MONOCULAR HORIZON NAVIGATION

Will Driessen[1], Siddarth Kaki[2], Andrew Liounis[3*], Daniel McGann[4], Paul McKee[5], Andrew Tennenbaum[6], and Alvin Yew[3]; [1]Space Exploration Analysis Laboratory at Georgia Institute of Technology, North Ave, Atlanta, GA 30332. [2]Jacobs Technology, Inc., NASA Johnson Space Center, 2101 E NASA Pkwy, Houston, TX 77058. [3]NASA Goddard Space Flight Center, 8800 Greenbelt Rd, Greenbelt, MD 20771. [4]Robotics Institute at Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213. [5]NASA Johnson Space Center, 2101 E NASA Pkwy, Houston, TX 77058. [6]Advanced Navigation and Controls Lab at the University of Buffalo, State University of New York, Amherst, NY 14260-4400. [*][andrew.j.liounis@nasa.gov]

Abstract. *The Artemis program advances towards operations on the lunar surface, where precise surface localization is a driving need for safety and science. Using the observable horizon as an image landmark allows for estimating the photographer's position. This work analyzes the application of Perspective-n-Point (PnP) algorithms to this lunar localization problem. Batch simulations using lunar topography display the accuracy and drawbacks of these methods. Monte Carlo techniques show the pipeline's solutions as measurements provided to a navigation filter. When filtered, these solutions have position errors of 50 meters or better, which is similar to or better than the performance of other methods of surface localization.*

Introduction. Precise localization on the surface of any solid body is critical to numerous aspects of surface operations. For science, precise localization enables geotagging of science observables and contextualizes result interpretation. For human operations, precise localization is required for crew safety, ensuring the crew is close enough to the base or a rover to return in an emergency, for efficient traverses, and for geotagging of science experiments and sample collections performed by the crew. In robotic science operations, localization is the first step in effective path planning, which is crucial for achieving mission goals and ensuring safe surface travel.

Radiometric tracking approaches for the Moon, such as provided under LunaNet navigation services,[1] similar to the Global Positioning System (GPS) on Earth, are promising to localize on the lunar surface. However, these capabilities require significant near and long term investments and complex operations with an associated phased deployment schedule. Additionally, depending on the size and design of the satellite constellation, the coverage provided by such a system can be limited in terms of solution availability. Without sufficient infrastructure, there is a pressing need for near-term, Size, Weight, Power, and Cost (SWaP-C) constrained capabilities during the development of these localization services.

This paper describes a potential, low SWaP-C approach for localization technology based on in-situ measurements of the observable horizon using only a monocular camera with a moderate field-of-view (FOV), a single board computer, and an Inertial Measurement Unit (IMU).

Related Work. Lunar surface navigation has been identified as an area that requires further research and development, especially with more and more crewed and uncrewed missions to the Moon scheduled as part of the Artemis program.[2] Navigation using the visible horizon of a planet or Moon has been demonstrated, as has navigation on the surface of another celestial body. Still, these have never been done in concert. Horizon-based Optical Navigation (OpNav)[3] flew successfully on the Artemis 1 mission,[4] demonstrating the ability to navigate autonomously in cis-lunar space using monocular images. Engineers at NASA's Jet Propulsion Laboratory (JPL) have been working on image-based navigation on the surface of Mars for decades.[5,6] Much of this work amounts to Hazard Detection and Avoidance (HDA) and navigation relative to a last known location.

Autonomous global navigation – estimating vehicle latitude and longitude using on-board measurements with no help from human operators – still needs to be demonstrated on another celestial body. When accomplished, it will almost certainly involve passive or active optical sensors such as cameras or Light Detection and Ranging (LiDAR). One technique that could perform autonomous global navigation is the Stellar Positioning System (SPS)[7–9] which is the modern equivalent of classical maritime celestial navigation (i.e., using a sextant). One can also use images or LiDAR scans of the local environment to detect and identify craters, as in the case of the LunarNav algorithm.[10–12] Crater-based navigation using orbital images can make use of global crater databases[13,14] and simplifying assumptions about problem geometry that lends to the use of invariant theory for crater identification.[15] By contrast, when performing crater-based navigation on the lunar surface, one must create a custom crater catalog for a particular mission, likely using orbital images or Digital Elevation Map (DEM) data. Further, one is reduced to using more heuristic methods of crater identification.

Another autonomous global navigation technique uses the visible skyline as a navigation observable. Panoramic images can be used to reconstruct a 360-degree skyline, which is (in theory) unique to an observer's position and invariant to an observer's heading direction. No analytical relationship allows for position to be deduced from skyline measurements. However, a look-up table consisting of many pre-rendered skylines can be used, and the closest match gives a coarse estimate of an observer's po-

sition. This has been studied in the context of lunar[16] and Martian[17] exploration.

Any image-based navigation technique will be sensitive to illumination conditions, and lighting at the lunar south pole (the area of greatest interest to the Artemis program) is highly unfavorable. Due to a phenomenon called the opposition effect, wherein near-zero phase angle, particulate media can increase in brightness and create illumination conditions that obscure crater rims at the lunar poles.[18] Inaccuracies in crater rim detection cascade into problems with crater identification, which ultimately lead to pose estimation errors. The same is true for issues measuring the distant skyline and matching it to a database, which will lead to localization errors.

Current Work. A method is presented for using the lunar horizon in a single monocular image for navigation relative to known maps of the lunar surface. This contrasts coarse position estimation techniques that require an entire panoramic skyline. One benefit of this is that an intelligent operator can choose a portion of the horizon to image, which is well-lit and provides good position observability – a mountain range in the distance provides far more information about the observer's position than a flat plain. Further, requiring only a single image means this analysis can be done with minimal change to mission operations. There is no need to plan and acquire enough images to reconstruct a panorama. Alternatively, observer localization can be done after a mission or surface operation is concluded. A single image, correlated with some activity (e.g., experiment, surface sample collection), is enough for an analyst to deduce the location of that activity after the fact.

Because the horizon tends to be a far-field observation, the required map resolution for comparisons is generally lower than that required for more near-field-focused approaches like crater navigation. The Lunar Reconnaissance Orbiter (LRO) Lunar Orbiter Laser Altimeter (LOLA) supplies topographical data that is generally already sufficiently high enough resolution for horizon-based navigation needs globally across the lunar surface.[19–21] These DEMs enable the generation of a known three-dimensional (3D) horizon at any position on the Moon.

Methodology. To solve the surface localization problem on the Moon with low SWaP-C hardware, we propose a small system of a moderate FOV ($20°$–$50°$) camera, a single board computer, and an IMU rigidly attached to the camera. Notably, much of this hardware is already included on robotic rovers and human space suits planned to explore the lunar surface, indicating that the techniques described in this paper could conceivably be implemented without additional hardware.

This system's concept-of-operations (ConOps) is described in Fig. 1. Begin by assuming we have some rough knowledge (on the order of 100 m, 1σ) of our location on the surface.* Then, a user (whether robotic or human) points the camera towards the lunar sky to capture a star field image. This star field image is processed using lost-in-space star identification algorithms (for examples see Refs. [27–29]) to determine the orientation of the camera at the time the image was captured. The user then slews the camera towards a visible portion of the lunar horizon, during which the angular rate information from the IMU is integrated to maintain knowledge of the camera's attitude. A well-exposed image of the horizon is captured. This image is processed to extract the observed horizon from the image, which is then matched with a predicted horizon from a DEM. This observed and predicted horizon pairing is used to solve a PnP problem for either the relative location and attitude of the camera with respect to the DEM reference frame or, more helpfully, just the relative location of the camera with respect to the DEM reference frame.

This position estimate is then fed to a navigation filter, such as an extended Kalman filter (EKF), as a measurement update. The EKF uses the IMU-reported acceleration to integrate the state until the next image is captured. We now discuss each of these steps in more detail.

Attitude Estimation. As previously mentioned, we rely on existing algorithms to solve for the precise pointing of the camera from a single image of a star field.[27–29] As shown in Ref. [27], for moderate FOV cameras with reasonable star centroiding performance, attitude knowledge from a single image can approach better than five arc seconds, 1σ. Additionally, reasonably sized, space-grade IMUs can have stable bias and random walk characteristics leading to minimal noise added through integration, on the order of 10-20 arc seconds assuming 10 seconds of integration time.[30] Therefore, it is expected that knowledge of the pointing of the camera for the horizon image (after star field capture/processing and camera slew) can be on the order of 20-30 arc seconds or about $0.006°$–$0.008°$. This accuracy will be necessary, as discussed later.

Horizon Extraction. After handling the star field image for attitude estimation, the user then captures a well-exposed image where the horizon is visible. This horizon image is then processed to extract the observed horizon points (Pts) in the image. Generally, the horizon forms a strong and easily distinguished feature in the image as a crisp gradient where the Moon's illuminated surface abruptly ends in the darkness of space.

Determination of the pixels that correspond to the horizon in the image proceeds is described in Refs. [31–34]. In summary, we begin by scanning in the illumination direction projected onto the image plane (which is usually nearly vertical in the image). We then find each scan's first high-intensity gradient from dark to light.† We then

*This assumption can be met with the same hardware using some "lost on the Moon" algorithms like those in Refs. [22–26].

†With real, noisy images, care needs to be taken that we

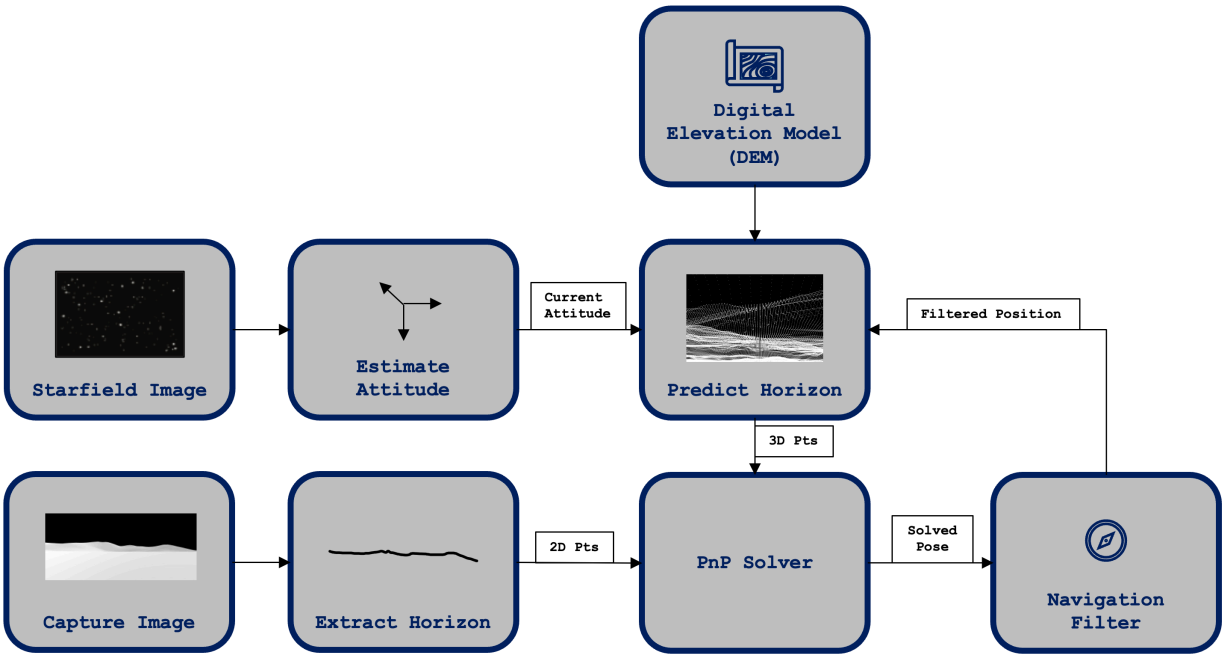

Figure 1. Horizon matching pipeline concept-of-operations.

use subpixel detection methods to find the subpixel location of the limb.[31,35] Finally, we correct the effects of distortion induced by the camera lens's optics so that the limb points represent a true gnomic projection from three dimensions to two dimensions. This gives us a set of observed, 2-dimensional (2D) horizon points in the image, typically with an accuracy of around 0.1 pixel, 1σ.

Horizon Prediction. Given that we have an *a priori* estimate of the location of the camera in the DEM reference frame[‡], along with the estimate of the pointing of the camera as discussed in the Attitude Estimation subsection, we have a full, six-degree-of-freedom (DOF) pose (position and orientation) estimate between the Moon fixed reference frame and the camera frame. We also have a set of 2D, gnomically projected points corresponding to the line of sight vectors in the image frame from the Horizon Extraction subsection. Additionally, LRO's lunar orbiting laser altimeter (LOLA)[19,20] has been used to produce extensive global DEMs of the lunar surface[20,21] which provide sufficient resolution for most surface-based far-field horizon observations. Therefore, combining the *a priori* pose knowledge of the camera, the observed horizon directions in the camera frame, and the DEM of the lunar surface where we expect the camera to be, we can predict what the observable horizon should be to the camera.

We determine the 3D DEM points corresponding to the horizon using the ray-tracing-based binary search algorithm described in Refs. [32,33]. In summary, this looks like:

1. For each observed horizon point, we define the bounds of a binary search.
   (a) The "upper" bound corresponds to a ray that we guarantee will not strike the lunar surface. This is determined by rotating the observed ray from the image away from the Moon's origin by some angle, which accounts for potential errors induced from the *a priori* pose knowledge.
   (b) The "lower" bound corresponds to a ray that we guarantee will strike the lunar surface. This is determined by rotating the observed ray from the image towards the Moon's origin by some angle, which accounts for potential errors induced from the *a priori* pose knowledge.
2. Define a ray with a direction halfway between the upper and lower bounds of the binary search.
3. Trace this ray through the DEM and determine whether it intersects the surface.
   (a) If the ray intersects the surface, define this ray as the new "lower" bound.
   (b) If the ray does not intersect the surface, define this ray as the new "upper" bound.
4. Return to step 2 and proceed with the new "lower" and "upper" bounds.

do not mistakenly detect a noise spike or even a star before finding the horizon.

[‡]Again, recall that this is a rough location estimate, on the order of hundreds of meters, 1σ

5. Break when requested relative and absolute tolerances have been met. The DEM intersection location by the final "lower" bound is interpreted as the 3D DEM horizon point.

Pose Estimation. Given a set of 2D image points, $\mathbf{p}_i = [\ u_i \quad v_i\ ]^T$, which correspond with 3D model points, $\mathbf{m}_i = [\ x_i \quad y_i \quad z_i\ ]^T$ we want to determine the relative pose which optimally minimizes the projective distance between these associated points. This is known as the PnP problem and is described mathematically as

$$\underset{(\mathbf{R},\mathbf{t})}{\arg\min} \sum_i d_{\text{proj}}(\mathbf{p}_i, \mathbf{m}_i). \tag{1}$$

wherein d_{proj} is the projective distance function between image point $\mathbf{p}_i$ and model point $\mathbf{m}_i$, $\mathbf{R}$ is the rotation matrix that aligns the axes of the model frame and the image frame, and $\mathbf{t}$ is the translation vector in the image frame which aligns the origin of the model frame and the image frame. We can define d_{proj} according to

$$d_{\text{proj}}(\mathbf{p}_i, \mathbf{m}_i) = \left\| \mathbf{p}_i - \mathbf{K}[\ \mathbf{R} \quad \mathbf{t}\ ] \begin{bmatrix} \mathbf{m}_i \\ 1 \end{bmatrix} \right\| \tag{2}$$

where $\| \bullet \|$ indicates the 2-norm of the vector and $\mathbf{K}$ is the usual gnomic intrinsic camera matrix.

$$\mathbf{K} = \begin{bmatrix} k_x & \alpha & c_x \\ 0 & k_y & c_y \end{bmatrix}$$

One method to solve the PnP problem is to directly optimize over this non-linear cost function using any standard non-linear optimization technique (Gradient Descent, Gauss-Newton, Levenberg-Marquardt, etc.). However, the optimization-based approach is a local method that requires an initial position and attitude estimate. This makes the direct non-linear optimization approach sensitive to initialization and can result in inconsistent solutions. Within the PnP problem, there is an additional geometric structure, which prior works have exploited to derive more accurate global methods that do not rely on an initial pose. An example and the modern "standard" method is the Efficient PnP (EPnP) algorithm.[36] However, the field has seen advancements in recent years, resulting in novel approaches like the SQPnP algorithm.[37] This algorithm is not only an efficient global solver for the PnP problem, but it also provides a guaranteed optimal solution.

We assume we are solving for all the 6-DOF of the pose in the previously mentioned algorithms. As will be discussed later, the 6-DOF PnP problem is prone to observability issues when most of the observations are nearly "coplanar" as with horizon points. The poor observability is due to high correlations between the pitch/yaw of the rotation matrix and vertical/lateral translation. However, as discussed in the Attitude Estimation subsection, we assume we have reasonably accurate knowledge of the camera pointing when taking our horizon image. Therefore, to use this strong *a priori* attitude knowledge, we also develop an alternative solution to the PnP problem using a translation-only direct linear transform (TDLT).[38] The details of this solver are described in Appendix: Direct Linear Transform for Translation-Only Solutions. It suffices to state that it solves for only the translation that best aligns the camera frame origin with the model frame origin, assuming that the camera frame and model frame axes have already been aligned through a known rotation.

We provide a brief, qualitative summary of the different PnP algorithms in Table 1.

Table 1. Properties of the three tested PnP algorithms.

	TDLT	**SQPnP**	**EPnP**
Relative Speed	Medium	Slow	Fast
Supports Weighting	Yes	Yes	No
Estimates	$\mathbf{t}$	$\mathbf{R}, \mathbf{t}$	$\mathbf{R}, \mathbf{t}$

Data Association. At this point in our solution, we have an *a priori* 6-DOF pose estimate (utilizing the *a priori* position estimate and the attitude estimate from the Attitude Estimation) subsection, 3D predicted horizon points from the Horizon Prediction subsection, and 2D observed gnomic image locations from the Horizon Extraction subsection. It is easy to recognize we now have a variant of the general model-to-image registration problem. With an image (taken by the robot or astronaut) and a model (lunar DEM from an orbiter mission), we attempt to "register" them (compute the relative pose between the camera and the model). This general problem has been extensively studied in computer vision and robotics, and by taking this broad perspective for horizon-based localization, we can take advantage of the advancements in these fields over the past decades.

Our model features are comprised of 3D points of the form $\mathbf{m} = [\ x \quad y \quad z\ ]^T$. Our image features are comprised of 2D-pixel points of the form $\mathbf{p} = [\ u \quad v\ ]^T$. With these features, the next step in the Model-to-Image registration pipeline is to associate data between the image features and model features.

In this case, we solve data association using our *a priori* pose knowledge. With this prior pose, we can project the estimated 3D model points into the image and associate points using their projective distances.

The most common method to perform this association is the Iterative Closest Point (ICP) algorithm. This form of data association requires iteratively re-solving the relative pose, as the name suggests. At each iteration, we associate each image point with its closest model point according to their projective distance and re-solve, assuming these associations are correct. A downside of ICP is that it is highly sensitive to initial conditions and is prone to converging to local optima.

To overcome ICP's sensitivity to initial conditions, we propose using Graduated Non-Convexity (GNC).[39] Like

ICP, GNC is an iterative method that requires re-solving the relative pose. GNC is an example of a "continuation" method. Continuation methods are a broad class of optimization techniques that attempt to solve problems with traditionally difficult cost landscapes (e.g., nonlinear, non-convex). The key idea of GNC is that we first solve a "convexified" version of our problem and use the solution to initialize the solving of progressively more non-convex versions of the problem until we recover the original. While GNC cannot provide any formal guarantees on its ability to improve convergence, prior work has found substantiative empirical evidence that GNC broadens the basin of convergence of these difficult optimization problems and allows the recovery of accurate solutions in otherwise intractable optimization tasks.

We adapt GNC to perform data association as follows. Let us define a matrix $\mathbf{A} \in \mathbb{R}^{N \times M}$ where N is the number of horizon points detected in the image and M is the number of horizon points estimated from the model. Let $\mathbf{A}_{ij}$ be proportional to the probability that image point $\mathbf{p}_i$ correctly associates to model point $\mathbf{m}_j$. Assuming we have $\mathbf{A}$ we can solve a weighted version of the PnP problem:

$$\underset{(\mathbf{R},\mathbf{t})}{\arg\min} \sum_{i,j} \mathbf{A}_{ij} d_{\text{proj}}(\mathbf{p}_i, \mathbf{m}_j). \tag{3}$$

When all elements of $\mathbf{A}$ are identical, we have convexified the problem by removing the non-convexity induced by data association. On the contrary, if we constrain each row/column to, at most, a single non-zero value, we recover the highly non-convex original problem. Further, we can define a procedure for smoothly transitioning from the convex to the non-convex case. We first compute a weight matrix for all possible data associations according to the weighting scheme, which has been derived from the SoftPOSIT algorithm:[40]

$$\mathbf{W}_{ij} = \gamma \exp -\mu\Big(d_{\text{proj}}(\mathbf{p}_i, \mathbf{m}_j)^2 - \alpha\Big) \tag{4}$$

where $\mathbf{W}$ is the weight matrix, $\gamma = 1/\big(\max(N, M) + 1\big)$ is a scale factor based on the number of points, $\alpha = 9.21\sigma_{\text{px}}^2 + 1$ defines the width of the weighting kernel based on the noise model of detected horizon points (σ_{px}^2), and μ is the GNC "control" parameter that transitions from the convex case ($\mu = 0$) to the non-convex case ($\mu = 1$). An example of how the weights are computed for different values of μ can be found in Fig. 2.

Once we have computed the weight matrix $\mathbf{W}$, we use it to derive the association matrix A. We do so by applying Sinkhorn's method.[41] This method transforms the weights into pseudo-probabilities, enforces a soft one-to-one assignment constraint, and enables us to handle outlier points (either image or model points that do not associate). We first augment the weight matrix with an additional row and column initialized to γ. When these "slack" elements converge to a large probability, it indicates no good association between the two points. We

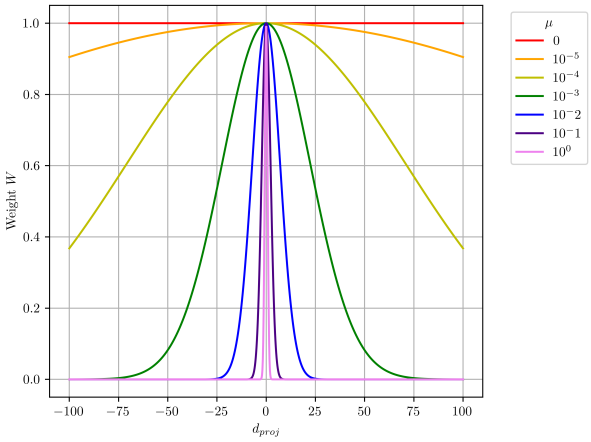

Figure 2. Example weights by projective distance for varying control parameter μ.

then alternate between normalizing the non-slack rows and non-slack columns. We do so until there is little change in the matrix's Frobenius norm. For additional discussion regarding the intuition behind how this method enforces the soft one-to-one assignment, refer to the SoftPOSIT paper.[40] We can now summarize the GNC method for data association as follows.

With feature points associated, we can compute the relative pose, $X \in SE(3)$, by solving the PnP problem for which many algorithms have been proposed. Another term used in literature for PnP problems with unknown data association is the blind-PnP problem, which does not necessarily assume that the 3D points are derived from some model.

Implementation Details. In the previous subsections, the steps of the horizon-based localization algorithm are shown from a theoretical perspective. However, several practical implementations are worthy of discussion. Algorithm 1 shows the PnP iteration when using GNC. Two criteria are implemented for the algorithm to accept an update step to avoid allowing degenerate configurations to result in algorithm divergence. If a step is rejected, the estimated pose is not changed, but the iteration of μ continues.

The first acceptance check is based on a trust region. The user supplies a trust region to the algorithm with a covariance on the 6-DOF pose and a χ^2 threshold. At each iteration of the algorithm, the difference between the new and previous estimated pose is evaluated by computing the Mahalanobis distance with the supplied covariance. This distance is compared to the critical value defined by the χ^2 threshold. For large enough updates, this step is rejected to avoid entering a degenerate configuration.

The second acceptance check is based on minimizing the projective distance. After computing the new pose estimate, the weighted projective distance is re-evaluated using the new pose but the exact correspondence. The

updated pose should result in a better solution assuming a fixed weighting. The pipeline rejects the pose solution when the projection from the estimated pose does not decrease the error between the estimated and actual horizon points.

Algorithm 1 Graduate Non-Convexity PnP

```
Initialize μ
Initialize pose (R, t) from a priori estimate
while μ < μ_final do
    Project the model points onto the image using the current pose estimate
    Compute the weight matrix W(μ) using Eq. 4
    Compute the correspondence matrix A using Sinkhorn's method[41]
    Solve for then new pose using weighted PnP using any weighted method from the Pose Estimation subsection
    if Covariance and Projection checks passed then
        Save estimated pose as updated pose
    end if
    μ ← μ + Δμ
end while
return Estimated pose (R, t)
```

Typically, the first iteration of the GNC algorithm should weigh all associations equally. However, practically doing so often causes the algorithm to diverge and reject the fact that we have some *a priori* knowledge of the pose. Minimizing the projective distances under the equal weight configuration is often achieved by moving the camera so all 3D points project into the center of the image, which is usually far from the true pose solution. It is nearly impossible to find a suitable solution from configurations like this. We avoid this scenario by initializing μ to a non-zero value. We have found that values of 10^{-3} or 10^{-4} work best from initial testing. Theoretically, we should also continue to iterate until $\mu = 1$, when we have recovered the fully non-convex problem. However, we have found that beyond a value of $\mu = 0.5$, the algorithm makes little additional progress. Therefore, we usually set the algorithm to terminate after μ surpasses 0.5.

Results. Multiple tests were run to investigate the performance of the pipeline. First, a single pipeline run gives an example of a point solution. Next, for the three PnP algorithms described (EPnP, TDLT, and SQPnP), many runs are shown to give an idea of the error properties of the estimation. Without loss of generality, in these tests, perturbations are performed in the true camera frame x and z-directions with conventions shown in Fig. 3.

Single Point Solution with Image. We first demonstrate the performance of the entire pipeline with image processing and DEM-based ray tracing. This test considers a pre-rendered synthetic image of the lunar horizon based on a lunar DEM at some known pose. This image is ingested, and the 2D horizon points in the image are extracted. Next, this known pose is perturbed in the position component but not the attitude component (representing having a rough *a priori* position estimate but an accurate *a priori* attitude estimate from a star tracker). The observer is then placed in the DEM at this perturbed pose representing the *a priori* pose estimate. The observed 3D DEM points corresponding to the visible horizon at this *a priori* pose are extracted via the techniques described in the Horizon Prediction subsection. When extracting the 3D horizon points from the DEM, the horizontal camera FOV is increased by a factor of 1.25 (compared to the actual camera FOV) for increased robustness. The 2D-3D correspondence problem is solved with GNC as described in the Data Association subsection, and the final pose estimate is solved with SQPnP. The pose solution is then used to bootstrap a subsequent pipeline run wherein the solved-for pose is employed as the new *a priori* pose estimate.

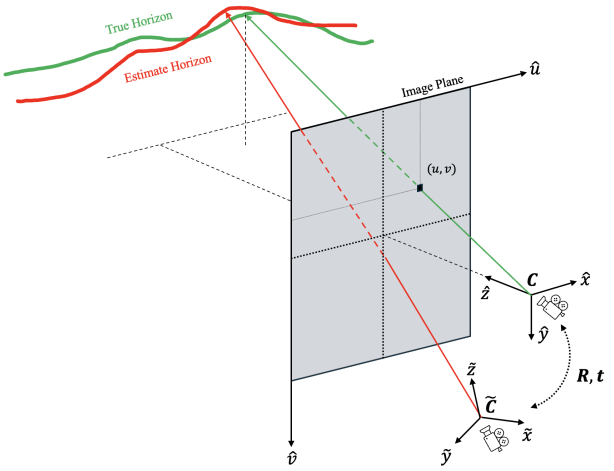

Figure 3. PnP pose estimation. Frame C is the definition of the canonical camera frame. Frame $\tilde{\mathbf{C}}$ represents the a priori camera frame.

In the presented example, 20 such iterations of the entire pipeline are conducted with the same 2D image input (but updated 3D DEM points per iteration based on updated *a priori* pose estimates). The *a priori* position error before the first run is approximately 400 m, and the final position error after the last run is approximately 73 m. Fig. 4 depicts the ingested image of the horizon superimposed by the extracted 2D horizon points in red (denoted "IMG PTS") and the 3D horizon points observed at the *a priori* pose in the DEM projected to the image plane in blue (denoted "DEM PTS") for the first iteration of the pipeline. While much of the red and blue lines overlap, a noticeable divergence is visible in the left part of the image corresponding to the mountain slope. The blue line extends on either side past the image due to the 1.25x horizontal FOV increase mentioned earlier. Fig. 5 depicts the same image with the 2D image points and 3D DEM

points projected to the image plane superimposed for the twentieth iteration of the pipeline. Here, the red and blue lines overlap almost perfectly, indicating convergence of the final pose solution closer to the true pose.

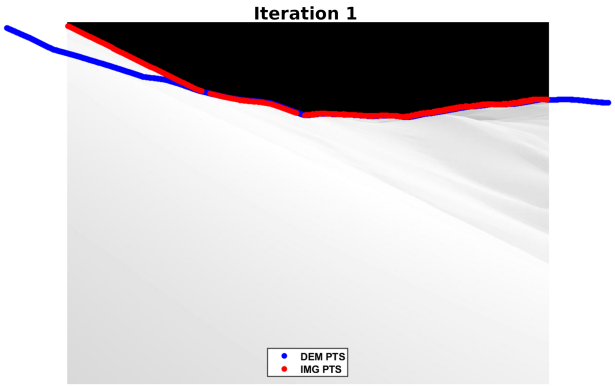

Figure 4. Synthetic image with 2D image points and projected 3D DEM points for the 1st iteration

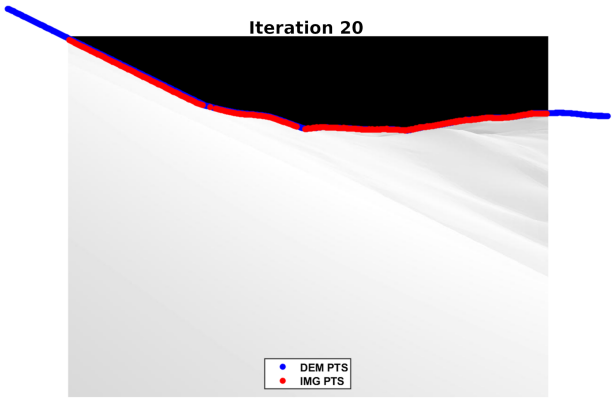

Figure 5. Synthetic image with 2D image points and projected 3D DEM points for the 20th iteration

Single Point Solution without Image. Having shown that the full pipeline works, including image processing, we can now devise a test that skips the expensive step of rendering simulated images. We begin this section by describing a single instance of this test to provide details. The test can then be repeated across a wide range of conditions and perturbations to gain insight into the algorithm's performance.

First, we place an observer with a true pose on the lunar surface. We then use the binary search algorithm described in the Horizon Prediction subsection (using the true pose) to determine the observed image points. The horizon profile of the observed image points is shown in Fig. 6 in solid black. Next, we perturb the true position in the true camera frame to form the *a priori* estimate of the pose. Note that we do not perturb the orientation as we assume this is known sufficiently, as discussed in the Attitude Estimation subsection. This results in the

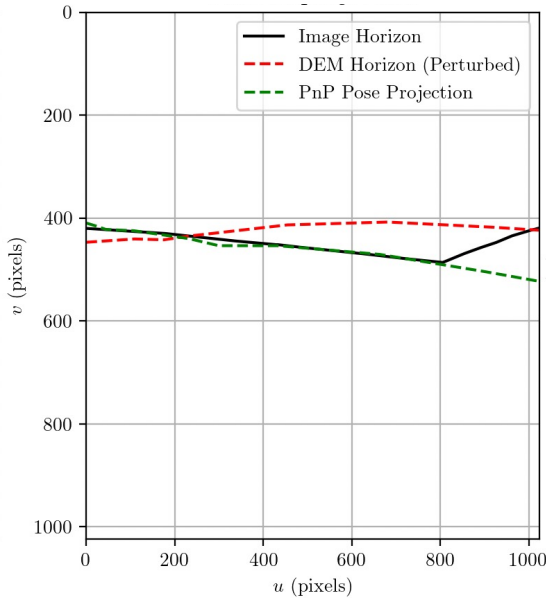

Figure 6. Image horizon, a priori horizon, and projected solved horizon corresponding to positions in Fig. 7.

dashed red line in Fig. 6. Finally, we provide these inputs to the GNC/PnP algorithms described in the Pose Estimation and Data Association subsections, resulting in a new pose estimate which produces the new projection shown as the green dashed line in Fig. 6. As seen in Fig. 6, the reprojected horizon more closely matches the true horizon, but not exactly, due in part to the correlation between the rotation estimates and the translation estimates. As shown in Fig. 7, the solved pose is closer to the true pose.

Batch Analysis Results. Utilizing the previous steps enables us to quickly test over different lunar positions and perturbations to understand the convergence statistics of the pose solutions. A set of locations between $25^\circ - 26^\circ$ latitude and $30^\circ - 31^\circ$ longitude on the lunar surface were chosen. Then, these locations were each perturbed away from the actual position to represent errors in the *a priori* pose. The horizon locations were extracted from a section of the DEM with a resolution of 256 pixels per degree, equivalent to $\tilde{1}18$ m in latitude.[42]

Figure 8 shows an example of the EPnP position estimation errors for 2500 pipeline runs, displaying the vast differences in convergence properties based on location. Also, errors are significant in all DOFs with only a perturbation in the x-direction. These errors represent a flaw in the geometry of the PnP problem itself; there is inherent unobservability between translation and rotation when all the observed correspondences are nearly co-planar, as for horizon measurements. This fact led to the development of the translation-only PnP solution TDLT, where the rotation is assumed to be well-known and thus does not obscure the translation solutions.

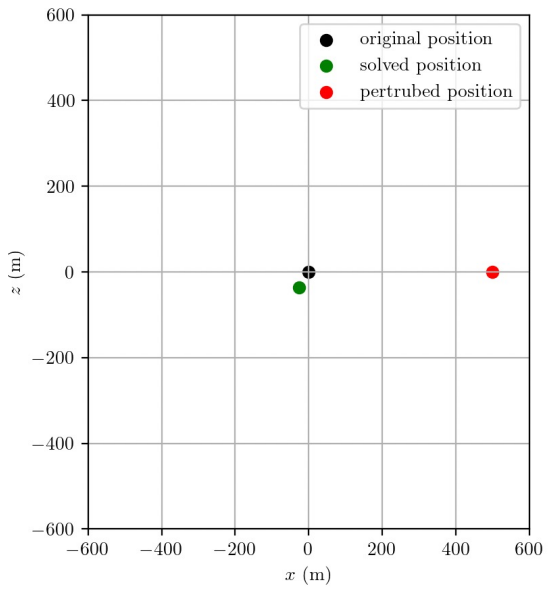

Figure 7. Solved PnP position as compared to the true (original) and a priori (perturbed) positions.

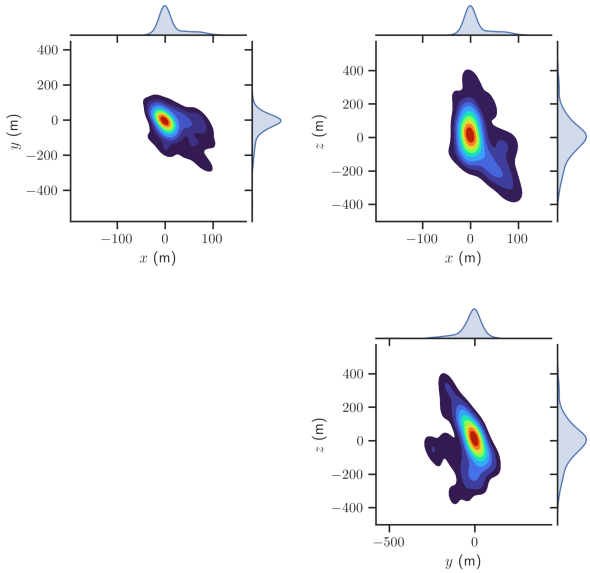

Figure 9. Coupling of estimation errors of a batch of SQPnP over a ± 500(m) perturbation in the x-direction.

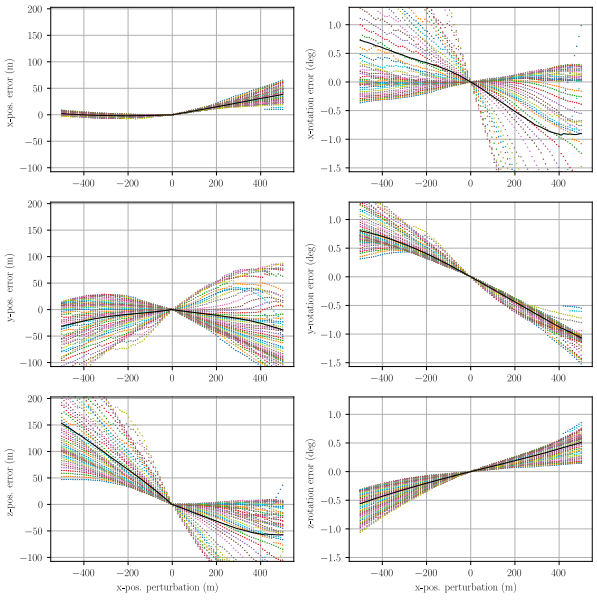

Figure 8. Position and rotation estimation errors for EPnP x-direction perturbation. The black line represents the average of the runs. Positions (m) are defined in the camera frame, and rotations are defined as rotations (deg) around the estimated camera frame axes.

The contour plots in Fig. 10 show the error distributions for all translation directions in the camera frame. The error along the z-direction, along the camera boresight, is the largest, consistent with the knowledge that range information is the least observable from single, monocular images. The odd tails that branch out from the main convergence area are considered failed convergences that still pass the checks implemented in the EPnP algorithm.

Figure 9 shows an example of the SQPnP position estimation errors for the same 2500 pipeline runs used with the EPnP solutions. These error contours are less tailed and more elliptically distributed around zero. The SQPnP problem solves a quadratic optimization problem, and sometimes, it can fail to converge to a true globally optimal solution, so the elongated blobs are thought to be failed pose estimates converged to another local optimum. The errors are within a similar order of magnitude to those of EPnP.

Figure 11 shows an example of the TDLT position estimation errors for the same 2500 pipeline runs demonstrated with EPnP. Recall that the relative orientation between actual and estimated frames is assumed to be known for this algorithm. Convergence is an order of magnitude less than the EPnP and SQPnP. This is due to the removed estimation of the orientation, where, in other PnP algorithms, errors in the estimated rotation seed errors in translation, and vice versa. Ultimately, the TDLT is a promising solution if an accurate *a priori* orientation can be integrated from a star field image, as discussed in the Attitude Estimation subsection.

From varying locations and perturbations shown for

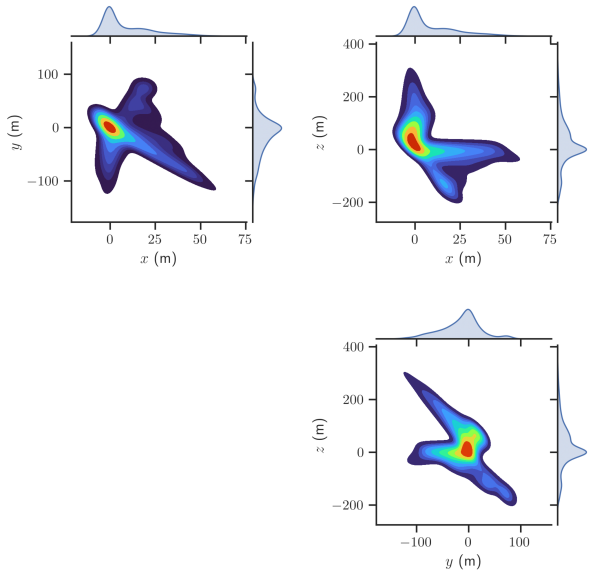

Figure 10. Coupling of estimation errors of a batch of EPnP over a ± 500(m) perturbation in the x-direction.

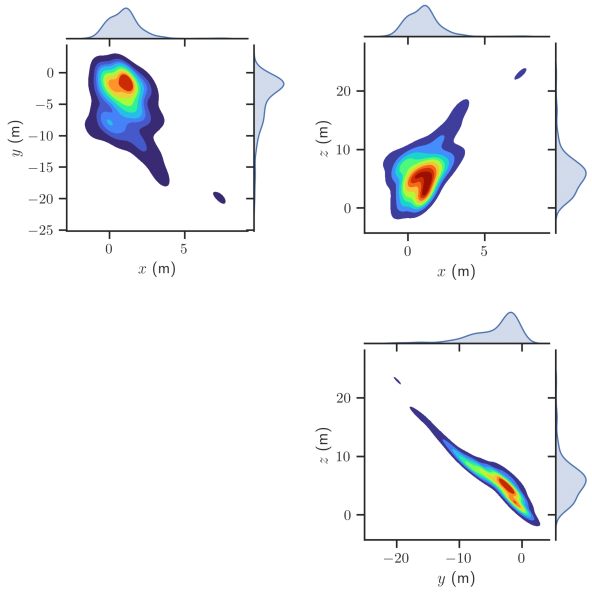

Figure 11. Coupling of estimation errors of a batch of TDLT over a ± 500(m) perturbation in the x-direction.

the EPnP algorithm in Fig. 8, statistics for the correlation between estimation errors were generated. The EPnP results were used because of their efficiency in a real-time hardware application. The overlayed heatmap in Fig. 12 shows the highly correlated behaviors between significant rotation and translation errors. The upper triangular elements are correlation coefficients, and the main diagonal represents the translation and rotation estimation error standard deviations. Terms are correlated based on physical unobservability in the camera frame. For example, the x-translation and the y-rotation have a high negative correlation. This represents the unobservability of yawing versus translating side to side in the camera frame. A similar unobservability occurs with the y-translation and the x-rotation, where the solution cannot differ between pitching the camera and moving the camera up and down. It is significant to see that the standard deviations for the position estimations are below 100 m and around 1° for orientation.

In comparison, we generated the same heatmap but for the TDLT dataset in Fig. 13. In this case, since we do not estimate rotation, the covariance/correlation coefficients are only 3×3. Notably, the standard deviations of the position components are significantly smaller (by an order of magnitude), which aligns with an intuitive understanding that by limiting the parameters we estimate, the estimator can put more of the information available into these parameters. This, in combination with the high correlation in the fully coupled 6-DOF pose estimates, demonstrates the benefits of having an accurate, independent attitude estimate for the horizon image as discussed in the Attitude Estimation subsection.

Algorithm Timing. We summarize the timing results from the three PnP solution methods in Tab. 2.[§] TDLT and EPnP run the pipeline the fastest. SQPnP fails to converge, but when it does, it produces more accurate results than EPnP. This gives insight into which algorithm to choose for real-time operation. Trials were performed on an M3 Macbook Pro with 16GB RAM, with portions of the algorithms written in Python. These numbers are meant for relative comparison only. Based on previous experience, the authors expect that these runtimes will still be reasonable (though slower) on available embedded hardware for space applications.

Navigation Filter Results. Though the point solutions are of a similar order of magnitude as other localization methods, the true power of this pipeline is that it can inform a navigation filter to predict position in real time. The advantage of having a navigation filter running in parallel is that it will use a well-studied method for fusing measurements from different epochs and even different

[§]Note that these run times are for the PnP solution only. They do not account for the image processing. From other testing, extraction of limb points from monocular images can be accomplished in a few seconds on representative space-grade processors even without field programmable gate array (FPGA) acceleration.

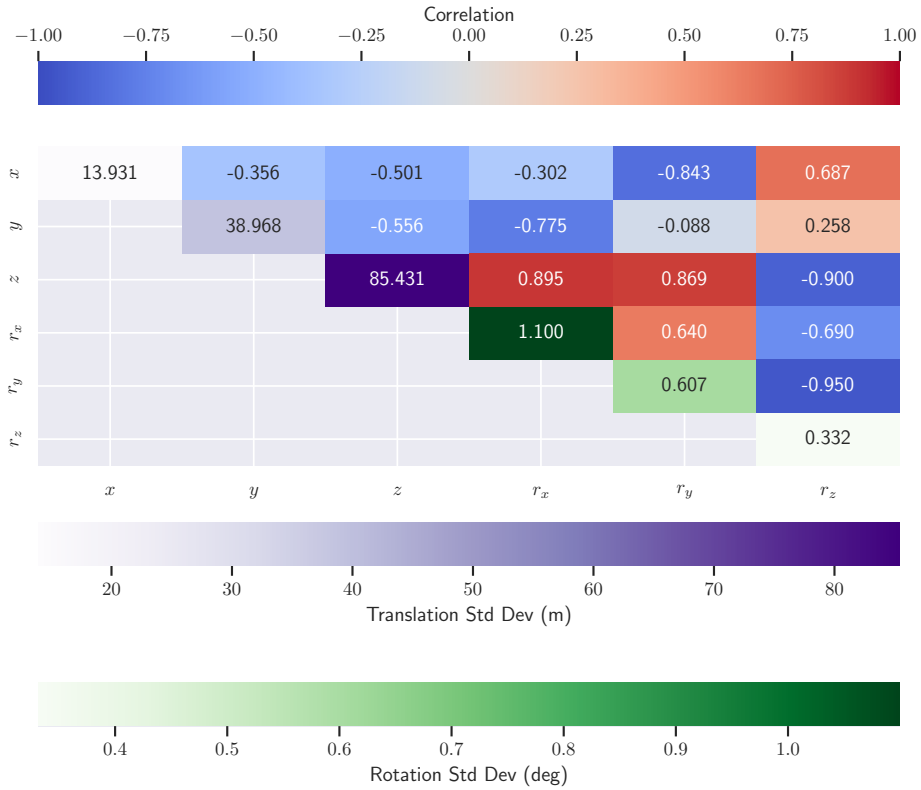

Figure 12. Overlayed heatmap to represent a covariance matrix for the 6-DOF EPnP estimation.

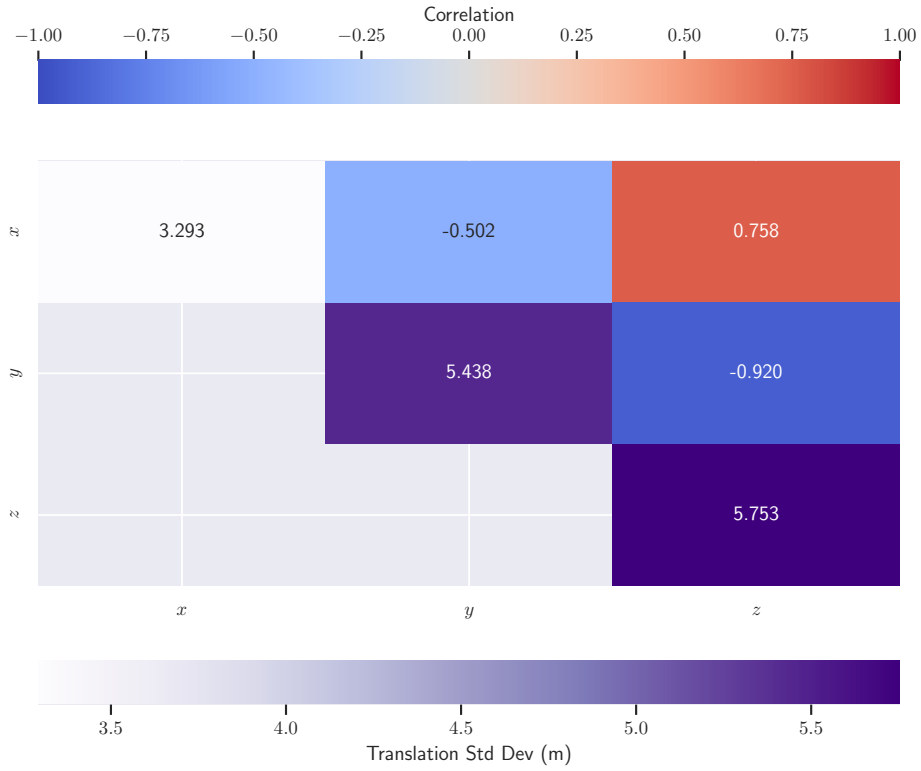

Figure 13. Overlayed heatmap to represent a covariance matrix for 3-DOF TDLT the estimation.

Table 2. Mean and standard deviation runtime for 100 runs of the horizon processing pipeline (excluding image processing).

	EPnP	**SQPnP**	**TDLT**
Mean (s)	0.667	8.11	1.04
Std. Dev. (s)	0.105	0.875	0.0477
Failed Runs	0%	30%	0%

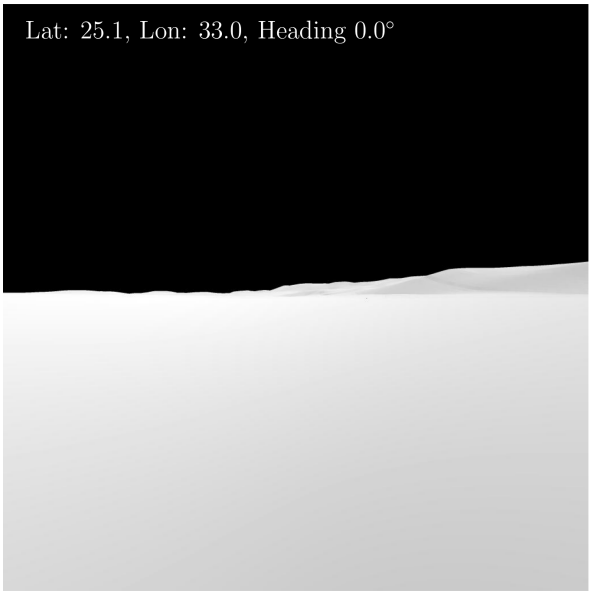

Figure 14. Image output from the GIANT ray-tracer.

instruments. Therefore, the overall errors can be reduced significantly using a sequential estimation framework.

A simplistic EKF formulation can be derived if Gaussian errors are assumed.[¶] The position standard deviation values, from the diagonal of Fig. 12, represent the measurement covariance matrix $\mathbf{R}$.[‖] A zero-mean, normally distributed position and velocity state are generated using the same standard deviation values to initialize a random state.

At the EKF update step, the current estimate of the position is added to the position output of the pipeline. This lets the current estimate of the position render the topographic points, and the images are generated along the truth trajectory to give the image horizon points. Fig. 14 shows an example of a simulated image.

[¶]We note that the error statistics discussed in the Batch Analysis Results subsection are obviously non-Gaussian. This subsection represents preliminary investigation of fusing multiple of these measurements together in time, therefore, for the time being the authors believe the Gaussian assumption to be reasonable.

[‖]Much better performance could be had using the TDLT solutions; however, given the non-Gaussian structure of the observables we chose the more poorly performing case for this analysis.

Two dynamical approaches were taken to validate the filter. First, a constant velocity approach of a significant landmark is investigated; this application is akin to an autonomous or Earth-controlled rover driving toward a desired target. The second method is an in-place 360° rotation, which applies to an astronaut taking multiple pictures of the horizon around them. For the first approach, an Extended Kalman Filter (EKF) with constant velocity dynamics propagates a surface transversal toward an object of interest. Dynamics are simulated with a state vector of $\mathbf{x} = (x, y, z, \dot{x}, \dot{y}, \dot{z})^T$.[**] The measurements are the position updates in the Moon Centered Moon Fixed Frame (MCMF) that are output after image processing of ray-traced images generated for the trajectory. Figures 15-16 show the post-update position residuals for position estimation in the camera frame for constant translation dynamics and constant rotation respectively.

The error profile follows the standard funnel shape for idealized EKF residuals, showing the decreasing estimation error as time progresses. A significant structural difference can be seen from the thicker funnels in Fig. 15 compared to the thinner funnels in Fig. 16. This represents the accelerated decrease in estimation error when the measurements are obtained from multiple panning, or yaw, angles. This could also result from not moving the observer's position while taking images.

These residuals show the 3σ bounds, where most errors should lie. Using Monte Carlo techniques, 1000 runs were performed to analyze the large-scale statistics. The covariance bounds are calculated analytically within the EKF, and the statistics of every run in the Monte Carlo are used. The agreement between the 3σ Monte Carlo error bounds and the analytical filter error bounds is promising. There is a slight difference that could be due to not enough runs representing the population or the pipeline's non-Gaussian error profile. There are many potential improvements to this filter, but this is a preliminary implementation to show the promise of estimating errors over time with a sequential filter.

Potential Hardware Design. The hardware requirements of a camera and processing card to implement this algorithm on a robotic rover or spacesuit are already standard on most space missions. Digital body cameras are used on Extravehicular Activities (EVAs) to ensure astronauts' safe maneuvers and document their activities at all times. Since a lunar camera will likely be designed for manual and stand-alone operation, it is unlikely (though possible) that any of these cameras could be used for localization. Additionally, the available computational resources on general-purpose processors may be insufficient for the numerous algorithms described in this paper. Therefore, for the purposes of this paper, we consider a stand-alone unit. This section will explore options for this system's processor board, camera, IMU, and

[**]At this point, we do not assume we are integrating accelerometer data and that the user is moving with constant velocity.

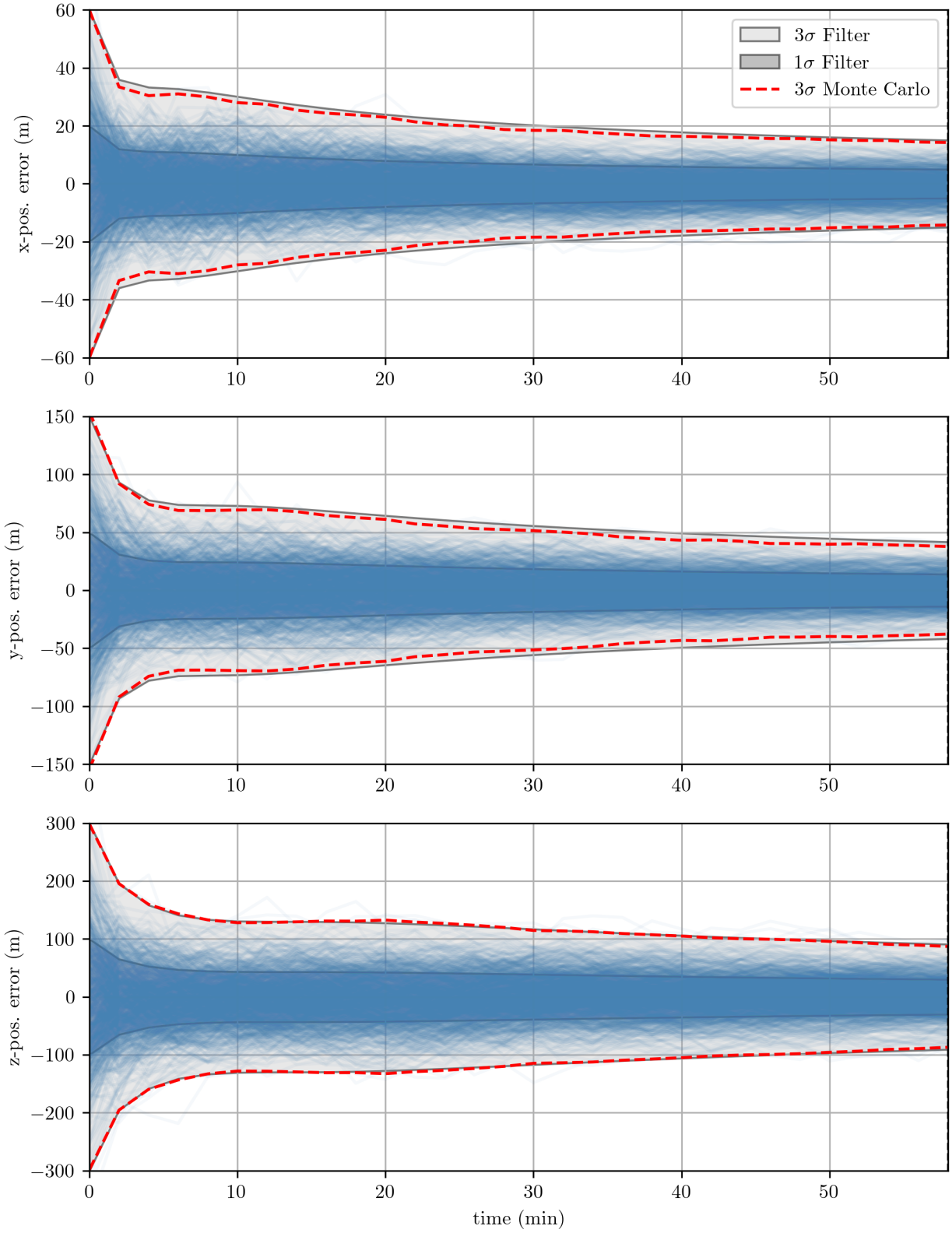

Figure 15. EKF x, y, z post-update position estimation residuals over one minute with observations every 2 seconds. 1000 runs of the filter are shown in blue lines contained within the analytic 1-3σ bounds. The measurements come from constant velocity dynamics.

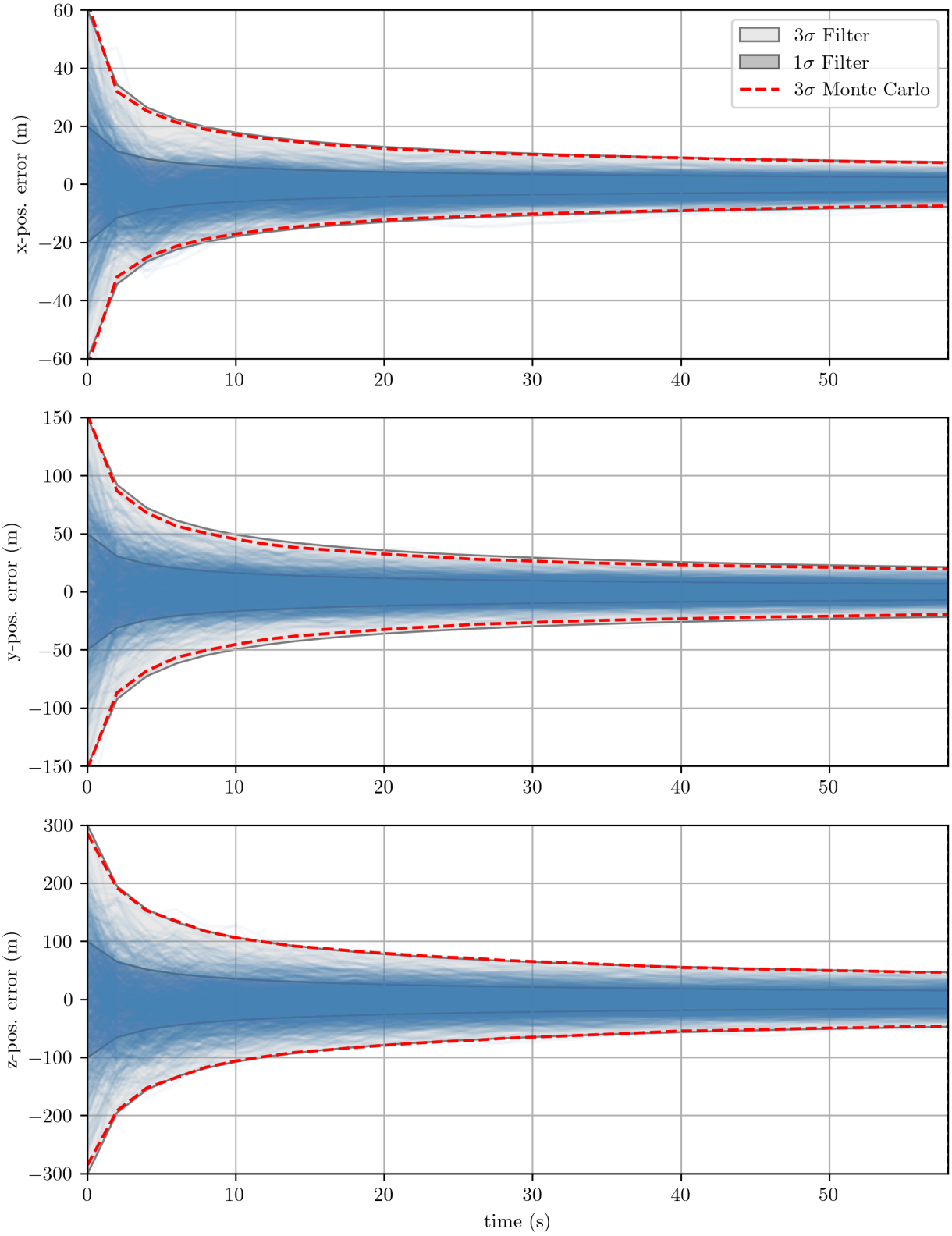

Figure 16. EKF x, y, z post-update position estimation residuals over one minute with observations every 2 seconds. 1000 runs of the filter are shown in blue lines contained within the analytic 1-3σ bounds. The measurements come from constant rotation dynamics.

lens.

The main concern is the computational resources available, where the power and time expenses are linked to the accuracy of any solution due to iterative methods and image size. The selection of development boards is driven by direct camera interfaces and enough RAM to store a DEM. Many development boards exist for use with external cameras via USB or ribbon cables. Low-cost boards include the PC/104 family, NVIDIA Jetson Nano, and Raspberry Pi Compute Module 4 (CM4).[43–45] Additionally, radiation-tolerant processors are of interest, such as the SpaceCube 3 Mini-Z.[46] The main outlier is the PC/104 family, a standard of embedded modules that can be stacked together to form computers. The chosen board must also have enough RAM to store the extensive DEM for a lunar region of interest.

An imaging sensor must be chosen to interface with the computation module. For the analysis shown in this paper, a sensor size of 1024×1024 pixels was used with a FOV of 30°. Star tracker cameras are usually monochrome Complementary Metal-Oxide Semiconductor (CMOS) sensors with locked optics focused at infinity, which may be satisfactory for the horizon. The main concern is obtaining sufficient dynamic range to not overexpose the illuminated objects in the foreground. If a Charge Coupled Device (CCD) were used for the lunar surface, it could lead to pixel overflow for highly exposed portions of the lunar surface compared to the darkness above the horizon. Therefore, a CMOS sensor is a likely candidate.

Three sensors were chosen for comparison in Table 3, and two of them have been investigated for their validity for use as star-trackers and OpNav cameras in a survey.[47] The Raspberry Pi High-Quality Camera is a low-cost option, likely optimal for prototyping before making a flight camera. Two lenses are compared in Table 4, and their comparison is analyzed further in a survey of low-cost star tracker designs.[47] The C-mount lens is compatible with all the sensors in Table 3.

Table 3. Comparison of IDS UI-3180CP-M-GL R2, Ximena MQ013MG-E2, and Raspberry Pi HQ cameras.[48–50]

Feature	IDS	Ximena	RPi HQ
Sensor Type	CMOS	CMOS	CMOS
Resolution	18.1 MP	13 MP	12.3 MP
Interface	USB 3.0	USB 3.0	CSI-2
Pixel Size	4.88 µm	3.45 µm	1.55 µm
Cost ($)	1200	150	50

Three computer modules are compared in Tables 5-7. First, the Nvidia Jetson Nano is a small development board with discrete graphics that are specialized for computer vision processing; it can interface with USB cameras and has CSI ribbon cable ports for other cameras. Next, the Raspberry Pi CM4 is a version of the Raspberry Pi without peripherals attached. Therefore, a separate board is needed to break out specific interfaces, such as the camera and USB interfaces. Next, the PC/104 family of boards is a stackable development board with the potential for substantial computational resources but trades off with its size and power use.

Table 4. Comparison of Fujinon HF12.5HA-1B and Fujinon HF35HA-1B lenses.[51]

Feature	12.5HA	35HA
Focal Length	12.5mm	35mm
Max Aperture	f/1.6	f/1.6
Mount	C-mount	C-mount

Table 5. Hardware specifications of NVIDIA Jetson Nano.

Spec.	NVIDIA Jetson Nano
CPU	4-core ARM @ 1.43 GHz
RAM	4 GB LPDDR4
GPU	128-core Maxwell
Storage	microSD (up to 512 GB)
Power	5W - 15W
Cost ($)	250

Table 6. Hardware specifications of Raspberry Pi CM4.

Spec.	Raspberry Pi CM4
CPU	BCM2711 Cortex-A72 @ 1.5 GHz
RAM	1-8 GB LPDDR4-3200
GPU	VideoCore VI
Storage	eMMC up to 32 GB
Power	3W - 7W
Cost ($)	35 - 80

Ultimately, a Raspberry Pi implementation is favorable for a preliminary prototyping application with a minimum budget. However, something like the NavCube 3.0 Mini-Z is more likely for actual implementation. These boards are highly documented and have many built-in libraries. The CM4 itself has a smaller package, is less power-hungry, and is cheaper than its competitors. The Raspberry Pi CM4 and the Raspberry Pi HQ Camera are recommended in conjunction with the Fujinon HF12.5HA-1B lens to test the validity of running the pipeline in real-time on Earth at the least cost to the user. A shorter focal length is preferable to obtain the most expansive FOV of the horizon as possible.

One drawback of the CM4 is that it requires a separate carrier board. An open-source carrier board was designed to connect a camera and power supply to the Raspberry Pi CM4. This board breaks out I2C for a separate IMU and allows for Ethernet connectivity to simplify networking when prototyping. Fig. 17 shows a rendering of a circuit board designed to interface the CM4 with a camera sensor.

Table 7. Hardware specifications of PC104 Family.

Spec.	PC104 Family
CPU	up to Intel Core i7
RAM	Up to 32 GB DDR4
GPU	Varies (integrated or discrete)
Storage	SSD or HDD
Power	5W - 47W
Cost ($)	200 - 300

Table 8. Hardware specifications of the SpaceCube Mini-Z.

Spec.	SpaceCub Mini-Z
CPU	ARM Cortex-A9
RAM	Up to 1GB DDR3
GPU	N/A
Storage	4GB NAND Flash
Power	1.6W - 3.6W
Cost ($)	Unpublished

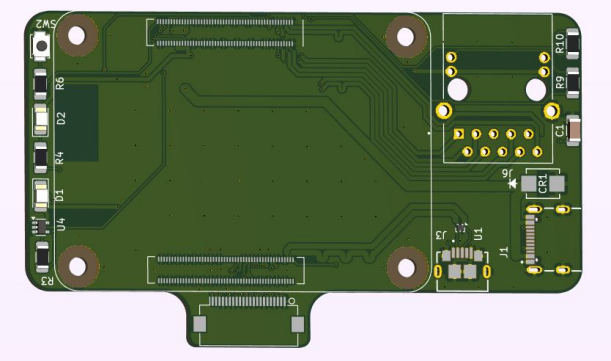

Figure 17. Rendering of CM4 carrier board.

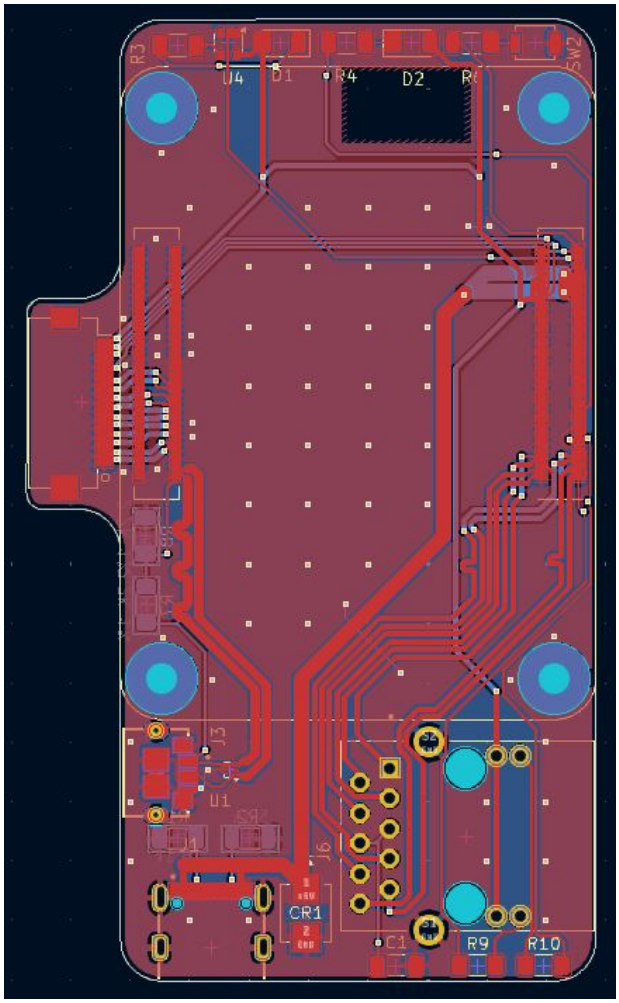

Figure 18. PCB layout, vias, for top and bottom layers.

Conclusion and Future Work. The software pipeline proposed in this paper for lunar surface localization using horizon images and topographical maps is promising for positioning astronauts and robotic systems on the lunar surface with 10s of meters of accuracy. Large-scale statistics inform potential measurement models for implementing the PnP problem parallel to a navigation filter. Most pose output errors are subject to unobservable products of the projective geometry of the PnP problem. With the insufficient depth information cameras provide, positioning errors are the largest along the camera boresight direction. Further improvements in positioning capability are shown with EKF implementations with simple dynamics to simulate imaging while translating and rotating on the surface.

Hardware solutions for a low-cost implementation are discussed, with the Raspberry Pi CM4 as a selected processor. A potential carrier board was designed to decrease package size compared to other boards.

Potential limitations of this method include the significant weight on the accuracy of the *a priori* position and orientation information; this is not a "lost-on-the-Moon" solution. The methods to obtain an initial position within the PnP pipeline's convergence properties are nontrivial and subject to other drawbacks but do exist. Additionally, this technique still does rely on favorable illumination conditions which may not be achievable in all situations.

There are many opportunities for future work on this topic. First, a more realistic navigation filter implementation should be investigated, wherein a simulated IMU is used to provide acceleration and angular velocity data. The PnP algorithm could be further constrained to only query surface points for the solution. Also, a relationship between convergence and location should be determined to increase fidelity within craters or on top of hills. A proper hardware prototype could be made and tested on a rocky, well-mapped part of Earth. Overall, a hardware and software solution described in this paper would allow for safe robotic and human lunar surface operations before more extensive infrastructure is constructed around the Moon.

References.

[1] D. J. Israel, K. D. Mauldin, C. J. Roberts, J. W. Mitchell, A. A. Pulkkinen, L. V. D. Cooper, M. A. Johnson, S. D. Christe, and C. J. Gramling, "Lunanet: a flexible and extensible lunar exploration communications and navigation infrastructure," in *2020 IEEE Aerospace Conference*, pp. 1–14, 2020.

[2] E. Anzalone, L. Carpenter, C. Gramling, L. Mann, and T. Moody, "Early artemis surface navigation: Challenges, approaches, and opportunities," in *Proceedings of the 36th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2023)*, pp. 3597–3611, 2023.

[3] J. A. Christian, "A tutorial on horizon-based optical navigation and attitude determination with space imaging systems," *IEEE Access*, vol. 9, pp. 19819–19853, 2021.

[4] R. Inman, G. Holt, J. Christian, K. W. Smith, and C. D'Souza, "Artemis i optical navigation system performance," in *AIAA SCITECH 2024 Forum*, p. 0514, 2024.

[5] M. Bajracharya, M. W. Maimone, and D. Helmick, "Autonomy for mars rovers: Past, present, and future," *Computer*, vol. 41, no. 12, pp. 44–50, 2008.

[6] V. Verma, M. W. Maimone, D. M. Gaines, R. Francis, T. A. Estlin, S. R. Kuhn, G. R. Rabideau, S. A. Chien, M. M. McHenry, E. J. Graser, *et al.*, "Autonomous robotics is driving perseverance rover's progress on mars," *Science Robotics*, vol. 8, no. 80, p. eadi3099, 2023.

[7] J. J. Parish, A. S. Parish, M. Swanzy, D. Woodbury, D. Mortari, and J. L. Junkins, "Stellar positioning system (part i): an autonomous position determination solution," *Navigation*, vol. 57, no. 1, pp. 1–12, 2010.

[8] D. P. Woodbury, J. J. Parish, A. S. Parish, M. Swanzy, R. Denton, D. Mortari, and J. L. Junkins, "Stellar positioning system (part ii): Improving accuracy during implementation," *Navigation*, vol. 57, no. 1, pp. 13–24, 2010.

[9] J. Amert and M. Fritzinger, "Hardware demonstration and improvements of the stellar positioning system," in *45th Annual AAS Guidance, Navigation and Control (GN&C) Conference*, (Breckenridge, CO), Feb. 2023.

[10] S. Daftry, Z. Chen, Y. Cheng, S. Tepsuporn, S. Khattak, L. Matthies, B. Coltin, U. Naal, L. M. Ma, and M. Deans, "Lunarnav: Crater-based localization for long-range autonomous lunar rover navigation," in *2023 IEEE Aerospace Conference*, pp. 1–15, IEEE, 2023.

[11] L. Matthies, S. Daftry, S. Tepsuporn, Y. Cheng, D. Atha, R. Swan, S. Ravichandar, and M. Ono, "Lunar rover localization using craters as landmarks," pp. 1–17, 03 2022.

[12] A. Cauligi, R. M. Swan, H. Ono, S. Daftry, J. Elliott, L. Matthies, and D. Atha, "Shadownav: Crater-based localization for nighttime and permanently shadowed region lunar navigation," in *2023 IEEE Aerospace Conference*, pp. 1–12, IEEE, 2023.

[13] S. J. Robbins and B. M. Hynek, "A new global database of mars impact craters ≥ 1 km: 1. database creation, properties, and parameters," *Journal of Geophysical Research: Planets*, vol. 117, no. E5, 2012.

[14] S. J. Robbins and B. M. Hynek, "A new global database of mars impact craters ≥ 1 km: 2. global crater properties and regional variations of the simple-to-complex transition diameter," *Journal of Geophysical Research: Planets*, vol. 117, no. E6, 2012.

[15] J. A. Christian, H. Derksen, and R. Watkins, "Lunar crater identification in digital images," *The Journal of the Astronautical Sciences*, vol. 68, no. 4, pp. 1056–1144, 2021.

[16] E. E. Palmer, J. N. Head, R. W. Gaskell, M. V. Sykes, and B. McComas, "Mercator—independent rover localization using stereophotoclinometry and panoramic images," *Earth and Space Science*, vol. 3, no. 12, pp. 488–509, 2016.

[17] S. Chiodini, M. Pertile, S. Debei, L. Bramante, E. Ferrentino, A. G. Villa, I. Musso, and M. Barrera, "Mars rovers localization by matching local horizon to surface digital elevation models," in *2017 IEEE International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, pp. 374–379, IEEE, 2017.

[18] B. Hapke, *The opposition effect*, p. 221–262. Cambridge University Press, 2012.

[19] D. E. Smith, M. T. Zuber, G. B. Jackson, J. F. Cavanaugh, G. A. Neumann, H. Riris, X. Sun, R. S. Zellar, C. Coltharp, J. Connelly, R. B. Katz, I. Kleyner,

P. Liiva, A. Matuszeski, E. M. Mazarico, J. F. McGarry, A.-M. Novo-Gradac, M. N. Ott, C. Peters, L. A. Ramos-Izquierdo, L. Ramsey, D. D. Rowlands, S. Schmidt, V. S. Scott, G. B. Shaw, J. C. Smith, J.-P. Swinski, M. H. Torrence, G. Unger, A. W. Yu, and T. W. Zagwodzki, "The lunar orbiter laser altimeter investigation on the lunar reconnaissance orbiter mission," *Space Science Reviews*, vol. 150, no. 1, pp. 209–241, 2010.

[20] D. E. Smith, M. T. Zuber, G. A. Neumann, E. Mazarico, F. G. Lemoine, J. W. Head III, P. G. Lucey, O. Aharonson, M. S. Robinson, X. Sun, M. H. Torrence, M. K. Barker, J. Oberst, T. C. Duxbury, D. Mao, O. S. Barnouin, K. Jha, D. D. Rowlands, S. Goossens, D. Baker, S. Bauer, P. Gläser, M. Lemelin, M. Rosenburg, M. M. Sori, J. Whitten, and T. Mcclanahan, "Summary of the results from the lunar orbiter laser altimeter after seven years in lunar orbit," *Icarus*, vol. 283, pp. 70–91, 2017. Lunar Reconnaissance Orbiter - Part II.

[21] 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, pp. 346–355, 2016.

[22] F. Cozman, E. Krotkov, and C. Guestrin, "Outdoor visual position estimation for planetary rovers," *Autonomous Robots*, vol. 9, no. 2, pp. 135–150, 2000.

[23] E. Palmer, J. Head, R. Gaskell, M. Sykes, and B. McComas, "Mercator - independent rover localization using stereophotoclinometry and panoramic images," *Earth and Space Science*, vol. 3, 11 2016.

[24] F. Stein and G. Medioni, "Map-based localization using the panoramic horizon," in *Proceedings 1992 IEEE International Conference on Robotics and Automation*, pp. 2631–2637 vol.3, 1992.

[25] F. Stein and G. Medioni, "Map-based localization using the panoramic horizon," *IEEE Transactions on Robotics and Automation*, vol. 11, no. 6, pp. 892–896, 1995.

[26] S. Chiodini, M. Pertile, S. Debei, L. Bramante, E. Ferrentino, A. Villa, and I. Musso, "Mars rovers localization by matching local horizon to surface digital elevation models," *2017 IEEE International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, pp. 374–379, 06 2017.

[27] J. A. Christian and J. L. Crassidis, "Star identification and attitude determination with projective cameras," *IEEE Access*, vol. 9, pp. 25768–25794, 2021.

[28] B. B. Spratling and D. Mortari, "A survey on star identification algorithms," *Algorithms*, vol. 2, no. 1, pp. 93–107, 2009.

[29] D. Lang, D. W. Hogg, K. Mierle, M. Blanton, and S. Roweis, "Astrometry.net: Blind astrometric calibration of arbitrary astronomical images," *The Astronomical Journal*, vol. 139, pp. 1782–1800, 5 2010.

[30] "Ln-200s inertial measurement unit." Accessed: August 29, 2024. `https://www.northropgrumman.com/what-we-do/ln-200s-inertial-measurement-unit`.

[31] J. A. Christian, "Accurate planetary limb localization for image-based spacecraft navigation," *Journal of Spacecraft and Rockets*, vol. 54, no. 3, pp. 708–730, 2017.

[32] A. J. Liounis, "Limb-based optical navigation for irregular bodies," 2018.

[33] A. J. Liounis and K. Getzandanner, "Operational Performance of Limb-Based Navigation from Osiris-Rex at Bennu," in *3rd Space Imaging Workshop*, 2022.

[34] A. J. Liounis, C. Gnam, J. Swenson, K. Getzandanner, J. Small, and J. Lyzhoft, "The Goddard Image Analysis and Navigation Tool," Jan. 2022.

[35] A. Trujillo-Pino, K. Krissian, M. Alemán-Flores, and D. Santana-Cedrés, "Accurate subpixel edge location based on partial area effect," *Image and Vision Computing*, vol. 31, no. 1, pp. 72–90, 2013.

[36] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o(n) solution to the pnp problem," *International Journal of Computer Vision*, vol. 81, 02 2009.

[37] G. Terzakis and M. Lourakis, "A consistently fast and globally optimal solution to the perspective-n-point problem," in *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I*, (Berlin, Heidelberg), p. 478–494, Springer-Verlag, 2020.

[38] Y. Abdel-Aziz and H. Karara, "Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry," *Photogrammetric Engineering & Remote Sensing*, vol. 81, no. 2, pp. 103–107, 2015.

[39] H. Yang, P. Antonante, V. Tzoumas, and L. Carlone, "Graduated non-convexity for robust spatial perception: From non-minimal solvers to global outlier rejection," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1127–1134, 2020.

[40] P. David, D. DeMenthon, R. Duraiswami, and H. Samet, "Softposit: Simultaneous pose and correspondence determination," *International Journal of Computer Vision*, vol. 59, no. 3, pp. 259–284, 2004.

[41] R. Sinkhorn, "A Relationship Between Arbitrary Positive Matrices and Doubly Stochastic Matrices," *The Annals of Mathematical Statistics*, vol. 35, no. 2, pp. 876 – 879, 1964.

[42] "LRO LOLA - PDS Geosciences Node Data and Services."

[43] "NVIDIA Jetson Nano." Accessed: September 4, 2024. `https://developer.nvidia.com/embedded/jetson-nano`.

[44] "Raspberry Pi Compute Module 4." Accessed: September 4, 2024. `https://www.raspberrypi.com/products/compute-module-4/?variant=raspberry-pi-cm4001000`.

[45] "PC/104 consortium standard." Accessed: September 4, 2024. `https://pc104.org/hardware-specifications/pc104/`.

[46] C. Wilson, "GSFC annual scan technology review spacecube on-board processor update." Accessed: September 4, 2024. `https://ntrs.nasa.gov/citations/20200000976`.

[47] S. Pedrotty, R. Lovelace, J. Christian, D. Renshaw, and G. Quintero, "Design and performance of an open-source star tracker algorithm on commercial off-the-shelf cameras and computers," 02 2020.

[48] "IDS Imaging U3-3180CP-M-GL Camera." Accessed: September 4, 2024. `https://www.edmundoptics.com/p/ids-imaging-u3-3180cp-m-gl-1-monochrome-usb3-camera/49792/`.

[49] "XIMEA - MQ013mg-e2 camera sensor." Accessed: September 4, 2024. `https://www.ximea.com/en/products/cameras-filtered-by-sensor-types/mq013mg-e2`.

[50] "Camera - Raspberry Pi Documentation." Accessed: September 4, 2024. `https://www.raspberrypi.com/documentation/accessories/camera.html`.

[51] "HF-HA-1S Series." Accessed: September 4, 2024. `https://www.fujifilm.com/us/en/business/optical-devices/mvlens/hf-ha-1s-series`.

[52] A. V. Nefian, X. Bouyssounouse, L. Edwards, T. Kim, E. Hand, J. Rhizor, M. Deans, G. Bebis, and T. Fong, "Planetary rover localization within orbital maps," 2014.

[53] D. Gale and L. S. Shapley, "College admissions and the stability of marriage," *The American Mathematical Monthly*, vol. 69, no. 1, pp. 9–15, 1962.

[54] P. David, D. DeMenthon, R. Duraiswami, and H. Samet, "Softposit: Simultaneous pose and correspondence determination," *International Journal of Computer Vision*, vol. 59, 09 2003.

[55] R. Sinkhorn, "A Relationship Between Arbitrary Positive Matrices and Doubly Stochastic Matrices," *The Annals of Mathematical Statistics*, vol. 35, no. 2, pp. 876 – 879, 1964.

[56] E. Mazarico, M. K. Barker, and J. B. Nicholas, "Advanced illumination modeling for data analysis and calibration. application to the moon," *Advances in Space Research*, vol. 62, no. 11, pp. 3214–3228, 2018.

[57] O. Vincent and O. Folorunso, "A descriptive algorithm for sobel image edge detection," 01 2009.

[58] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *Int. J. Comput. Vision*, vol. 13, p. 119–152, oct 1994.

[59] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," in *Readings in Computer Vision* (M. A. Fischler and O. Firschein, eds.), pp. 726–740, San Francisco (CA): Morgan Kaufmann, 1987.

[60] J. L. Kloos, J. E. Moores, P. J. Godin, and E. Cloutis, "Illumination conditions within permanently shadowed regions at the lunar poles: Implications for in-situ passive remote sensing," *Acta Astronautica*, vol. 178, pp. 432–451, 2021.

[61] J. H. Friedman, J. L. Bentley, and R. A. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Trans. Math. Softw.*, vol. 3, p. 209–226, sep 1977.

[62] D. Gale and L. S. Shapley, "College admissions and the stability of marriage," *The American Mathematical Monthly*, vol. 69, no. 1, pp. 9–15, 1962.

[63] L. Matthies, S. Daftry, S. Tepsuporn, Y. Cheng, D. Atha, R. M. Swan, S. Ravichandar, and M. Ono, "Lunar rover localization using craters as landmarks," in *2022 IEEE Aerospace Conference (AERO)*, pp. 1–17, IEEE, 2022.

[64] F. Barone, M. Marrazzo, and C. J. Oton, "Camera calibration with weighted direct linear transformation and anisotropic uncertainties of image control points.," *Sensors (Basel)*, vol. 20, Feb 2020.

[65] W. Forstner, "New orientation procedures," *International Archives of Photogrammetry and Remote Sensing*, vol. 33, no. B3/1; PART 3, pp. 297–304, 2000.

Appendix: Direct Linear Transform for Translation-Only Solutions. The Direct Linear Transform (DLT) is a method for quickly computing the least squares solution for the linear system that best projects a set of 3D points into a 2D plane, given a set of correspondences.

For this work, a novel variant of DLT called the translation-only DLT is derived. In this problem, only the translation vector $\mathbf{t}$ is estimated between the world coordinate system and the camera coordinate system, assuming the camera matrix and rotation are already given. As a further refinement of this method, by iteratively reweighting the weight matrix by the pixel error to reprojection error Jacobian, the statistically optimal solution is converged upon rather than the least squares solution.

We begin by recasting our problem in the form of the DLT problem

$$\mathbf{y}_k \propto \mathbf{A}\mathbf{x}_k$$

where $\mathbf{y}_k \in \mathbb{R}^3$ are homogeneous coordinates, $\mathbf{x}_k \in \mathbb{R}^4$ are homogeneous coordinates, and $\mathbf{A} \in \mathbb{R}^{3\times 4}$ is the projection matrix. Redefine $\mathbf{K}$ to work with homogeneous coordinates

$$\mathbf{K}' = \begin{bmatrix} k_x & \alpha & c_x \\ 0 & k_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

and redefine $\mathbf{p}_i$ and $\mathbf{m}_i$ to be

$$\mathbf{p}_i' = \begin{bmatrix} u_i \\ v_i \\ w_i \end{bmatrix}$$

$$\mathbf{m}_i' = \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix}$$

We now have a proportional projection of

$$\mathbf{p}_i' \propto \mathbf{K}' \begin{bmatrix} \mathbf{I}_{3\times 3} & -\mathbf{t} \end{bmatrix} \mathbf{m}_i'.$$

To simplify, define projection matrix $\mathbf{D} = \mathbf{K}' \begin{bmatrix} \mathbf{I}_{3\times 3} & -\mathbf{t} \end{bmatrix}$ which is 3×4 and can be described according to the rows of the matrix using

$$\mathbf{D} = \begin{bmatrix} \mathbf{d}_1^T \\ \mathbf{d}_2^T \\ \mathbf{d}_3^T \end{bmatrix}.$$

Now, following the usual DLT steps,[65] we can remove the proportionality by enforcing $w_i = 1$ in $\mathbf{p}_i'$ by dividing through by the third row of the right hand side giving

$$\mathbf{p}_i' = \frac{\mathbf{D}\mathbf{m}_i'}{\mathbf{d}_3^T \mathbf{m}_i'}$$

This results in a system of two equations which we can rearrange to get

$$\begin{bmatrix} k_x & \alpha & -a_i & a_i z_i - \alpha y_i - k_x x_i \\ 0 & ky & -b_i & b_i z_i - k_y y_i \end{bmatrix} \begin{bmatrix} \mathbf{t} \\ 1 \end{bmatrix} = \mathbf{0}_{2\times 1}$$

where $a_i = u_i - c_x$ and $b_i = v_i - c_y$ are defined for convenience. Using n observations with these equations give a system of $2n$ equations with only 3 unknowns. We can choose to solve the equations for the null space of the matrix using SVD as is typically done for DLT, enforcing the constraint that the scale of the vector much be such that the fourth component is 1. Alternatively we can rearrange to solve a $2n \times 3$ least squares problem. It is trivial to back out the estimate for $\mathbf{t}$ in either case.

Weighted DLT. At this point, we have an unweighted solution; however, we would like a weighted solution. Following the procedure in Ref. [64], we can add weights for each observation by left-multiplying by the weight matrix for each observation (nominally computed according to the Data Association subsection) using

$$\mathbf{W}_i \mathbf{M}_i \begin{bmatrix} \mathbf{t} \\ 1 \end{bmatrix} = \mathbf{0}_{2\times 1}$$

where $\mathbf{W}_i$ is the 2×2 weight matrix for the i^{th} observation and $\mathbf{M}_i$ is the coefficient matrix from the system of equations.