MODELING, GRIDDING & STORAGE OF EFFECTIVE FIELDS OF VIEW FOR TERASCALE, POINT-BASED PLANETARY DATASETS: CASE STUDY – LRO DIVINER

E. Sefton-Nash[1], J-P. Williams[1] and D. A. Paige[1], [1]Department of Earth and Space Sciences, University of California Los Angeles, 595 Charles Young Drive East, Box 951567, Los Angeles, CA 90095-1567, USA (esn@ucla.edu).

Introduction: Point based planetary datasets are typically stored as discrete records that represent an infinitesimal location on the target body. Instrumental effects and spacecraft motion during integration time can cause single points to inadequately represent the total area on the target that contributes to an observation. Production of mapped data products from these data for scientific analysis proceeds by binning points onto rectangular grids. Empty bins occur where data coverage is insufficient relative to grid resolution, and remedial interpolation can lead to high uncertainty areas and artifacts in maps.

To address such issues and make better use of available data, we present a method to calculate the effective field of view (EFOV) for point-based datasets, using knowledge of instrumental characteristics and observation geometry. We demonstrate a fast minimum-storage approach to store, access and produce gridded data products from a geometrically corrected database of EFOVs.

We apply this approach to data from the Diviner Lunar Radiometer Experiment [1], a visible to far-infrared radiometer aboard NASA's Lunar Reconnaissence Orbiter, which since July 2009 has returned > $2x10^{11}$ observations of the Moon in 9 spectral channels between 0.35 – 400μm, forming a raw point dataset > 20 TB in size.

Calculating effective field of view (EFOV): The total contribution to an observation may be defined by (i) the in-track time broadening due to spacecraft motion relative to the target body, *b(t)* (ii) the detector response as a function of time, *d(t)* and, (iii) the instrument's instantaneous field of view (IFOV), *f(t)*. Knowing the shape of these functions, for each observation we may construct a 2D probability distribution in the focal plane of the detector – equal to a convolution of the component functions evaluated over the integration time: $[(b * d) * f](t)$ (Fig. 1). This EFOV describes the likelihood of quanta e.g. photons, being incident on the detector within an observation's time-integrated footprint.

In order to produce mapped data products, the probability function must be discretized into points so that they may be projected onto a Digital Elevation Model (DEM) of the target body (assuming there is knowledge of the observation geometry), and subsequently binned onto regular grids.

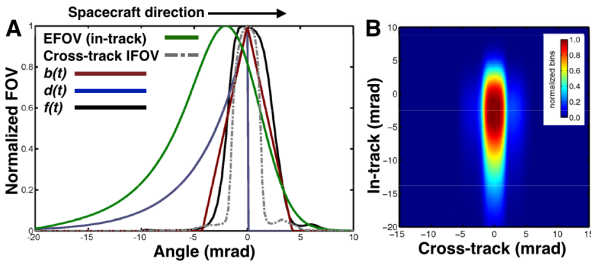

Figure 1: A - Component functions. B - Convolved over cross- and in-track directions to form 2D probability distribution for one EFOV.

This is accomplished by populating the EFOV with a Monte Carlo distribution comprising *n* points, where the weight of each point, $w = 1/n$ observations. *n* must be sufficient to adequately resolve the EFOV, which may have a non-trivial shape, perhaps due to low probability wings or non-nadir, secondary maxima in the IFOV (e.g. dashed-dotted line in Fig. 1). The modeled EFOV therefore tends to the actual EFOV with increasing *n*.

Efficient storage of an EFOV database: However, populating the EFOV with a Monte Carlo approach and projecting onto the DEM is computationally intensive because the number of points is $n \times$ the number of observations. It is therefore desirable to perform this step only once, rather than for each time an observation is included in a mapped data product. The data volume increase in modeling EFOVs in this way is by a factor of *n*, therefore techniques to efficiently store, access and create mapped data products from EFOVs must be employed.

The dimensions of the central EFOV maximum dictate the maximum useful spatial resolution of gridded products. To save disk space, EFOV points are binned onto a grid with a bin size that approximates the area containing a large majority (in this case ~90%) of the EFOV. An equal-area grid is required, because spherical coordinate (*lat,lon*) approaches are not equal-area and suffer from reduction in bin size with distance from the equator [2].

We implement a fast minimum-storage scheme to bin data points onto a hierarchical triangular grid that approximates a sphere. The scheme (Fig. 2) operates as follows: 1) Define a starting icosahedron, similar to the approaches of [2,3]. 2) Calculate parametric coordinates relative to each triangle for a ray passing from the origin of the target body through the data point [4] 3) The data point lies in the triangle where $0 \leq$ para-

metric coordinate < 1. 4) Subdivide the triangle and test for intersection on each triangle in the new sub-grid. 5) Recursively perform steps 2—5 until the desired grid spatial resolution is reached, i.e. triangle area ≈ 90% of EFOV.

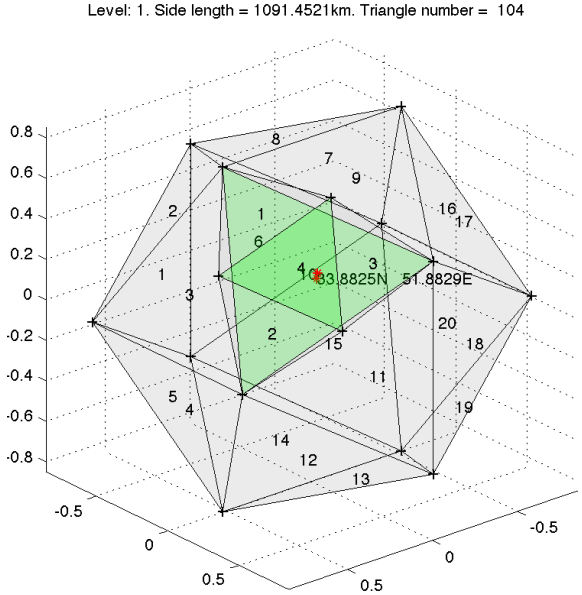

Figure 2: Level 0 icosahedron & level 1 grid. The icosahedrons' triangle 10 intersects with the point (red asterix) and so is divided into a sub-grid of 4 level 1 triangles. The process iterates until the desired spatial resolution is reached. At each subdivision, new vertices are normalized to a sphere (origin at [0,0,0]).

The grid 'level' refers to the number of times that triangles are divided into 4 sub-triangles. In this case LRO's orbit and Diviner's observation geometry yeild a best-case EFOV spatial resolution that is similar in size to triangles in a level 14 grid (Fig. 3).

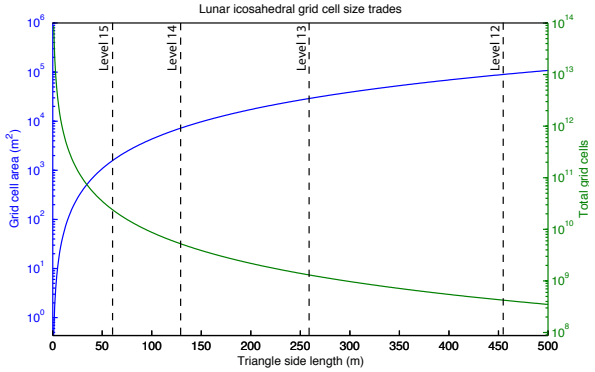

Figure 3: A calculation used to determine appropriate grid level: Grid cell area and total grid cells vs triangle side length for the lunar reference sphere (R_M=1737400 m).

Within one observation, EFOV points that fall within the same triangle are grouped together. Fields that vary due to projection onto the DEM (such as lat., lon. or phase angle) are averaged within in each triangle, except for weight, which is summed. The resulting data points sample the EFOV probability function for every triangular grid cell that falls in an observations' footprint. The weight of each data point represents its fraction of the original observation (Fig. 4). It should be noted that knowledge of the total triangular grid is not required for binning, since sub-grids are calculated recursively as needed. Our approach therefore negates pre-calculation and storage of large triangle meshes, and is computationally parallelized.

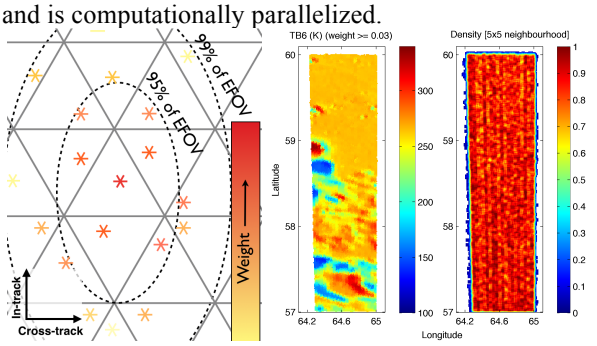

Figure 4: Left - Schematic of EFOV points binned onto triangular grid. Right - Map of Diviner ch. 6 brightness temperature and its density mask, produced using data from EFOV database.

For further reduction in data volume, records are separated into two linked lists. The first contains fields that are invariant for each observation, while the second contains those that vary within each observation, such as data point weight or geometric parameters. Linked lists are stored as tables in HDF5 files with 'blosc' compression [5]. Files are grouped by commonly queried fields such as latitude, channel number or date.

Retrieval of data is performed using a simple query tool. When binning EFOV data points onto rectangular grids, a point density mask is produced. Where there are empty grid cells but the point density exceeds a pre-specified threshold, it is deemed safe to interpolate over the empty cells. This restricts excessive interpolation in areas of poor data coverage, reducing uncertainties and artifacts in maps (Fig. 4 - right).

Summary: We present a technique to store and access geometrically corrected and EFOV-modeled large point-based planetary datasets. We envisage a future release of the code to the public domain for use in planetary remote sensing.

References: [1] Paige, D. A. et al. (2010), Space Sci. Rev., 150, 125–160. [2] Teanby, N. A. (2006), Computers & Geosciences 32, 1442-1450. [3] Massey, N. (2012), Computers & Geosciences 44, 42-51. [4] Möller, T. and Trumbore, B. (1997), J. Graphics Tools 2(1), 21-28. [5] Alted, F. and Vilata, I. (2002), PyTables.