\boldsymbol{x}} \propto \boldsymbol{L}\boldsymbol{\pi}$. Thus, the intersection of the LIDAR boresight direction with the plane is the point $\boldsymbol{x}_{L_k} \in \mathbb{R}^3$ given by

$$\begin{bmatrix} \boldsymbol{x}_{L_k} \\ 1 \end{bmatrix} \propto \boldsymbol{L}_z \boldsymbol{\pi}_{L_k} = \begin{bmatrix} 0 \\ 0 \\ -\pi_4 \\ \pi_3 \end{bmatrix} \tag{12}$$

which is the same as

$$\boldsymbol{x}_{L_k} = \begin{bmatrix} 0 \\ 0 \\ -\pi_4/\pi_3 \end{bmatrix} \tag{13}$$

Moreover, we can compute the plane normal vector $\boldsymbol{n}_{L_k} \in \mathbb{R}^3$ as

$$\boldsymbol{n}_{L_k} \propto \boldsymbol{S}\boldsymbol{\pi}_{L_k} \tag{14}$$

where $\boldsymbol{S} = [\boldsymbol{I}_{3\times3}, \boldsymbol{0}_{3\times1}]$. We choose the scaling $\|\boldsymbol{n}_{L_k}\| = 1$ and select the sign such that $\boldsymbol{n}_{L_k}^T \boldsymbol{k} < 0$. This makes the normal vector point towards the LIDAR sensor (out of the lunar surface).

Local Horizontal from Filter State It is possible to construct the local horizontal using the filter's current state and knowledge of the LIDAR sensor's attitude. If we assume that the spacecraft altitude is small as compared to the radius of the Moon (a good assumption for the final phases of landing considered here), then the local horizontal is perpendicular to the position vector.

Thus, to begin, we may express the local horizontal in the inertial frame according to

$$\boldsymbol{\pi}_{I_k} \propto \begin{bmatrix} \boldsymbol{n}_{I_k} \\ -R_M \end{bmatrix} \tag{15}$$

where $\boldsymbol{n}_{I_k} = \boldsymbol{r}_{I_k}/\|\boldsymbol{r}_{I_k}\|$ is the plane normal (unit vector in MCI direction of the spacecraft) and R_M is the radius of the Moon.

If the inertial boresight is $\boldsymbol{b}_{I_k} = \boldsymbol{T}_I^{L_k}\boldsymbol{k}$ then the Plücker matrix is,

$$\boldsymbol{L}_b \propto \begin{bmatrix} [(\boldsymbol{r}_{I_k} \times \boldsymbol{b}_{I_k})\times] & -\boldsymbol{b}_{I_k} \\ \boldsymbol{b}_{I_k}^T & 0 \end{bmatrix} \tag{16}$$

and the intersection of the boresight and the reference plane is

$$\begin{bmatrix} \boldsymbol{x}_{I_k} \\ 1 \end{bmatrix} \propto \boldsymbol{L}_b \boldsymbol{\pi}_{I_k} \tag{17}$$

Finally, we may convert the relevant quantities to the LIDAR sensor frame

$$\boldsymbol{n}_{L_k} = \boldsymbol{T}_{L_k}^{I} \boldsymbol{n}_{I_k} \tag{18}$$

$$\boldsymbol{x}_{L_k} = \boldsymbol{T}_{L_k}^{I} \left(\boldsymbol{x}_{I_k} - \boldsymbol{r}_{I_k}\right) \tag{19}$$

Transforming LIDAR Observations to Elevation Observations

With the reference plane defined using one of the methods above, the next step is to transform the LIDAR observations into a frame attached to this reference plane. To perform this transformation requires a new reference frame be defined. We choose to place the origin of this reference frame at the location where the boresight pierces the plane. We also choose to define the frame's orthonormal basis by the unit vectors $\{\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}}\}$. These may be expressed in the frame L_k as

$$\hat{\boldsymbol{z}} \propto \boldsymbol{n}_{L_k} \quad \text{and} \quad \hat{\boldsymbol{y}} \propto \hat{\boldsymbol{z}} \times \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad \text{and} \quad \hat{\boldsymbol{x}} \propto \hat{\boldsymbol{y}} \times \hat{\boldsymbol{z}} \tag{20}$$

Thus, the transformation from the LIDAR frame to the terrain relative frame is given by

$$\boldsymbol{T}_{H_k}^{L_k} = \begin{bmatrix} \hat{\boldsymbol{x}}^T \\ \hat{\boldsymbol{y}}^T \\ \hat{\boldsymbol{z}}^T \end{bmatrix} \tag{21}$$

Finally, each of the measured LIDAR points $\{\boldsymbol{\ell}_{L_k}^{(i)}\}_{i=1}^{n_k}$ may be re-expressed in the terrain relative frame as

$$\boldsymbol{h}_{(H_k)k}^{(i)} = \boldsymbol{T}_{H_k}^{L_k}(\boldsymbol{\ell}_{(L_k)k}^{(i)} - \boldsymbol{x}_{L_k}) = \begin{bmatrix} h_{xk}^{(i)} \\ h_{yk}^{(i)} \\ h_{zk}^{(i)} \end{bmatrix} \tag{22}$$

Orthographic Projection onto a Pixel Grid

While it is possible to dynamically size the image, practical constraints on memory and speed make it desirable to simply pick an image size. Here we choose a square image of size $N \times N$, where N is a designed parameter selected by the algorithm developer ahead of time.

The mapping of the LIDAR scan points to their corresponding pixel coordinates is illustrated in Figure 2. The first step is to determine the scaling from units of length to units of pixels. This is achieved by first finding the extent LIDAR scan values,

$$\Delta x = \max\left[\{h_{xk}^{(i)}\}_{i=1}^{n_k}\right] - \min\left[\{h_{xk}^{(i)}\}_{i=1}^{n_k}\right] \tag{23}$$

$$\Delta y = \max\left[\{h_{yk}^{(i)}\}_{i=1}^{n_k}\right] - \min\left[\{h_{yk}^{(i)}\}_{i=1}^{n_k}\right] \tag{24}$$

Since we are constructing a square grid of $N \times N$ pixels, the conversion from units of length to pixels may be computed as

$$\alpha = N / \max[\Delta x, \Delta y] \tag{25}$$

Moreover, if we expect the $[0, 0]$ pixel to be at the hypothetical minimum values of the LIDAR scan, the image center lies at pixel coordinates

$$u_o = -\alpha \min\left[\{h_{xk}^{(i)}\}_{i=1}^{n_k}\right] \tag{26}$$

$$v_o = -\alpha \min\left[\{h_{yk}^{(i)}\}_{i=1}^{n_k}\right] \tag{27}$$

Hence, we may convert any LIDAR scan point to a pixel coordinate of

$$\begin{bmatrix} u^{(i)} \\ v^{(i)} \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & 0 & u_o \\ 0 & \alpha & v_o \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} h_{xk}^{(i)} \\ h_{yk}^{(i)} \\ 1 \end{bmatrix} \tag{28}$$

with a corresponding height of

$$h^{(i)} = \boldsymbol{k}^T \boldsymbol{h}_{H_k}^{(i)} = h_{zk}^{(i)} \tag{29}$$

where $\boldsymbol{k}$ is a unit vector along the $\hat{z}$ direction perpendicular to the reference plane. To compact notation, define the matrix $\boldsymbol{K}$ as

$$\boldsymbol{K} = \begin{bmatrix} \alpha & 0 & u_o \\ 0 & \alpha & v_o \\ 0 & 0 & 1 \end{bmatrix} \tag{30}$$

The points $[u^{(i)}, v^{(i)}]$ are not generally integers and do not evenly cover the square image space. Thus, some pixels will contain no LIDAR scan points, some will contain only one scan point, and some will contain many scan points. To construct an elevation image, $I(u, v)$, one may loop through all the measurements and accumulate the data while also creating a visibility mask as described in Algorithm 1. We may ignore the pixels in the image where $M(u, v) = 0$.

Feature Matching and Extracting the 3D Location

Assuming sufficient terrain relief is present, 2D feature descriptors (e.g. SURF,[16] SIFT,[17] ORB,[18] KAZE[19]) can be used to detect elevation map features and match them. These types of feature descriptor encode the structure of pixel intensities in the neighborhood about a keypoint in the image. The keypoint pixel coordinates for m corresponding features are given by $\{\boldsymbol{u}_k'^{(j)}\}_{j=1}^m$.

Mapping a 2D feature's keypoint location to a 3D position requires the surface elevation, $h_k'^{(j)}$, at that point which can be obtained from biliner interpolation of the elevation map. Bilinear interpolation requires that the keypoint is surrounded by four populated pixels (non-zero elements in the visibility mask). The subset of pixels which can be interpolated are found in the erosion of the visibility mask using an appropriate structuring element.[20] Simply put, a pixel is included

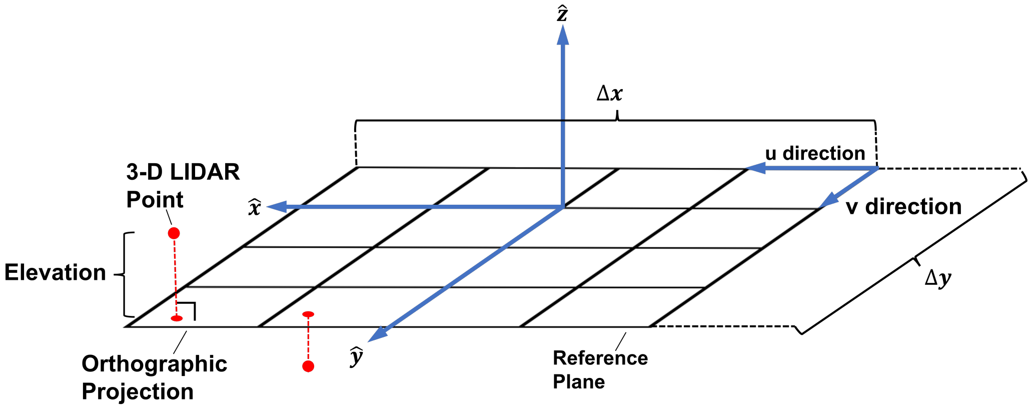

Figure 2. The orthographic projection mapping a LIDAR point onto a gridded reference plane.

Algorithm 1 Pseudocode to construct an elevation map

```
procedure $[I, M] = \textsc{GridScan}(\{\boldsymbol{h}_{H_k}^{(i)}\}_{i=1}^{n_k}, N)$
    compute $\boldsymbol{K}$                                          ▷ Eq. (30)
    allocate $M = \mathbf{0}_{N\times N}$
    allocate $I = \mathbf{0}_{N\times N}$
    for $i = 1$ to $n_k$ do
        compute $[u^{(i)}, v^{(i)}]$                                  ▷ Eq. (28)
        compute $u = \lfloor u^{(i)} \rfloor$ and $v = \lfloor v^{(i)} \rfloor$
        $I(u, v) = I(u, v) + h^{(i)}$
        $M(u, v) = M(u, v) + 1$
    for $i = 0$ to $N - 1$ do
        for $j = 0$ to $N - 1$ do
            if $M(i, j) > 0$ then
                $I(i, j) = I(i, j)/M(i, j)$
return $[I, M]$
```

in the erosion when the structuring element is centered on the pixel and all the other pixels in the structuring element correspond to non-zero mask pixels. A simple 3x3 square structuring element is sufficient to perform bilinear interpolation. In fact, the erosion will always underestimate the true valid region regardless of the size or shape of the structuring element selected. Corrections are not made to reclaim the missing region for two reasons. First, the size of the missing region is negligible compared to the remaining erosion. Second, the region of support used to generate the 2D feature descriptors is likely to be larger than the missing region. In practice it may be prudent to select a larger structuring element to intentionally reject keypoints near the edge. Now, the keypoints contained in the erosion can be interpolated are given by the set $\{\boldsymbol{u}_k'^{(q)}\}_{q=1}^{m}$.

For bilinear interpolation, we require the u-v components of the four pixels surrounding a given keypoint, $\boldsymbol{u}_k'^{(q)}$, are

$$u_k'^{(q)-} = \lfloor u_k'^{(q)} \rfloor \quad \text{and} \quad u_k'^{(q)+} = \lceil u_k'^{(q)} \rceil \quad \text{and} \quad v_k'^{(q)-} = \lfloor v_k'^{(q)} \rfloor \quad \text{and} \quad v_k'^{(q)+} = \lceil v_k'^{(q)} \rceil \tag{31}$$

which are used to look-up the elevations corresponding to each pixel in the image

$$h_{1,k}'^{(q)} = I(u_k'^{(q)-}, v_k'^{(q)-}) \tag{32}$$

$$h_{2,k}'^{(q)} = I(u_k'^{(q)-}, v_k'^{(q)+}) \tag{33}$$

$$h_{3,k}'^{(q)} = I(u_k'^{(q)+}, v_k'^{(q)-}) \tag{34}$$

$$h_{4,k}'^{(q)} = I(u_k'^{(q)+}, v_k'^{(q)+}) \tag{35}$$

The pixels in the image are unit width simplifying the bilinear interpolation to

$$h'^{(q)} = \begin{bmatrix} u_k'^{(q)+} - u_k'^{(q)} & u_k'^{(q)} - u_k'^{(q)-} \end{bmatrix} \begin{bmatrix} h_{1,k}'^{(q)} & h_{2,k}'^{(q)} \\ h_{3,k}'^{(q)} & h_{4,k}'^{(q)} \end{bmatrix} \begin{bmatrix} v_k'^{(q)+} - v_k'^{(q)} \\ v_k'^{(q)} - v_k'^{(q)-} \end{bmatrix} \tag{36}$$

which estimates the elevation. The matched keypoints can be mapped to the terrain relative frame using $\boldsymbol{K}^{-1}$

$$\begin{bmatrix} h'^{(q)}_{xk} \\ h'^{(q)}_{yk} \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\alpha} & 0 & \frac{-u_0}{\alpha} \\ 0, & \frac{1}{\alpha} & \frac{-v_0}{\alpha} \\ 0, & 0 & 1 \end{bmatrix} \begin{bmatrix} u'^{(q)}_{k} \\ v'^{(q)}_{k} \\ 1 \end{bmatrix} = \boldsymbol{K}^{-1} \begin{bmatrix} u'^{(q)}_{k} \\ v'^{(q)}_{k} \\ 1 \end{bmatrix} \tag{37}$$

and the feature location is constructed from the elevation map as

$$\boldsymbol{h}'^{(q)}_{H_k} = \begin{bmatrix} h'^{(q)}_{xk} \\ h'^{(q)}_{yk} \\ h'^{(q)} \end{bmatrix} \tag{38}$$

Finally, the feature location in the LIDAR frame is obtained by rearranging Eq. (22) as

$$\boldsymbol{\ell}'^{(q)}_{(L_k)k} = \boldsymbol{T}^{H_k}_{L_k} \boldsymbol{h}'^{(q)}_{H_k} + \boldsymbol{x}_{L_k} \tag{39}$$

which is the form required to compute the LIDAR translation in Eq. (7).

Robust Feature Matching Between Images

We can expect a subset of the correspondences to be mismatched during the feature matching process, and these incorrect correspondences must be removed to produce an accurate translation estimate.[21] Since the landmarks on the lunar surface are fixed with respect to the body, the true feature matches will have the same apparent motion. A simple test is to hypothesize a model for the correct feature translation and then determine which of the remaining feature matches fit the model. The pairs which fit the model are geometrically consistent and these inliers are added to the consensus sent for the given hypothesis. One hypothesis is generated by computing the translation for each pair of matched features

$$\boldsymbol{d}'^{(q)}_{(L_k)k} = \boldsymbol{\ell}'^{(q)}_{(L_k)k} - \boldsymbol{T}^{L_{k-1}}_{L_k} \boldsymbol{\ell}'^{(q)}_{(L_{k-1})k-1} \tag{40}$$

and it's added to the matched feature locations from the t_{k-1} point cloud. The result is compared to the expected matched feature location from the t_k point cloud. If the error, computed as the Euclidean distance, of the two points is within a tolerance then, the matched feature pair is accepted as an inlier for the given hypothesis. The hypothesis with the largest consensus set is selected, and the inliers for the consensus set are used to estimate the LIDAR translation using Eq. (7).

LIDAR ODOMETRY SIMULATION

The LO framework is evaluated using synthetic LIDAR scans generated along a reference lunar descent trajectory. Scans are simulated by ray casting from a an assumed sensor location onto a synthetic terrain mesh. The LO algorithm's performance is then quantified by estimating the LIDAR's translation from only the scan measurements.

LIDAR scans are generated from synthetic terrain because current digital terrain maps (DTMs), such as those from SELENE[22] or the Lunar Reconnaissance Oriter (LRO),[22,23] do not contain adequate resolution at low altitudes. The synthetic terrain in Figure 3 was created in Blender by Intuitive Machines to simulate scans for low altitude descent trajectories. A Python script generates a LIDAR scan with a triangle pattern and the point cloud is generated by casting each LIDAR ray onto the surface mesh. Neither motion blur from the spacecraft motion nor measurement errors are incorporated in the synthetic scans.

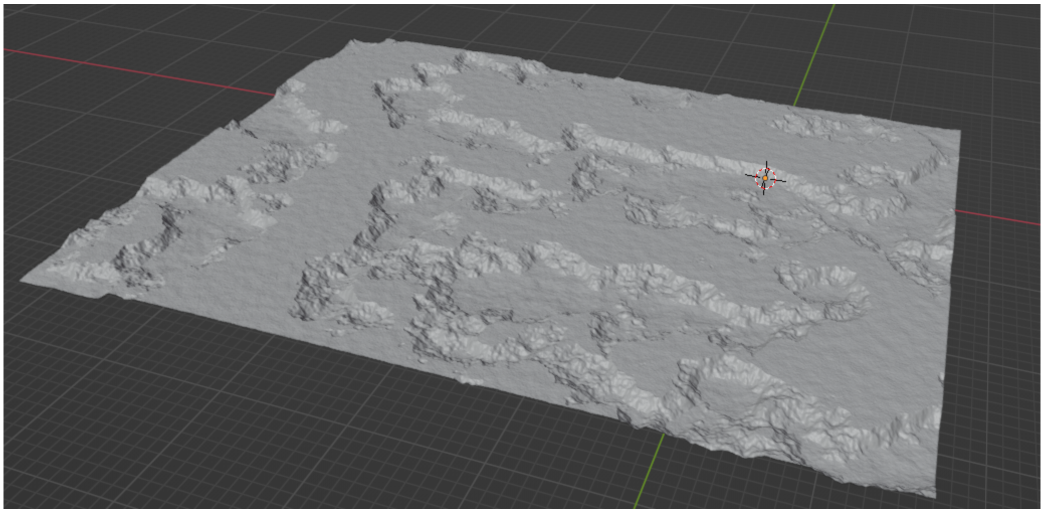

Figure 3. A synthetic lunar DEM was created to simulate LIDAR measurements within the Blender environment. The synthetic DEM shown here was created by Justin Westmoreland and Lillian Hong of Intuitive Machines.

LIDAR scans are generated along a reference trajectory for Intuitive Machines' IM-1 mission. While the IM-1 mission will not utilize LO techniques, LIDAR measurements are assumed to start near the same altitude as other sensors in the IM-1 hazard detection and avoidance (HDA) suite.[10] While processing the scans, the best fit plane is used to generate elevation maps with a grid resolution of 512x512 cells. The SURF feature descriptor is used to match features, and an error of 1 meter is used for a robust geometric constraint.

Three LO examples from the descent trajectory are shown in Figures 4, 5, and 6. Each image shows the elevation maps for a pair of scans with the robust matches overlaid on top. The spacecraft translation distance for each image pair is approximately 62 meters with the maximum altitude change of 7 meters. The first elevation map in each pair is shown on the left, and the second is shown on the right. Black pixels indicate a grid locations that do not contain any LIDAR scan measurements.

Each pair of elevation maps show robust feature matches tend to lie along regions with significant terrain relief. The number of robust matches for each image pair are tabulated in Table 1 along with the number of matched SURF features and interpolated features. The translation error is less than 1% for all three test cases.

In general, image pairs with more feature matches will also have a larger number of robust feature matches. Many of the SURF matches are adjacent to cells without any LIDAR data (black pixels) meaning the features can't be interpolated. The stripes of black pixels in the images are due to both the triangle scan pattern and the elevation map grid resolution. The elevation map resolution is the only algorithm parameter which can eliminate this effect so selecting an appropriate value to support the data is crucial. Note that the LIDAR ground sample distance (GSD) increases with altitude. Thus, if the grid resolution is fixed throughout flight, then the limiting case corresponds to

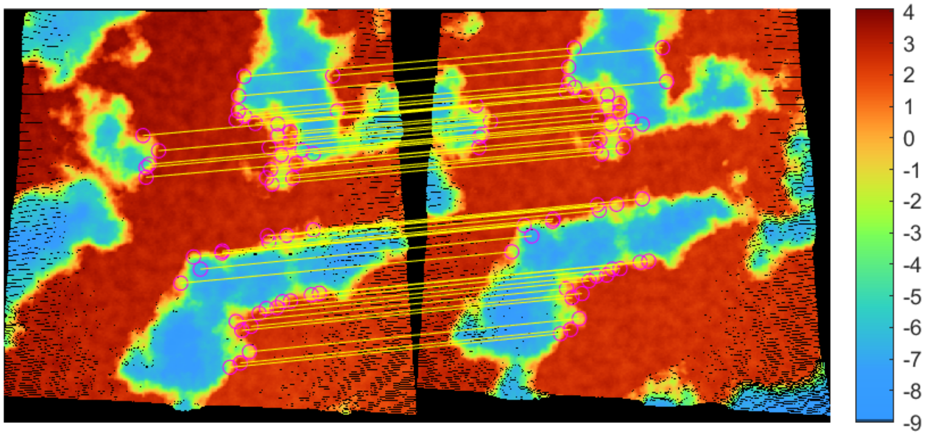

Figure 4. Robust feature matches overlayed on the elevation maps for the first test case.

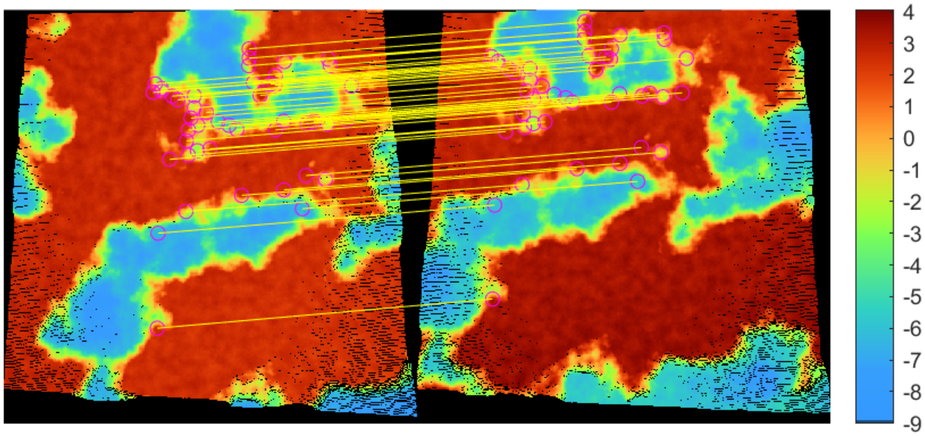

Figure 5. Robust feature matches overlayed on the elevation maps for the second test case.

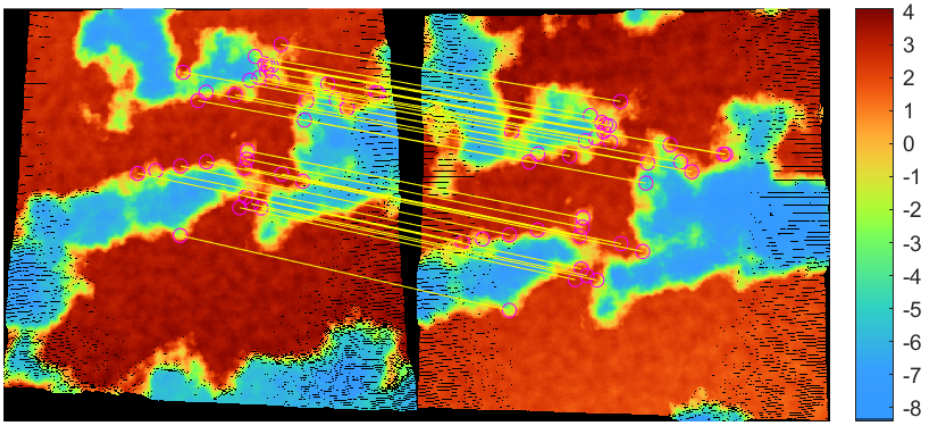

Figure 6. Robust feature matches overlayed on the elevation maps for the third test case.

the highest altitude at which LIDAR measurements are expected to be captured.

Table 1. Summary of feature correspondences and translation error for three test cases.

	Case 1	Case 2	Case 3
Matched Features	109	98	74
Interpolated Features	67	75	57
Robust Matches	48	46	34
Translation Percent Error (%)	0.15	0.09	0.23
True translation (m)	62.8	62.8	62.1

CONCLUSIONS

LIDAR odometry (LO) is well suited to provide terrain relative navigation (TRN) measurements for future missions which expect poor surface lighting conditions. A framework for LO was developed and it is general enough to work regardless of the LIDAR scan pattern. Sufficient scan overlap and significant terrain relief are required to detect opportunistic features using popular feature descriptors. The LO algorithm performance was demonstrated using three pairs of synthetic LIDAR scans along the descent segment of the IM-1 lunar reference trajectory. The examples provided indicate the framework is capable of estimating spacecraft translations for real-time applications.

REFERENCES

[1] A. E. Johnson and J. F. Montgomery, “Overview of Terrain Relative Navigation Approaches for Precise Lunar Landing,” *2008 IEEE Aerospace Conference*, 2008, pp. 1–10, 10.1109/AERO.2008.4526302.

[2] R. D. Olds and et al., “The Use of Digital Terrain Models for Natural Feature Tracking at Asteroid Bennu,” *The Planetary Science Journal*, Vol. 3, 2022, 10.3847/PSJ/ac5184.

[3] A. Johnson and et al., “Mars 2020 Lander Vision System Flight Performance,” *AIAA SciTech Forum*, No. AIAA 2022-1214, 2021, 10.2514/6.2022-1214.

[4] Y. Cheng, A. Ansar, and A. Johnson, “Making an Onboard Reference Map From MRO/CTX Imagery for Mars 2020 Lander Vision System,” *Earth and Space Science*, Vol. 8, No. 8, 2021, 10.1029/2020EA001560.

[5] J. Christian, H. Derksen, and R. Watkins, “Lunar Crater Identification in Digital Images,” *The Journal of the Astonautical Sciences*, Vol. 68, 2021, pp. 1056–1144, 10.1007/s40295-021-00287-8.

[6] D. Scaramuzza and F. Fraundorfer, “Visual Odometry [Tutorial],” *IEEE Robotics Automation Magazine*, Vol. 18, No. 4, 2011, pp. 80–92, 10.1109/MRA.2011.943233.

[7] Y. Cheng, J. Goguen, A. Johnson, C. Leger, L. Matthies, M. Martin, and R. Willson, “The Mars exploration rovers descent image motion estimation system,” *IEEE Intelligent Systems*, Vol. 19, No. 3, 2004, pp. 13–21, 10.1109/MIS.2004.18.

[8] D. S. Bayard, D. T. Conway, R. Brockers, J. Delaune, L. Matthies, H. F. Grip, G. Merewether, T. Brown, and A. M. San Martin, “Vision-Based Navigation for the NASA Mars Helicopter,” *AIAA SciTech Forum*, No. AIAA 2019-1411, 2019, 10.2514/6.2019-1411.

[9] J. A. Christian, L. Hong, P. McKee, R. Christensen, and T. P. Crain, “Image-Based Lunar Terrain Relative Navigation Without a Map: Measurements,” *Journal of Spacecraft and Rockets*, Vol. 58, No. 1, 2021, pp. 164–181, 10.2514/1.A34875.

[10] G. Molina, M. Hansen, J. Getchius, R. Christensen, J. A. Christian, S. Stewart, and T. Crain, “Visual Odometry for Precision Lunar Landing,” *AAS Guidance, Navigation, & Control Conference*, No. AAS 22-113, 2022.

[11] J. A. Christian and S. Cryan, “A Survey of LIDAR Technology and its Use in Spacecraft Relative Navigation,” *AIAA Guidance, Navigation, and Control (GNC) Conference*, 2013, 10.2514/6.2013-4641.

[12] S. Rusinkiewicz and M. Levoy, “Efficient variants of the ICP algorithm,” *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, 2001, pp. 145–152, 10.1109/IM.2001.924423.

[13] H. Yang, J. Shi, and L. Carlone, “TEASER: Fast and Certifiable Point Cloud Registration,” *IEEE Transactions on Robotics*, Vol. 37, No. 2, 2021, pp. 314–333, 10.1109/TRO.2020.3033695.

[14] A. Rhodes, J. Christian, and T. Evans, “A Concise Guide to Feature Histograms with Applications to LIDAR-Based Spacecraft Relative Navigation,” *Journal of Astronautical Sciences*, Vol. 64, 2017, pp. 414–445, 10.1007/s40295-016-0108-y.

[15] R. Spezialetti, S. Salti, L. Di Stefano, and F. Tombari, “3D Local Descriptors—from Handcrafted to Learned,” *3D Imaging, Analysis and Applications*, 2020, pp. 319–352, 10.1007/978-3-030-44070-1_7.

[16] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-Up Robust Features (SURF),” *Computer Vision and Image Understanding*, Vol. 110, No. 3, 2008, pp. 346–359. Similarity Matching in Computer Vision and Multimedia, https://doi.org/10.1016/j.cviu.2007.09.014.

[17] D. E. Lowe, “Distnictive Image Features from Scale-Invariant Keypoints,” *International Journal of Computer Vision*, Vol. 60, 2004, pp. 91–110, 10.2514/2.4988.

[18] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: An efficient alternative to SIFT or SURF,” *2011 International Conference on Computer Vision*, 2011, pp. 2564–2571, 10.1109/ICCV.2011.6126544.

[19] P. F. Alcantarilla, A. Bartoli, and A. J. Davison, “KAZE Features,” *Computer Vision – ECCV 2012* (A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, eds.), Berlin, Heidelberg, Springer Berlin Heidelberg, 2012, pp. 214–227.

[20] R. M. Haralick, S. R. Sternberg, and X. Zhuang, “Image Analysis Using Mathematical Morphology,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-9, No. 4, 1987, pp. 532–550, 10.1109/TPAMI.1987.4767941.

[21] F. Fraundorfer and D. Scaramuzza, “Visual Odometry : Part II: Matching, Robustness, Optimization, and Applications,” *IEEE Robotics Automation Magazine*, Vol. 19, No. 2, 2012, pp. 78–90, 10.1109/MRA.2012.2182810.

[22] M. Barker, E. Mazarico, G. Neumann, M. Zuber, J. Haruyama, and D. Smith, “A new lunar digital elevation model from the Lunar Orbiter Laser Altimeter and SELENE Terrain Camera,” *Icarus*, Vol. 273, 2016, pp. 346–355, https://doi.org/10.1016/j.icarus.2015.07.039.

[23] D. E. Smith, M. T. Zuber, G. A. Neumann, F. G. Lemoine, E. Mazarico, M. H. Torrence, J. F. McGarry, D. D. Rowlands, J. W. Head III, T. H. Duxbury, O. Aharonson, P. G. Lucey, M. S. Robinson, O. S. Barnouin, J. F. Cavanaugh, X. Sun, P. Liiva, D.-d. Mao, J. C. Smith, and A. E. Bartels, “Initial observations from the Lunar Orbiter Laser Altimeter (LOLA),” *Geophysical Research Letters*, Vol. 37, No. 18, 2010, https://doi.org/10.1029/2010GL043751.