Terrain-Relative Navigation with Neuro-Inspired Elevation Encoding

Kristen Michaelson
Dept. of Aerospace Engineering and Engineering Mechanics
The University of Texas at Austin
Austin, TX, USA
kmichaelson@utexas.edu

Felix Wang
Center for Computing Research
Sandia National Laboratories
Albuquerque, NM, USA
felwang@sandia.gov

Renato Zanetti
Dept. of Aerospace Engineering and Engineering Mechanics
The University of Texas at Austin
Austin, TX, USA
renato@utexas.edu

***Abstract*—Terrain-relative autonomous navigation is a challenging task. In traditional approaches, an elevation map is carried onboard and compared to measurements of the terrain below the vehicle. These methods are computationally expensive, and it is impractical to store high-quality maps of large swaths of terrain. In this article, we generate position measurements using NeuroGrid, a recently-proposed algorithm for computing position information from terrain elevation measurements. We incorporate NeuroGrid into an inertial navigation scheme using a novel measurement rejection strategy and online covariance computation. Our results show that the NeuroGrid filter provides highly accurate state information over the course of a long trajectory.**

***Index Terms*—kalman filter, inertial navigation, terrain-relative navigation, neuro-inspired**

I. Introduction

During aided inertial navigation, information from onboard inertial measurement units is fused with periodic observations of the environment. Global Positioning System (GPS) receivers have become ubiquitous in aerospace applications due to their accuracy and ease of use. Recently, however, interest has grown in GPS-denied navigation. One approach to GPS-denied navigation is terrain-relative navigation (TRN). TRN relies on measurements of the terrain below the vehicle. If a patch of terrain can be uniquely identified and correlated to an onboard map, then positioning information becomes available. In this work, we generate position measurements with NeuroGrid; a neuro-inspired, grid-based localization algorithm proposed by Wang et al [1]. The measurement is derived from a sum of "grid activations" computed from features in the environment, analogous to brain activity associated with positioning in rats [2]–[9].

While numerous TRN algorithms exist, they all share some commonality: measurements of the terrain below the vehicle are used to estimate the vehicle's position above it. This estimate is computed (directly or indirectly) from a known map of the terrain. Often, the map is a digital elevation model (DEM), which represents the elevation of the terrain at regular intervals over a given area. In traditional approaches, the DEM is carried onboard. Terrain Contour Matching (TERCOM) and Sandia Inertial Terrain-Aided Navigation (SITAN) are two examples of early TRN algorithms [10], [11]. TERCOM computes a position measurement by sweeping a short burst of elevation measurements over the DEM, whereas SITAN processes each elevation measurement individually to continuously update the state estimate. These methods are comparable in nature to iterative closest point (ICP), a point cloud matching algorithm popular in the robotics community [12].

This work presents an inertial navigation filter with a neuro-inspired terrain-relative position measurement. Neuro-inspired methods are an attractive approach to terrain-relative navigation problems, since many "matching" algorithms—like the ones described above—are computationally expensive optimizations. Much recent work on neuro-inspired TRN employs neural networks to identify craters and other features on the moon. For example, convolutional neural networks are able to accurately bound and label craters in images of the lunar surface. Information about the geography of successfully identified craters is accessed in an onboard catalog of lunar craters. Using this approach, satellites operating near the moon could use craters as navigation aids, rather than raw data from images or laser scans [13]–[16]. Machine learning methods have also been proposed for autonomous hazard detection [17], [18] and guidance [19] during spacecraft landing.

Instead of a DEM or a catalog of surface features, the NeuroGrid algorithm computes positioning information using a *phase candidate dictionary* that associates elevation values with two-dimensional phase coordinates used in the algorithm's grid-based representation (see Section II-B). When the vehicle measures elevations below it, candidate phase values are accessed using simple dictionary lookups. A single phase is

This article has been authored by an employee of National Technology Engineering Solutions of Sandia, LLC under Contract No. DE-NA0003525 with the U.S. Department of Energy (DOE). The employee owns all right, title and interest in and to the article and is solely responsible for its contents. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this article or allow others to do so, for United States Government purposes. The DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan https://www.energy.gov/downloads/doe-public-access-plan.

chosen for each *grid* using a winner-take-all approach. These phase values are used to construct *grid images* reminiscent of activity observed in the brains of animals as they explore a new environment. The grid images are summed to produce a position estimate.

In this work, we use the NeuroGrid algorithm to produce two-dimensional position measurements from LIDAR scans. These position measurements are used to update the state estimate in an Extended Kalman Filter (EKF). Because NeuroGrid requires a heading value to compute position, we add a magnetometer to the vehicle. We also employ a simple measurement rejection technique at the phase candidate level, which saves computation time. Finally, we introduce a way to compute the uncertainty of each NeuroGrid measurement online using the grid sum.

II. Background

A. Biological Inspiration

Researchers study navigation in animals by recording electrical activity in their brains. Activity in one group of cells, the *grid cells*, has been identified as a key component of self-localization. Fig. 1 shows the results of an experiment where rats explore a rectangular enclosure [8].

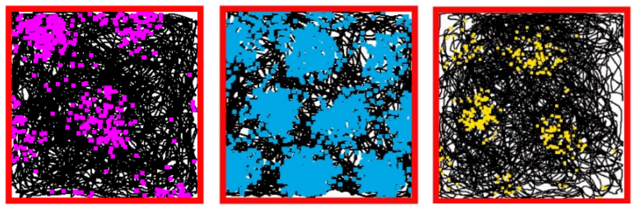

Fig. 1. Grid cell activity during three experiments with rats. A probe measures the activity of a single grid cell in each animal's brain. Spikes in activity in the cell (left: purple, center: blue, right: yellow) are superimposed on the animal's path (black). Adapted from [8] (see Supplementary Information, Fig. 3).

Remarkably, even though the animal roams freely, the grid cell activity is characterized by distinct spatial clusters. These clusters form a hexagonal grid pattern. The brain contains hundreds of grid cells. Each cell fires with distinct *scale*, *orientation*, and x- and y- *spatial offset*. Solstad et al. (2006) proposed a computational representation of grid cell activity based on these four parameters that we employ in this work [3].

While it is clear that grid cell activity is related to localization, the activity of a single grid cell does little to identify the animal's position in the space. Indeed, if we wished to decode the activity of a single cell, all we could claim is that the animal is located in one of the areas on the grid associated with activity in that cell. However, in combination, the activity of a large number of grid cells can yield a very precise position estimate. Exactly how animals process grid cell activity to form a position estimate is still not completely clear. Solstad et al. propose simply summing the grid activations associated with all the cells firing at a certain time; in this way, the animal's location "pops out" at a point where all the activated grids intersect (see, e.g., [3] Fig. 2 or [1] Fig. 9).

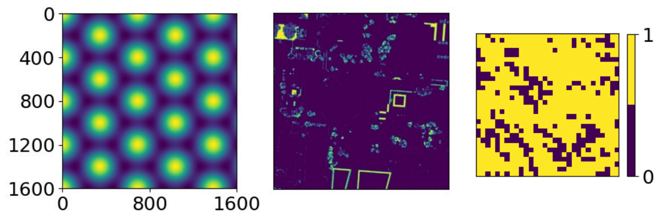

Fig. 2. Left: Visualization of grid with spatial scale $\lambda = 400$ px and angle $\theta = 0°$. Center: Elevation contour containing points between 174 m and 176 m (see Fig. 5). The size of the map, like the grid image (left), is 1600 $\times$ 1600 px. Right: Phase candidate matrix for points on the elevation contour 175 m to 175.1 m for grid ($\lambda = 400$ px, $\theta = 0°$). Extra elevation bins are shown in the center image for clarity.

In order to adapt this approach for precise positioning in autonomous navigation, grid cell activity must be associated with environmental features. This way, data from observations of the environment can be translated into unique position estimates.

B. The Phase Candidate Dictionary

The phase candidate dictionary is a data structure proposed in Wang et al. (2022) that associates elevation values with grid activity [1], [3]. In this work, a vehicle flies at constant altitude and records elevation values at points in a small area below it (see Section III). To build the phase candidate dictionary, we first randomly generate a collection of N *grid modules* with spatial scale λ_i and orientation θ_i, where $i = 1, \ldots, N$. The grid modules represent all the grids with the same scale and orientation (λ_i, θ_i) but different spatial phase offset $(\phi_{x,i}, \phi_{y,i})$. Then, the DEM is divided into elevation bins. These bins form elevation contours (Fig. 2, center). For each pair (x_j, y_j) on each elevation contour, we determine the phase offset $(\phi_{x,i}, \phi_{y,i})$ associated with grid (λ_i, θ_i) such that a peak of grid (λ_i, θ_i) would appear at (x_j, y_j) (see [1], Eq. 1).

The phase offsets $(\phi_{x,i}, \phi_{y,i}) \in [0, 2\pi) \times [0, 2\pi)$ are placed into phase bins. In this work, we use 30 phase bins with size $2\pi/30$. When a phase value $(\phi_{x,i}, \phi_{y,i})$ is found on the elevation contour, a 1 is placed at the index corresponding to the appropriate phase bin on a 30×30 *phase candidate matrix*. Broadly speaking, the phase candidate matrix encodes information about spatial periodicity in the elevation contour. This process is illustrated in Fig. 2. The phase candidate dictionary contains one phase candidate matrix for each grid (λ_i, θ_i) for each elevation contour. If we choose to encode N grids, and we slice the DEM into M elevation contours, then the phase candidate dictionary contains $N \times M$ phase candidate matrices. The phase candidate matrices are accessed using simple dictionary lookups. The lookup keys are grid module (λ_i, θ_i) and an elevation bin.

C. NeuroGrid

The NeuroGrid algorithm produces an (x, y) position estimate using data from elevation measurements. In this work, a vehicle flying at constant altitude records LIDAR measurements of the terrain below it. The (x, y) position estimate is used in the measurement update in an EKF. NeuroGrid is summarized in Table I.

TABLE I
THE NEUROGRID ALGORITHM

Require: (1) A phase candidate dictionary built from the navigation map in the manner described in Section II-B; (2) a sensor measurement of P elevation values; (3) A set of spatial offsets $(x_{o,j}, y_{o,j})$, $j = 1 \ldots P$: the locations of the elevation measurements relative to the vehicle in the inertial frame

1) Perform a phase candidate lookup for each of the P elevations and each of the N grid modules.
2) Perform a shifting operation using the spatial offsets $(x_{o,j}, y_{o,j})$. The phase candidate matrices must be shifted such that the phase estimate is reflective of the vehicle position, not the location of each individual elevation measurement.
3) Sum the shifted phase candidate matrices over all measurements for each grid module (λ_i, θ_i) (see Fig. 3).
4) From this sum, choose the most likely accompanying $(\phi_{x,i}, \phi_{y,i})$ for each grid module. This value is simply the `arg max` of the summed candidate matrices.
5) Compute and sum grid activations from $(\lambda_i, \theta_i, \phi_{x,i}, \phi_{y,i})$ for each grid module (see Fig. 4). The value `arg max` of this sum is the (x, y) position estimate.

First, a dictionary lookup is performed for each elevation value for grid module (λ_i, θ_i). This garners P phase candidate matrices (see Fig. 2, right). After a shifting operation (since the elevation values are spread out around different (x, y) points beneath the vehicle), the phase candidate matrices are summed (see Fig. 3). The phase estimate $(\phi_{x,i}, \phi_{y,i})$ for grid module (λ_i, θ_i) is then the `arg max` of the sum. These operations are repeated for each of the N grid modules. An image of each grid is generated using the four values $(\lambda_i, \theta_i, \phi_{x,i}, \phi_{y,i})$ (see Fig. 2, left). The grid image has the same pixel resolution as the DEM. Finally, the N grid images are summed as they are computed. The `arg max` of the sum of grid images is the location estimate (x, y). Fig. 4 shows a sum of 25 grid images.

III. INERTIAL NAVIGATION

The NeuroGrid location estimate is used as an aiding measurement in an inertial navigation filter. The filter carries five states:

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \\ \theta \end{bmatrix} \tag{1}$$

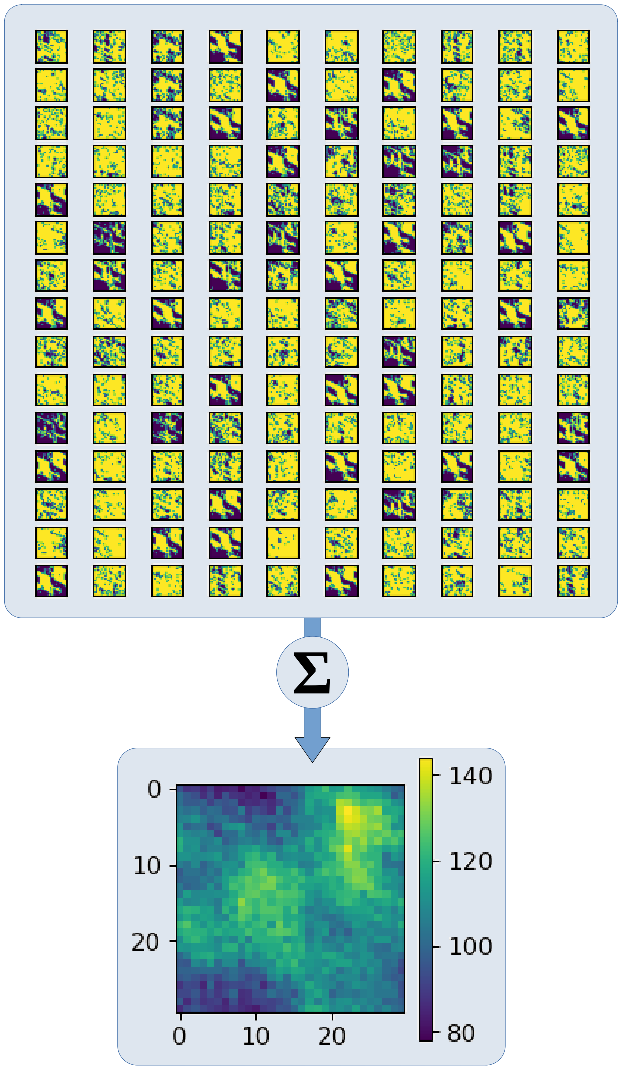

Fig. 3. Sum of shifted 30×30 binary phase candidate matrices for $P = 150$ elevation measurements for grid module $(\lambda_i = 357\text{px}, \theta_i = 206°)$. The `arg max` value is [3,22], which corresponds to phase estimate $(\phi_{x,i} = 22 * \frac{2\pi}{30}, \phi_{y,i} = 3 * \frac{2\pi}{30})$.

where $\mathbf{p} = \begin{bmatrix} p_x & p_y \end{bmatrix}^T$ is the position of the vehicle in the inertial frame, $\mathbf{v} = \begin{bmatrix} v_x & v_y \end{bmatrix}^T$ is the velocity, and θ is the heading angle. The vehicle flies at a constant altitude of 350 m.

A. Dynamics Propagation

The dynamics are propagated using measurements from an onboard inertial measurement unit (IMU). The IMU measures acceleration $\mathbf{a_m}$ and angular rate ω_m in the body frame:

$$\mathbf{a_m} = T_i^b \mathbf{a} + \eta_a \tag{2}$$

where $\mathbf{a}$ is the true inertial-frame acceleration, T_i^b is the 2×2 direction cosine matrix that represents the the transformation from the inertial frame to the body frame, and η_a is an additive white Gaussian noise with power spectral density $PSD_a = 1.361 \times 10^{-6}$ m^2/s^3 [20].

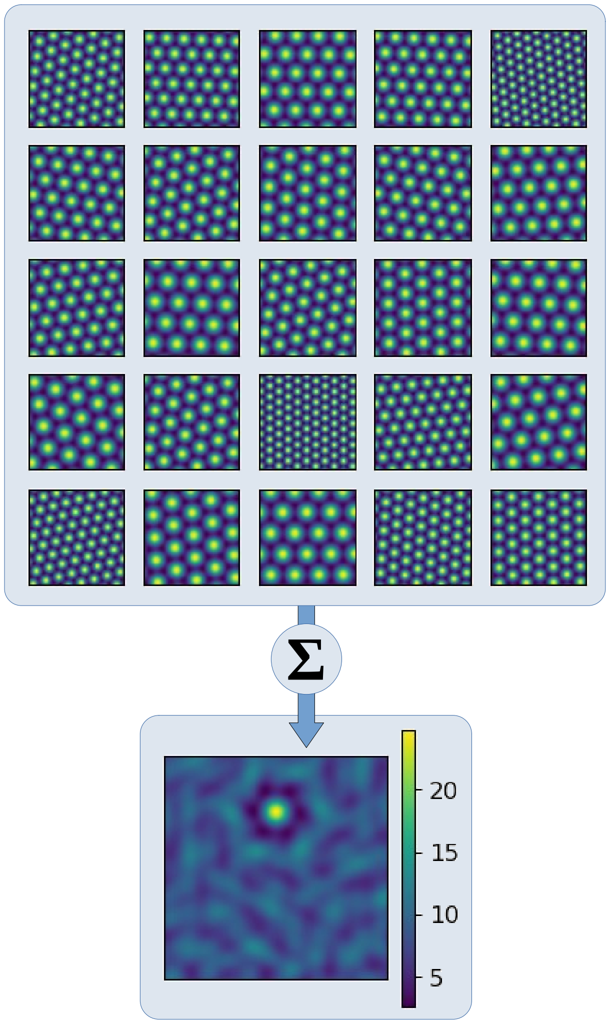

Fig. 4. Grid sum of $N = 25$ 1600×1600 grid images. The vehicle is located at the center of the yellow circle in the bottom image.

Similarly, the gyroscope measurement is

$$\omega_m = \omega + \eta_g \tag{3}$$

where ω is the true angular rate and η_g is an additive white Gaussian noise with power spectral density $PSD_g = 6.250 \times 10^{-6}$ deg^2/s [20].

The state dynamics are $\dot{\mathbf{x}} = f(\mathbf{x})$, where

$$f(\mathbf{x}) = \begin{bmatrix} \mathbf{v} \\ \mathbf{a} \\ \omega \end{bmatrix}. \tag{4}$$

The estimated state, $\hat{\mathbf{x}}$, is propagated in discrete time at the IMU rate:

$$\hat{\mathbf{p}}_{k+1} = \hat{\mathbf{p}}_k + \hat{\mathbf{v}}_k \Delta t + \frac{1}{2}\hat{\mathbf{a}}_k \Delta t^2 \tag{5}$$

$$\hat{\mathbf{v}}_{k+1} = \hat{\mathbf{v}}_k + \hat{\mathbf{a}}_k \Delta t \tag{6}$$

$$\hat{\theta}_{k+1} = \hat{\theta}_k + \omega_m \Delta t \tag{7}$$

where

$$\hat{\mathbf{a}}_k = \hat{T}_b^i \mathbf{a_m} \tag{8}$$

$\hat{T}_b^i$ is a direction cosine matrix computed from the heading angle estimate $\hat{\theta}$, and Δt is the time between time step k and time step $k+1$.

The discrete-time covariance propagation is

$$P_{k+1} = \Phi P_k \Phi^T + BQB^T \tag{9}$$

where $\Phi = e^{F\Delta t}$, and F is the Jacobian of the continuous-time error dynamics:

$$F = \begin{bmatrix} \mathbf{0}_{2\times2} & I_{2\times2} & \mathbf{0}_{2\times1} \\ \mathbf{0}_{2\times2} & \mathbf{0}_{2\times2} & \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \hat{\mathbf{a}}_k \\ \mathbf{0}_{1\times2} & \mathbf{0}_{1\times2} & 0 \end{bmatrix}. \tag{10}$$

Note that the skew-symmetric term in Eq. 10 manifests as a result of a partial derivative of direction cosine matrix $\hat{T}_b^i$ with respect to the heading angle. B is the Jacobian of the error dynamics with respect to the noises η_a and η_g:

$$B = \begin{bmatrix} -\frac{1}{2}\hat{T}_b^i \Delta t & \mathbf{0}_{2\times1} \\ -\hat{T}_b^i & \mathbf{0}_{2\times1} \\ \mathbf{0}_{1\times2} & 1 \end{bmatrix} \tag{11}$$

and

$$Q = \begin{bmatrix} (PSD_a \Delta t) I_{2\times2} & \mathbf{0}_{2\times1} \\ \mathbf{0}_{1\times2} & PSD_g \Delta t \end{bmatrix}. \tag{12}$$

B. Measurement Update

The LIDAR measurement is a three-dimensional point cloud in the body frame. Each point is measured as

$$\mathbf{p}_{m,j} = T_i^b \mathbf{p}_j + \epsilon_j \tag{13}$$

where $\mathbf{p}_j = \begin{bmatrix} x_j & y_j & z_j \end{bmatrix}^T$ is the location of the point in the inertial frame, and

$$\epsilon_j \sim \mathcal{N}(\mathbf{0}_{2\times1}, R_{LIDAR}) \tag{14}$$

where $R_{LIDAR} = 0.05^2 I_{3\times3}$ m^2.

The elevations e_j of each point are known, since the vehicle flies at constant altitude $\bar{z}$.

$$e_j = \bar{z} - z_{m,j} \tag{15}$$

Here, $z_{m,j}$ is the distance to a point below the vehicle along the body z-axis, which points downward along the inertial z-axis. P elevations are randomly selected from the point cloud measurement and sorted into bins. The bin size is the same one used to build the phase candidate dictionary.

NeuroGrid also requires an estimate of the inertial (x, y) offsets between the location of the vehicle and each of the elevation values e_j. To this end, we add a magnetometer that measures the heading angle at the time of the LIDAR measurement. The magnetometer measurement is

$$\theta_m = \theta + \xi \tag{16}$$

$$\xi \sim \mathcal{N}(0, R_{mag}) \tag{17}$$

where θ is the true heading angle, and $R_{mag} = 0.8333^2$ deg^2 [21]. The inertial (x, y) offsets are then

$$\begin{bmatrix} x_{o,j} \\ y_{o,j} \end{bmatrix} = T_b^i(\theta_m) \begin{bmatrix} x_{m,j} \\ y_{m,j} \end{bmatrix} \tag{18}$$

where the heading measurement θ_m has been used to express $\begin{bmatrix} x_j & y_j \end{bmatrix}^T$ from the LIDAR measurement in the inertial frame. The inputs to NeuroGrid are the P elevation values e_j and and associated spatial offset values $(x_{o,j}, y_{o,j})$.

The NeuroGrid algorithm produces an (x, y) position estimate that corresponds to an (i, j) index on the DEM that was used to build the phase candidate dictionary. The NeuroGrid measurement model is

$$\mathbf{y} = \begin{bmatrix} p_x \\ p_y \end{bmatrix} + \gamma \tag{19}$$

where

$$\gamma \sim \mathcal{N}(\mathbf{0}_{2\times 1}, R) \tag{20}$$

and R is computed in the manner described in Section III-D.

The Kalman update is

$$\Delta\mathbf{y} = \mathbf{y} - \hat{\mathbf{y}} \tag{21}$$

$$W = HP_k^- H^T + R \tag{22}$$

$$K = P_k^- H^T W^{-1} \tag{23}$$

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + K\Delta\mathbf{y} \tag{24}$$

$$P_k^+ = (I - KH)P_k^-(I - KH)^T + KRK^T \tag{25}$$

where $\hat{\mathbf{y}}$ is the predicted measurement $\begin{bmatrix} \hat{p}_x & \hat{p}_y \end{bmatrix}^T$ computed from the state values and Eq. 25 is the Joseph-form covariance update [22].

C. Measurement Rejection

NeuroGrid is a "winner-take-all" approach to generating a position estimate. As Wang et al. point out in [1], some of the phase estimates computed from the LIDAR measurement simply are not meaningful. This happens when the sum of the phase candidate matrices is noisy (see Fig. 3 and [1], Fig. 8). If the phase candidate sum associated with grid (λ_i, θ_i) does not have a clear maximum value, then the corresponding phase estimates $(\phi_{x,i}, \phi_{y,i})$ are liable to produce a grid image (see Fig. 2, left) that does not have a peak at the location of the vehicle.

The grid sum that is used compute the NeuroGrid position estimate is the sum of N grid images. Grids that have a peak at the vehicle's location contribute to a large spike in the grid sum at that point. Grids that do not have a peak at the vehicle's location fade into the background.

To save computation time, we propose a simple measurement rejection technique at the phase candidate sum level. The measurement acceptance criteria is

$$\max(S_{pc}) - \text{mean}(S_{pc}) > P/10 \tag{26}$$

where S_{pc} is the phase candidate sum. If the inequality in Eq. 26 is satisfied, then the phase candidate sum is accepted. Otherwise, it is rejected. The maximum possible value at any index is P, since there are P elevation measurements. Criteria (26) requires that the maximum value of the phase candidate sum is greater than $P/10$ higher than the mean. If grid (λ_i, θ_i) is rejected based on criteria (26), its grid image is not computed nor added to the grid sum.

D. Online Covariance Computation

The NeuroGrid position estimate is the `arg max` of the grid sum, an image with the same pixel resolution as the DEM (see Fig. 4). Grids that have peaks at the location of the vehicle constructively interfere, causing a large spike in activity. This activity is also high in an approximately circular area around the vehicle. We define the covariance of the NeuroGrid measurement based on the slope of the activity peak. If the `arg max` of the grid sum is a clear maximum, and the pixels surrounding it fall off in value quickly, then the standard deviation of the position measurement is close to the pixel width of the grid sum. If multiple surrounding pixels have values close to the `arg max`, then the standard deviation is multiple pixel widths.

TABLE II
COVARIANCE OF THE NEUROGRID POSITION MEASUREMENT

```
Require: (1) A grid sum image $G$; (2) A pixel-to-m conversion factor $\alpha$, which is the resolution of $G$
1) Find the index $(i, j) = \texttt{arg max}\{G\}$.
2) Step down the rows $n$ times until $G(i+n, j) < c * \texttt{arg max}\{G\}$, where $0 < c < 1$.
3) Set $\sigma = \alpha n$.
4) Set $R = \begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{bmatrix}$.
```

The covariance computation scheme is summarized in Table II. It requires a constant value c, where $0 < c < 1$. As c increases, the algorithm is more and more likely to compute a σ-value of exactly one pixel. It returns a diagonal covariance matrix $R = \begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{bmatrix}$.

IV. RESULTS

We test the filter on data from a simulated flight over a section of the University of Texas at Austin campus [23]. The vehicle flies in a circular trajectory at a constant altitude of 350 m. The radius of the circle is 100 m. The vehicle completes one revolution around the circle in 60 s. As it flies, its heading points in the direction of the velocity vector. The IMU propagation rate is 100 Hz. Fig. 5 shows the trajectory.

The initial uncertainty is 10 m in position, 1 m/s in velocity, and 1 deg in heading:

$$P_0 = \begin{bmatrix} 10^2 I_{2\times 2} & 0 & 0 \\ 0 & 1.0^2 I_{2\times 2} & 0 \\ 0 & 0 & (1.0\frac{\pi}{180})^2 \end{bmatrix}. \tag{27}$$

A pair of LIDAR and heading measurements are recorded every 5 seconds. The phase candidate dictionary was built

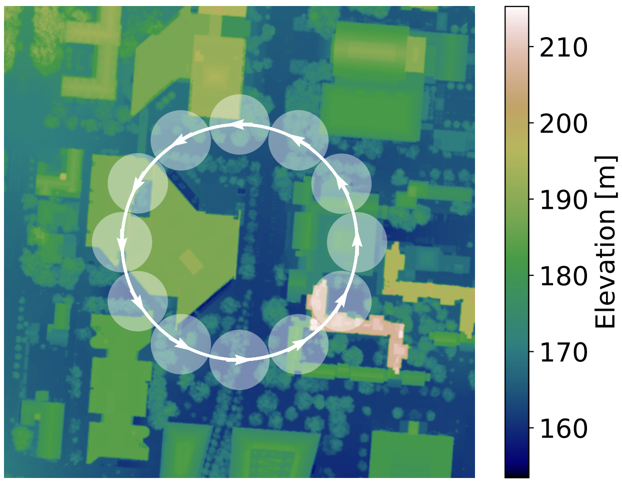

Fig. 5. Trajectory for the simulated flight. Arrows indicate the vehicle's heading. Measurements of the terrain are recorded every 5 seconds. $P = 150$ random points from each measurement are selected for processing. The chosen points lie within a 25 m ground radius of the vehicle (white circles).

with a DEM resolution of 0.25 m/px and an elevation bin resolution of 0.1 m. It encodes phase information for 50 randomly generated grids with minimum scale $\lambda_i = 10$ m and maximum scale $\lambda_i = 100$ m.

Fig. 6 shows the state estimation error computed during 50 Monte Carlo runs. The measurement covariance is computed online using the method described in Section III-D, and phase candidates are rejected using the criteria in Eq. 26. A value of $c = 0.9995$ was chosen for the covariance computation. Even with such a large value of c, the filter is slightly conservative. This means that the NeuroGrid position measurements are slightly better than the "spread" of the maxima in the grid sum would suggest; even though there are values very close to the maximum in proximity to the maximum, the maximum is almost always the best estimate of the position.

Since NeuroGrid is a global position measurement, it is able to overcome very large initial position and velocity errors. The estimator is also consistent; the state errors remain within the filter's covariance bounds. Fig. 6 shows the mean 3σ filter covariance bounds over all 50 Monte Carlo runs, since the filter covariance values differ between runs. The covariance of the estimation error values is shown in red.

Table III explores the effect of the value of c on estimation accuracy. The position RMSE at time step k is:

$$\mathrm{RMSE}(k) = \sqrt{\frac{1}{N_m}\sum_{i=1}^{N_m}\|\mathbf{p}_k^i - \hat{\mathbf{p}}_k^i\|_2^2} \tag{28}$$

where N_m is the number of Monte Carlo runs. The position RMSE values in Table III are averaged over all time steps k.

While the estimator performs reasonably well without phase candidate rejection, it is clear that phase candidate rejection improves the NeuroGrid measurement by removing additional "noise" in the grid sum from grid images that are out of phase with the true position. Further, phase candidate rejection reduces computation time. Grid images are never computed from rejected phase candidates, saving time in the calculation of the grid sum. In fact, only about half of the grids are needed to compute a grid sum that provides an acceptable position measurement. The phase candidate dictionary contains 50 grids. On average, only about half the grids are needed to compute the grid sum. In all cases where measurement rejection was turned on, the mean number of rejected grids over all time steps and all 50 Monte Carlo runs was 24.4.

Finally, a note on memory usage: The DEM used to build the phase candidate dictionary and simulate measurements for this example is a 1600×1600 matrix that encodes elevation values in a 400×400 m section of the U.T. campus. If each elevation value is a 16-bit floating point number, then the the memory required to store the DEM onboard would be

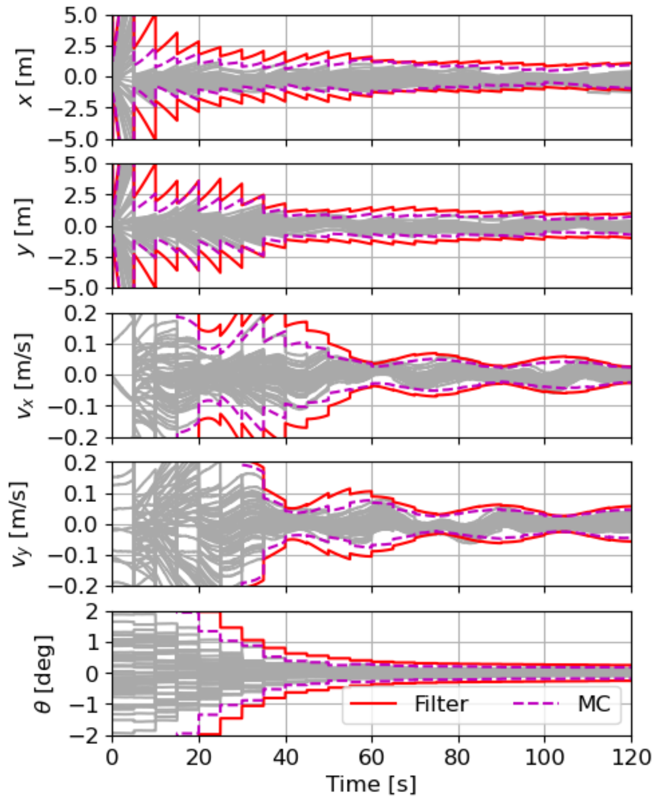

Fig. 6. Estimation error results (gray), mean 3σ covariance bounds (red), and Monte Carlo 3σ covariance (magenta, dashed).

TABLE III
ONLINE COVARIANCE COMPUTATION RESULTS

Meas. Reject.	c	Mean posn. RMSE [m]
OFF	0.9950	1.438
ON	0.9950	0.7881
ON	0.9990	0.6812
ON	0.9995	0.6795
ON	0.9999	0.7176

$1600 \times 1600 \times 16$ bits, or about 5.1 MB. The phase candidate dictionary contains phase candidates for 50 grids and 630 elevation bins. Each phase candidate is a 30×30 binary matrix. The size of the phase candidate dictionary is $50 \times 630 \times 30 \times 30$ bits, or about 3.5 MB. Thus, the phase candidate dictionary reduces the memory requirement by about 30%.

However, NeuroGrid still computes a floating-point matrix the size of the DEM at each measurement time. This is the grid sum, whose maximum value is used to compute the position measurement. A lower-quality grid sum could theoretically be computed, but ultimately the position measurement is only as precise as the resolution of the grid sum. Practitioners therefore face a tradeoff between measurement precision and memory usage at runtime.

V. Conclusion

Taking inspiration from navigation in animals, the NeuroGrid algorithm computes a position "measurement" with respect to a DEM using data from elevation measurements. NeuroGrid is not a neural network. Instead, it uses a data structure called a phase candidate dictionary to encode information that relates elevation values to distinct vehicle positions. We have demonstrated that, in principal, the phase candidate dictionary can be smaller in size than the DEM. Further, the NeuroGrid position measurement was shown to be compatible with an EKF using a simple online covariance computation. Computation time can also be reduced by rejecting grids with noisy phase candidate sums at measurement time; not every grid will have clear spatial phase values $(\phi_{x,i}, \phi_{y,i})$ for every measurement. Our measurement rejection scheme has also been shown to improve estimation accuracy.

This work demonstrates the feasibility of using position measurements generated by NeuroGrid in an EKF. For the results presented here, we rely on a raw magnetometer measurement to convert the body-frame position values of the elevation measurements to the inertial frame. In the future, we would like to incorporate auto-focusing; starting with a noisy heading measurement, we will recompute grid images with different heading values close to the measured value until we find the highest-quality grid sum. We would also like to incorporate state estimation into the phase estimates themselves. Since we know the dynamics of the vehicle, we should be able to propagate the phase estimates along with the vehicle states. This way, instead of carrying a phase candidate dictionary with many grids and rejecting half of them at measurement time, we could potentially maintain a smaller number of grids with much more accurate phase estimates.

References

[1] F. Wang, C. Teeter, S. Luca, S. Musuvathy, and J. B. Aimone, "Localization through grid-based encodings on digital elevation models," in *ICONS*, 2022.

[2] M.-B. Moser, D. C. Rowland, and E. I. Moser, "Place cells, grid cells, and memory," *Cold Spring Harbor perspectives in biology*, vol. 7, no. 2, p. a021808, 2015.

[3] T. Solstad, E. I. Moser, and G. T. Einevoll, "From grid cells to place cells: a mathematical model," *Hippocampus*, vol. 16, no. 12, pp. 1026–1031, 2006.

[4] I. R. Fiete, Y. Burak, and T. Brookings, "What grid cells convey about rat location," *Journal of Neuroscience*, vol. 28, no. 27, pp. 6858–6871, 2008.

[5] M. Lewis, S. Purdy, S. Ahmad, and J. Hawkins, "Locations in the neocortex: a theory of sensorimotor object recognition using cortical grid cells," *Frontiers in neural circuits*, vol. 13, p. 22, 2019.

[6] M. Klukas, M. Lewis, and I. Fiete, "Flexible representation of higher-dimensional cognitive variables with grid cells," *BioRxiv*, p. 578641, 2019.

[7] D. Bush, C. Barry, D. Manson, and N. Burgess, "Using grid cells for navigation," *Neuron*, vol. 87, no. 3, pp. 507–520, 2015.

[8] C. Barry, R. Hayman, N. Burgess, and K. J. Jeffery, "Experience-dependent rescaling of entorhinal grids," *Nature neuroscience*, vol. 10, no. 6, pp. 682–684, 2007.

[9] W. Dorrell, P. E. Latham, T. E. Behrens, and J. C. Whittington, "Actionable neural representations: Grid cells from minimal constraints," *arXiv preprint arXiv:2209.15563*, 2022.

[10] J. P. Golden, "Terrain contour matching (tercom): a cruise missile guidance aid," in *Image processing for missile guidance*, vol. 238, pp. 10–18, SPIE, 1980.

[11] L. Hostetler and R. Andreas, "Nonlinear kalman filtering techniques for terrain-aided navigation," *IEEE Transactions on Automatic Control*, vol. 28, no. 3, pp. 315–323, 1983.

[12] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Sensor fusion IV: control paradigms and data structures*, vol. 1611, pp. 586–606, Spie, 1992.

[13] R. E. Gold, S. G. Catalan, B. A. Jones, and R. Zanetti, "Extending capabilities of crater navigation and timing for autonomous lunar orbital operations," in *Space Imaging Workshop*, 2022.

[14] S. G. Catalan, J. S. McCabe, and B. A. Jones, "Implementation of machine learning methods for crater-based navigation," in *AAS/AIAA Astrodynamics Specialist Conference*, 2021.

[15] Z. R. McLaughlin, R. E. Gold, S. G. Catalan, R. Moghe, B. A. Jones, and R. Zanetti, "Crater navigation and timing for autonomous lunar orbital operations in small satellites," in *44th Annual AAS Guidance, Navigation, and Control Conference*, 2022.

[16] L. Downes, T. J. Steiner, and J. P. How, "Deep learning crater detection for lunar terrain relative navigation," in *AIAA SciTech 2020 Forum*, p. 1838, 2020.

[17] Driver, Travis*, Tomita, Kento*, K. Ho, and P. Tsiotras, "Deep monocular hazard detection for small body landing," in *AAS/AIAA Space Flight Mechanics Meeting*, pp. 1–17, 2023. *These authors contributed equally to this work.

[18] R. Moghe and R. Zanetti, "A deep learning approach to hazard detection for autonomous lunar landing," *The Journal of the Astronautical Sciences*, vol. 67, no. 4, pp. 1811–1830, 2020.

[19] A. Scorsoglio, A. D'Ambrosio, L. Ghilardi, B. Gaudet, F. Curti, and R. Furfaro, "Image-based deep reinforcement meta-learning for autonomous lunar landing," *Journal of Spacecraft and Rockets*, vol. 59, no. 1, pp. 153–165, 2022.

[20] Sensonor, *Ultra-high performance inertial measurement unit (IMU): STIM300 Product Brief*, Aug 2017.

[21] Bosch, *Data sheet: BMM150*, Apr 2020. Rev. 1.4.

[22] R. S. Bucy and P. D. Joseph, *Filtering for stochastic processes with applications to guidance*, vol. 326. American Mathematical Soc., 2005.

[23] P. Passalacqua, "Austin, tx, rapid response, 2015 airborne lidar survey," 2015.