Data Management Possibilities for Aperture 3 Hexagonal Discrete Global Grid Systems

Ali Mahdavi-Amiri
Troy Alderson
Faramarz Samavati

August 23, 2016

Abstract

In a Digital Earth framework, data sets are gathered from different sources in three main forms: imagery/elevation, vector, and quantitative data sets. In order to efficiently work with these data sets in a Digital Earth framework, effective methods to represent and transmit these data sets are required. While these representations may be different for each type of data set, they must all preserve the actual data as much as possible in order to accurately address queries. Furthermore, they also need to be compatible with the underlying structure of the Digital Earth framework. In this paper, we describe several data representations for an Aperture 3 Hexagonal Discrete Global Grid System which is a common approach to build a Digital Earth framework. We also discuss how they can be used to transmit data sets or address specific queries.

1 Introduction

The Digital Earth provides a representation of the Earth on which data sets of different types and sources are integrated, analyzed and visualized [17]; and is commonly implemented using a Discrete Global Grid System (DGGS) [33]. In a DGGS, the Earth is typically approximated by a spherical polyhedron that is refined by a specific factor (or aperture) in order to provide the multiresolution property [17]. In this process, the surface of the Earth is discretized into a set of cells that are then projected to the sphere. The projection that is used for a Digital Earth framework can be of different types, but an equal area projection is usually more desirable when data analysis is emphasized [45, 18].

One common DGGS is a hexagonal DGGS that is refined by factor of three (Aperture 3) due to its smooth transition among resolutions [29, 40, 19]. Cells in this Aperture 3 hexagonal (A3H) DGGS are nearly all hexagonal, with the exception of non-hexagonal cells located at the initial vertices of the underlying polyhedron of DGGS (See Figure 1). Among polyhedrons that can create such a DGGS, the icosahedron is more desirable

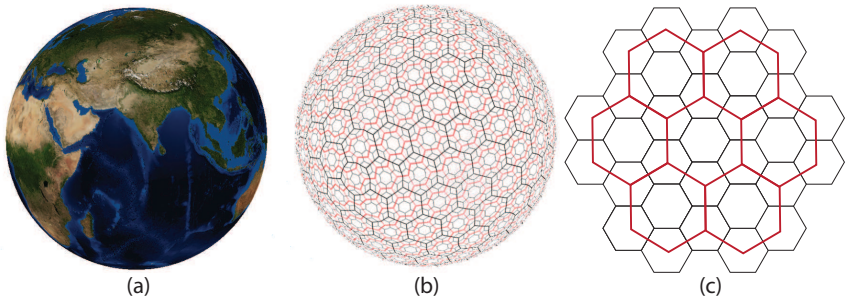


Figure 1: (a), (b) PYXIS Digital Earth that is an A3H DGGS and its cell structure. (c) The 1-to-3 hexagonal refinement (aperture 3).

as it produces less distortion in comparison with other platonic solids [29, 40, 19, 32]. However, the methods that we proposed in this paper is applicable to all types of A3H DGGS independent from its underlying polyhedron.

To assign data sets to the cells and analyze or visualize them at different resolutions, a data structure is needed that can support a multiresolution hierarchy among the cells. Furthermore, in order to assign and retrieve data sets to/from these cells, they need to be indexed. There are three main types of indexing: hierarchy-based, coordinate-based, and space filling curve (SFC) based [24]. Two types of indexing have been used in an A3H DGGS: one hierarchy-based and one coordinate based.

For the hierarchy-based indexing system on hexagonal cells of an A3H DGGS, a hierarchical relationship among the cells is needed [29, 40, 32]. Since hexagonal refinements are not congruent, this hierarchical relationship is not very straightforward to define. PYXIS hierarchy is one possibility to define such a hierarchical relationship between the cells at different resolutions [29, 40]. In PYXIS hierarchy, cells are categorized into two types — A and B — generating different fractal shapes called *tiles* throughout the resolutions that fit together and cover the entire surface of the spherical icosahedron (see Figure 2). Type B cells with index b have children with indices b_i ($0 \leq i \leq 6, i \in N$) while a type A cell has only one child with index a_0. a_0 and b_0 are considered to be of type B, while the other b_i cells are of type A [19, 40].

Hierarchy based indexing systems are efficient at addressing hierarchical queries, but neighborhood queries (i.e. finding the neighbors of a cell) cannot be performed in constant time and have $O(r)$ time complexity (where r is the resolution). To overcome this issue, a second indexing system in an A3H DGGS cab be defined based on a coordinate system defined on the hexagonal cells [19]. This coordinate system is based on the duality relationship between hexagons and triangles — the centroids of the hexagons are connected to form triangles, and pairs of triangles are combined into diamonds [20]. Hence, each vertex of each diamond corresponds to the centroid of a hexagonal cell (see Figure 2). The axes of the coordinate system are defined such that they align with the edges of the diamonds,

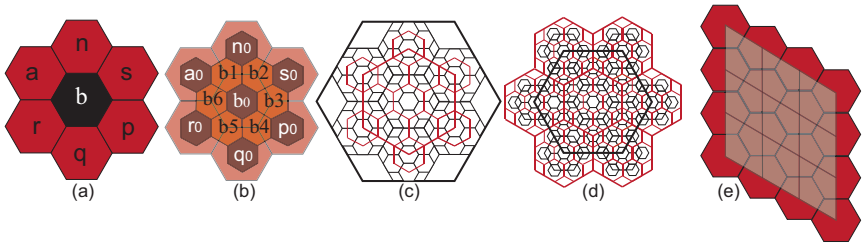


Figure 2: (a), (b) PYXIS hierarchical indexing method. (c), (d) Fractal tiles created from type A and B cells. (e) A diamond covering a set of hexagonal cells. The vertices of the refined diamond are aligned with the centroids of the hexagonal cells.

and, as a result, the vertices of the diamonds may be indexed using these axes.

These two mechanisms of arranging cells are the main systems for addressing and organizing data sets in A3H DGGS such as PYXIS framework. In this paper, we discuss how we can use these structures to manage and work with large-scale data types of three forms: imagery/elevation, vector, and quantitative data.

2 Related Work

Digital Earths, their applications, and the different methods used to build, visualize, and index them have been studied extensively in the literature [17]. One important application of Digital Earth is in data analysis. There are three important types of data sets for Digital Earths that we consider in this paper: imagery/elevation, vector, and quantitative data. In the following, we describe each of these data sets and note some of the methods that address analysis and representation in geospatial applications.

Imagery and Elevation Data sets: Geospatial imagery data sets are often beneficial to the visualization and analysis of locations. They are often used as textures of the Earth's cells or used to provide special views of the Earth (e.g. spherical panoramic views). Data assignment to the cells of a DGGS is most commonly performed via rasterization, where each cell of the DGGS is treated as a pixel-like entity, and attributes such as color or height can be assigned to these cells [32].

Data sets in Digital Earth frameworks are visualized at different resolutions, therefore a multiresolution representation for images is needed, for which different methods exist. Mip-mapping is a common approach that provides such a multiresolution representation. A mip-map is a pyramidal structure consisting of progressively lower resolution images of the initial given image [46]. In geospatial visualization, mip-mapping has been used in [11] to establish a continuous transition between different correlated images. In addition to mip-mapping, wavelet transforms are also powerful tools for providing a multiresolution representation of images [39]. A simple and useful wavelet transform is the Haar wavelet,

in which two neighboring pixels are replaced by their average (a coarse pixel) once row-wise and once column-wise. The difference between the average and one of the pixels is called a detail or wavelet. Using the details and the coarse pixels, it is possible to perfectly reconstruct the image. However, for the representation of images, only integer values are allowed. Due to truncation/rounding error, perfect reconstruction may not possible without special consideration. As a result, integer wavelets were proposed to solve this issue [44].

In Section 3, we first describe how we rasterize imagery data sets in an A3H DGGS. Since the Haar wavelet is designed for 1-to-4 refinements, it cannot be used directly in an A3H DGGS. We then describe how to extend Haar wavelets to A3H DGGS and describe how we can avoid truncation error by using integer wavelets

Elevation data sets are often very similar to images, as Data Elevation Models are uniform grids with values for heights instead of colors. As a result, their rasterization and wavelet transforms are very similar to those employed for images. The only difference is that there is no need to use integer wavelets for elevation data sets, as they are typically represented as floating point numbers.

Vector Data sets: Vector data sets are available in the forms of points, polylines, or polygons. These points are usually connected by spherical or ellipsoidal arcs and represent region boundaries such as roads, rivers, continents, cities, etc. Vector data can be produced via ground surveying, LIDAR, and photogrammetry [16]. Features in LIDAR data sets can be detected and vectorized using different techniques [3, 31, 27, 6], although imagery data alone can also be used to extract vector data sets [27, 7, 37, 8, 26, 4].

In order to store vector data in a DGGS, either rasterization or a cell-based approach can prove useful [32]. Under the rasterization approach, the vector can be represented as an ordered set of cells, where each cell corresponds to and contains one of the vectors vertices/points. Treating the cells of the DGGS as "buckets" allows data storage techniques similar to those used by quadtrees to be employed.

Three main methods are available to visualize vector data sets: texture-based, geometry-based, and shadow volume-based [48]. Texture-based approaches rasterize the vector data into textures that are then mapped onto the terrain surface [12]. Geometry-based approaches consider the geometry of the vector data separate from the geometry of the terrain, and the vector data or terrain data may be modified for consistency [36, 35, 1, 30, 1, 43]. Shadow volume-based approaches extrude the vector geometry into polyhedrons that are rendered via the stencil buffer to distinguish visible and invisible parts of the scene [13].

In Section 4, we describe both the cell-based and rasterization approaches in an A3H DGGS. We then describe how we use texture-based and geometry-based approaches for visualization, other queries such as data transmission, and buffering.

Quantitative Data sets: Quantitative data sets usually provide statistical information related to locations. These statistical data can be environmental, biological, or demographic in nature (e.g. average income of a particular location). They are usually collected by sampling regions through various means (e.g. surveys or sensors) [47] and are

assigned to the cells of a DGGS as attributes related to that particular location.

When the volume of such data sets gets very large and queries become complicated, it is important to simplify the data and still address the query reasonably well. To do so, wavelets can be again a powerful method [15, 41, 5, 42, 9]. Wavelets have been used as a component of data cubes, in which an estimation of the data is built by forming a hierarchical tree using wavelets [42, 25]. In Section 5, we describe this approach in detail and discuss its advantages and disadvantages when it is used in an A3H DGGS. We also provide a novel method for data transmission in which we can send an estimate of the data with a known error.

3 Imagery and Elevation Data Sets

Imagery and elevation data sets are important to any Digital Earth, and hence need to be integrated, analyzed, and visualized efficiently. In an A3H DGGS, cells are hexagonal and store information about the region each encompasses. This includes imagery or elevation data, which must be assigned to these cells. Since imagery and elevation data sets are represented similarly, we only describe the representation of imagery data sets here.

To facilitate the assignment of imagery data sets to hexagonal cells, they can be converted to triangular cells using simple hierarchical conversions (described in [20]). The basic idea behind this conversion is to use a dual conversion, by forming triangular cells by connecting the centroids of neighboring hexagonal cells. Therefore, each vertex of a diamond corresponds to a hexagonal cell. When these triangular cells are created, they are packed into congruent quadrilaterals, or diamonds, that can be used to simply sample available square images (e.g satellite imagery) and they can also be rendered efficiently on the GPU [19, 23, 21]. Figure 3 illustrates the diamonds that are formed using the dual conversion of hexagons. Note that the diamonds can be at different resolutions sampling hexagonal cells at different resolutions, but they are connected by a set of "zippers" to avoid gaps and cracks in the final surface of the Earth [22].

Given this conversion of hexagonal cells into diamonds compatible with standard sampling and rendering techniques, we can assign the imagery data sets to hexagonal cells. These images can be obtained from any source of geospatial imagery, so long as their location attributes are available. As an example, consider Microsoft's Bing Maps library of satellite images. Since Bing Maps and an A3H DGGS provide two different multiresolution representations of the Earth, a correspondence between the resolutions of these two representations must be constructed in order to select the correct images for cells in an A3H DGGS. This correspondence can be generated by considering the number of cells that these two multiresolution representations provide at specific resolutions (see [24] for more details on converting between two Digital Earth frameworks).

Once the correct resolution of the Bing Map texture to capture has been determined, we can obtain the image related to each cell. In order to assign, to each cell, imagery data

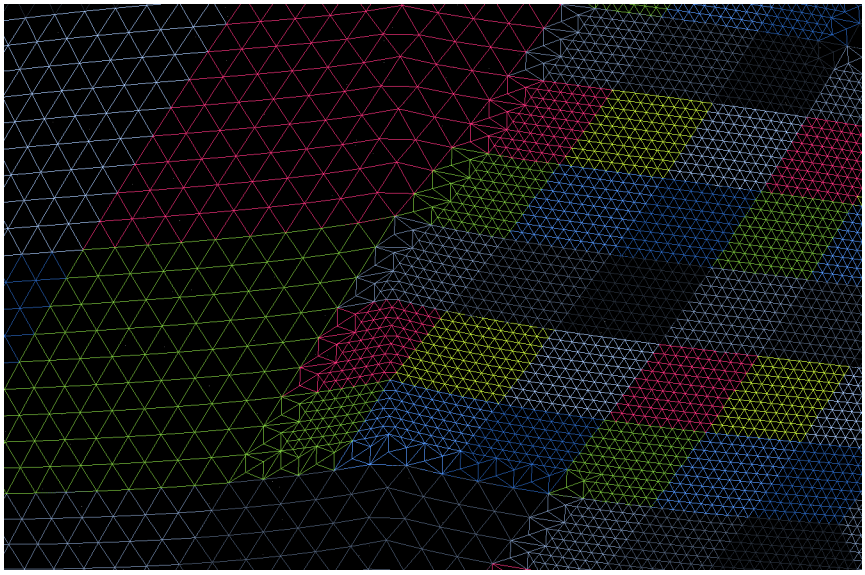


Figure 3: A zoomed portion of the Earth rendered using diamonds.

from Bing Maps, we sample along the sides of the diamond at each vertex, request images related to the samples from Bing Maps, and assign that image to the corresponding cell. Figure 4 shows an example of sampling the textures using diamonds.

In an A3H DGGS, imagery data sets are usually sampled and assigned on a server and sent to clients. However, sending the entire data set may not be efficient due to the potentially large volume of the data. As a result, a mechanism is needed to gradually send the data and improve it through the time. Wavelets can be very useful for this purpose. One of the simplest, yet efficient, wavelet transforms is the Haar wavelet. In this wavelet transform, low resolution data c_i are built by averaging two consecutive high resolution data values f_{2i} and f_{2i+1} in a process called *decomposition*. The details corresponding to the decomposition are found as the difference between the low resolution data and the high resolution data $d_i = c_i - f_{2i}$. Using the low resolution data and details, the high resolution data can be reconstructed as $f_{2i} = c_i - d_i$ and $f_{2i} = c_i + d_i$ (see Figure 5). As is readily apparent, the dimension of the data needed to reconstruct the high resolution data is the same as that of the high resolution data set (i.e. no additional information/storage space is needed).

Since the dimension of the low resolution data is half of the high resolution data, the Haar wavelet is binary. However, this transform is not readily compatible with an A3H DGGS, as a 1-to-3 refinement is employed. Two iterations of the 1-to-3 refinement, however, provides a ternary refinement in which the number of cells is tripled along the two main axes defined for cells (see Figure 6). As a result, we suggest a ternary version of Haar that is compatible with an A3H DGGS[28].

Figure 4: (a) A textured globe using diamonds. One of the diamonds has been textured differently to show the formation of diamonds. (b) Two textured diamonds beside each other. Red lines highlight the boundary of the two diamonds. (c) A close up look of the hexagonal sampling of textures.

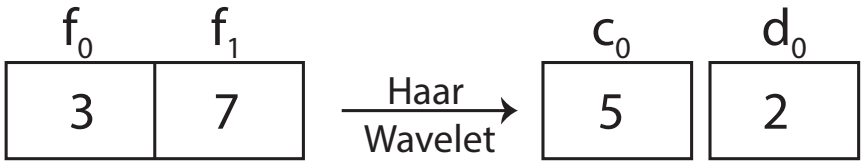


Figure 5: Two fine points f_0 and f_1 are averaged to get c_0 in the Haar wavelet transform. The difference between f_0 and c_0 is the detail d_0.

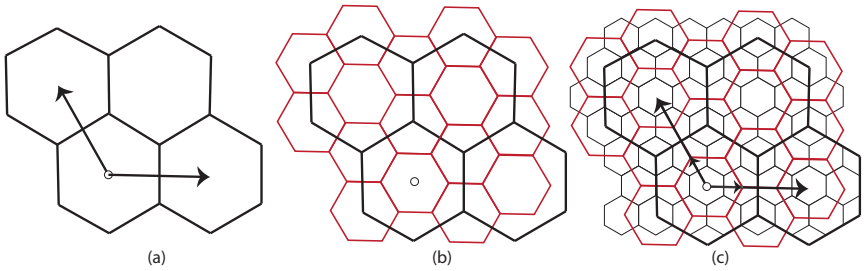


Figure 6: (a) A set of hexagons and a coordinate system defined for these hexagons. (b) The coarse hexagons in (a) are refined using 1-to-3 refinement. (c) After two iterations of 1-to-3 refinement, the fine hexagons are aligned with the coarse hexagons in (a).

Figure 7: Applying the ternary Haar wavelet to an image two times.

In the ternary Haar wavelet transform, low resolution data c_i are built by averaging three consecutive high resolution data values f_{3i}, f_{3i+1}, and f_{3i+2} in the decomposition process, i.e. $c_i = \frac{f_{3i}+f_{3i+1}+f_{3i+2}}{3}$. To perfectly reconstruct the high resolution data, we associate two details d_{2i} and d_{2i+1} with c_i, instead of just one, that are defined as $d_{2i} = c_i - f_{3i}$ and $d_{2i+1} = c_i - f_{3i+1}$. In the reconstruction process, high resolution data sets are reconstructed via $f_{3i} = c_i - d_{2i}$, $f_{3i+1} = c_i - d_{2i+1}$, and $f_{3i+2} = c_i + d_{2i} + d_{2i+1}$.

Note that in the Haar wavelet transform, if the high resolution values are integers, there is no guarantee that the obtained low resolution values will also be integers due to the averaging process. While this is not a problem for elevation data (as such data are naturally available in the floating point format), imagery data sets are represented in integer format. Therefore, when applying the Haar wavelet transform, we need to truncate the data in order to obtain integer values. In this scenario, however, we cannot perfectly reconstruct the high resolution data due to truncation error. To avoid this problem, we can employ integer ternary Haar wavelets, in which high resolution data, low resolution data and details always remain integers.

To define an integer ternary Haar wavelet, we modify the ternary Haar wavelet in such a way that all details and low resolution data sets become integer values. In the ternary Haar wavelet, we have $d_{2i} = c_i - f_{3i}$ and $d_{2i+1} = c_i - f_{3i+1}$. If we substitute $c_i = \frac{f_{3i}+f_{3i+1}+f_{3i+2}}{3}$ into these relations, we get $d_{2i} = \frac{-2f_{3i}+f_{3i+1}+f_{3i+2}}{3}$ and $d_{2i+1} = \frac{f_{3i}-2f_{3i+1}+f_{3i+2}}{3}$. Since c_i might not be an integer, instead of saving c_i, we save $\tilde{c}_i = \lfloor c_i \rfloor$. In this case, d_{2i} and d_{2i+1} are also not integers, so instead of d_{2i} and d_{2i+1}, we save $\tilde{d}_{2i} = -2f_{3i} + f_{3i+1} + f_{3i+2}$ and

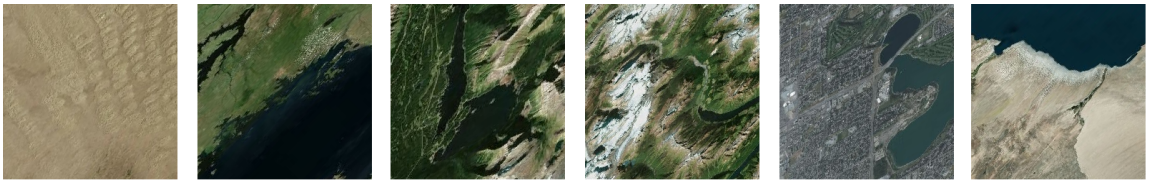


Figure 8: Test pictures used in Table 1.

$\tilde{d}_{2i+1} = f_{3i} - 2f_{3i+1} + f_{3i+2}$. In the reconstruction process, f_{3i} and f_{3i+1} are reconstructed quite simply as $f_{3i} = \tilde{c}_i - \left\lceil \frac{\tilde{d}_{2i}}{3} \right\rceil$ and $f_{3i+1} = \tilde{c}_i - \left\lceil \frac{\tilde{d}_{2i+1}}{3} \right\rceil$. Now, given f_{3i}, f_{3i+1} and $\tilde{d}_{2i}$, f_{3i+2} is reconstructed as $\tilde{d}_{2i} + 2f_{3i} + f_{3i+1}$. Using this wavelet transform, we can compress imagery data sets in a manner compatible with an A3H DGGS (see Figure 7).

To examine the behavior of our integer ternary Haar wavelet, we compared the Peak Signal to Noise Ratio (PSNR) of our integer ternary Haar wavelet with the PSNR of the integer binary Haar wavelet. In both cases, we only used eighty percent of the original data for reconstruction. The comparison shows that the integer ternary Haar wavelet is comparable with integer binary Haar wavelet and PSNR of both methods are very close. Table 1 provides the PSNR of both methods for images listed in Figure 8 for the three red, green, and blue channels.

Table 1: PSNR for images in Figure 8 for integer ternary and binary Haar wavelets. The tree rows in front of each method provides the PSNR of each red, green, and blue channels.

Method	a	b	c	d	e	f
Ternary	34.2025	29.8641	25.4580	22.9446	25.9309	29.3136
	34.1923	29.8623	25.6380	22.7123	25.9668	29.3894
	34.1824	30.2141	25.7534	22.9040	25.9336	29.3652
Binary	36.8128	32.7858	28.6623	25.9970	28.6422	32.2297
	36.8058	32.7972	28.7183	26.0223	28.6733	32.2598
	36.7980	33.0755	28.8164	26.0532	28.6650	32.2405

4 Vector Data Sets

Vector (or feature) data sets are defined as poly-lines, points, or polygons that describe geospatial features such as road networks or boundaries of countries and cities. To visualize these data sets in a Digital Earth, typically the points making up these vectors are mapped to the sphere and connected with geodesic arcs (i.e. great circle arcs), as geodesic arcs traverse the shortest path between two points on the surface of the sphere. Since these data sets maybe very large — consisting, for example, of millions of points — it is

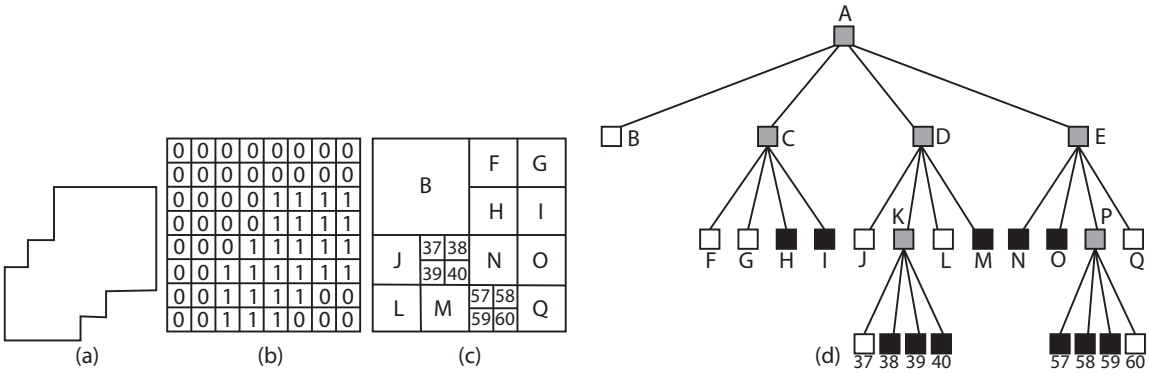


Figure 9: (a) A simple feature. (b) Cells that are inside or outside of the feature are assigned a number, 1 or 0, respectively. (c) Refinement of cells is used to approximate the feature. (d) Trees and coloring of the nodes.

necessary to have a representation of these data sets that supports the efficient handling of relevant queries, such as buffering or data transmission. In the context of a DGGS, such a representation can be provided by association with cells at different resolutions, by rasterizing the vector into an image, or by using wavelet transforms on the feature curves. Three representations for feature vectors are usually employed: hierarchical cell rasterization, direct rasterization, and spherical representation. In the following, we explain each method to an A3H DGGS and its advantages to some applications.

Hierarchical Cell Rasterization: In cell rasterization, we can use a similar technique to that of [34], in which quadtrees are used to approximate features on an image. Under this method, a quadtree is a set of quad cells that are recursively refined until a good approximation of the feature is obtained. A cell is refined if the feature is partially inside the cell, such as the grey cell in Figure 9. If the feature is fully inside or outside the cell, or if the cell size shrinks past a particular threshold, it is not refined any more.

Similarly, in an A3H DGGS, we can use a hierarchical tree structure to approximate a feature using PYXIS hierarchy. However, the tree created in this manner is naturally more complex due to the lack of congruency between hexagonal cells. While constructing the nodes of the tree is very similar to the quadtree method of [34], the refinement is instead a hexagonal 1-to-3 refinement (see Figure 10). Deciding whether a feature is completely inside or outside a cell is also more challenging, as the children of type A and B cells create a fractal coverage throughout the resolutions. To simplify this decision, the coverage of type A and B cells are approximated using two circles with different radii (see [32] for more details). Figure 11 illustrates the maximum and minimum radii for type B cells. If the feature is completely inside or outside the circle associated with a cell, the cell is not refined anymore. Otherwise, it is considered to be a grey cell and refinement continues.

Direct Rasterization: Another method to represent, store, transmit, and visualize

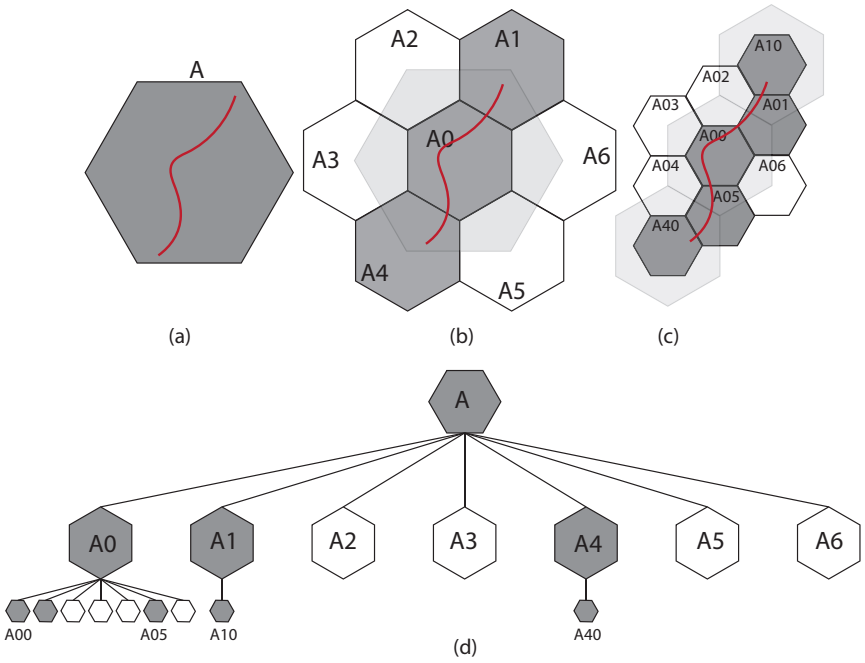


Figure 10: (a) Red feature on a coarse hexagon with index A. (b), (c) Cells are refined to approximate the feature. (d) Hierarchical tree associated with the refinement process.

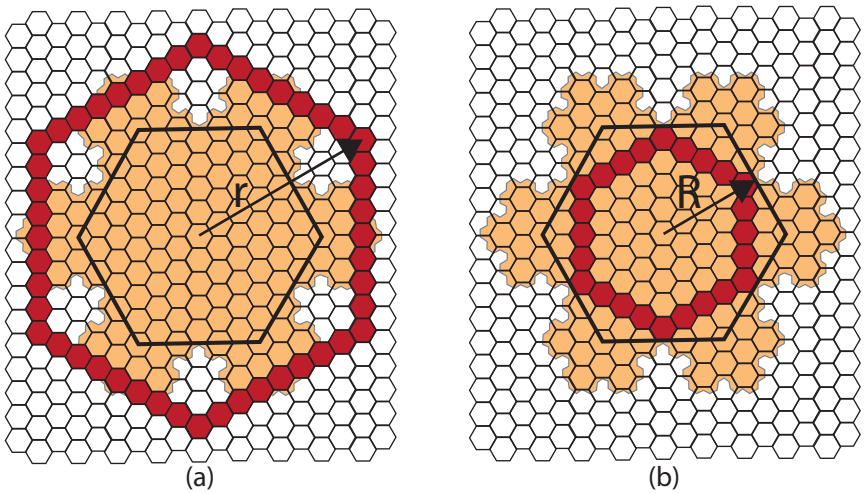


Figure 11: Minimum and maximum radius for the fractal coverage of type B cells.

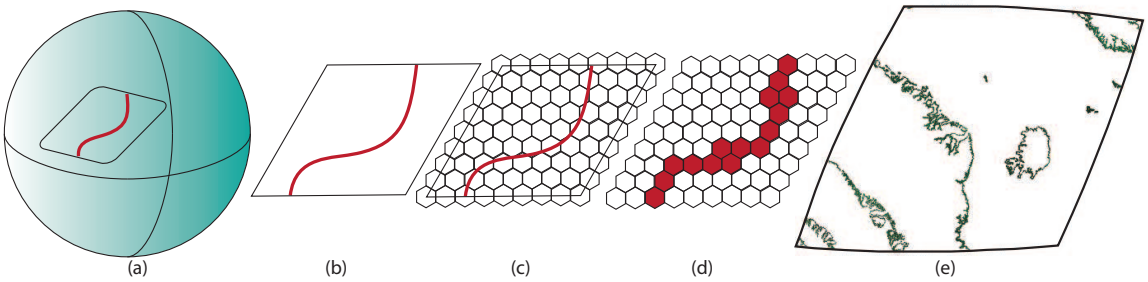


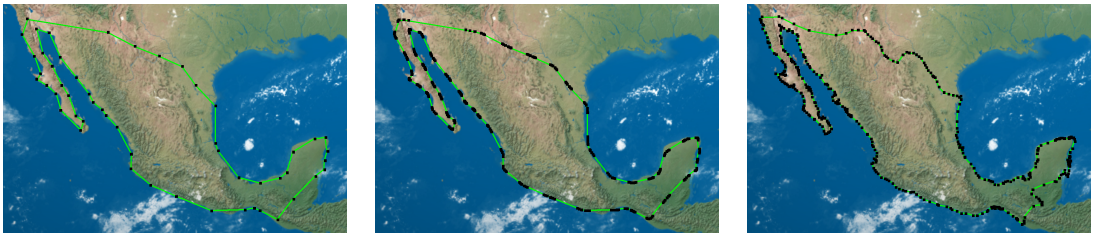
Figure 12: (a) Red feature and the diamond enclosing this feature on the sphere. (b) Projecting the feature and the diamond to the 2D diamond. (c) Sampling the feature using hexagonal cells. (d) Cells sharing a feature are colored. (e) A real example of a rasterized geospatial feature.

vector data sets is to rasterize them into an image . This method is similar to texture-based approaches to attaching vector data on a terrain [12]. These images can then be overlayed on the globe for visualization, and can be effectively compressed and transmitted through the network using known techniques for image compression and transmission [38]. To create these images in the context of A3H DGGS, features are initially projected to diamonds and are then rasterized using the hexagonal cells covering a diamond. Those cells that contain parts of the feature receive a color, while other cells are made to be transparent (see Figure 12).

While the cells representing a feature can usually be made fine enough to provide a good approximation of the feature, one problem with this representation is that the accuracy of the representation for the feature is fixed. Once the image is created using cells with a specific resolution, providing a more accurate representation of the feature requires reacquisition of the whole feature curve and creation of a different image.

Spherical representation: Another approach to represent a feature curve is to provide a multi-scale representation of the curve itself, instead of building a separate hierarchical structure on top of the feature curve. One natural approach is to project the feature curve onto a 2D intermediate domain (e.g. the lat/long domain) and apply a known multiresolution framework on the 2D curve before unprojecting it back to the spherical domain. However, due to distortions inherent in the projection and unprojection processes, this approach does not perform well, particularly for features covering a large area on the Earth [2].

To avoid such artifacts, we use spherical multiresolution frameworks to provide a multi-scale representation of feature curves. In [2], a simple geometric construction for multiresolution representations on the sphere is introduced that is based on a modified Lane-Riesenfeld algorithm [14]. As the construction is composed entirely of SLERP (**s**pherical **l**inear int**erp**olation) operations, it is possible to increase or decrease the resolution of



(a) (b) (c)

Figure 13: Progressive refinement of a geospatial feature curve. (a) Coarse feature curve. (b), (c) Subdivision of the feature in (a), without and with details.

(a) (b)

Figure 14: Adaptive refinement of a geospatial feature curve. (a) Coarse feature curve. (b) The curve can be locally reconstructed to full detail as necessary.

spherical curves without using intermediate domains, and thus avoid any distortions due to intermediate mappings. Furthermore, the schemes resulting from this method achieve perfect reconstruction despite being neither interpolating nor midpoint interpolating (see Figure 13).

As this and other multiresolution techniques typically operate on a local neighborhood, an important property of multiresolution is that it can be applied locally on a feature curve. Hence, one can build a good coarse approximation of the feature and then reconstruct a portion of the curve in full detail on demand. This is very useful for displaying the overall shape of the feature at a distance and a portion of the feature in detail when zoomed in on part of the curve (see Figure 14). Such a multi-scale representation of geospatial vector curves can additionally be beneficial to support a multi-scale representation of buffer/offset curves, finding intersection of features, and data transmission.

5 Geospatial Quantitative Data Sets

Quantitative data sets encompass a variety of data sets that are usually presented as numerical numbers. For instance, the average age of the population living in a region,

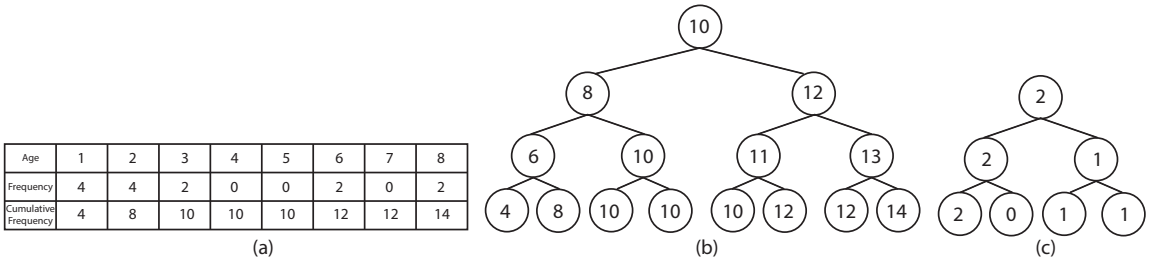


Figure 15: (a) Age data set and cumulative frequency. (b) Haar wavelet histogram. (c) Detail tree associated with the wavelet histogram in (b).

the number of endangered species in a region, and the rates of rain in certain cities are examples of quantitative data sets. One important query when dealing with quantitative data sets is the range query, in which the number of data values within a particular range is requested. To answer such range queries, these data sets need to be managed efficiently. In this section, we discuss how to address these range queries for quantitative data sets in the PYXIS framework.

Wavelet Histogram: The first approach to managing quantitative data sets is wavelet histograms for data cubes [25]. Suppose that we are given a data set outlining how many people are of a certain age — ranging between age one and eight — and we are interested to ask queries such as "How many people are between the ages of one and six?". To answer such queries, in [25], the total number of people having an age less than or equal to a certain age n is calculated for each age n. This number is called the cumulative frequency (CF). Formally, the cumulative frequency of age n is found as $\sum_{i=1}^{n} f_i$ in which f_i is the frequency related to each age i (see Figure 15). Given the cumulative frequency, the answer to the query "How many people are between the ages of one and six?" can be found by subtracting the CFs of ages six and one.

Since the range of the data might be quite large, a tree structure called a *wavelet histogram* is built over the range of the data set to efficiently address these queries. The lowest level of the tree is built by placing CF values at each node. The higher levels can be created using wavelet decomposition. For example, if using the Haar wavelet, higher level nodes are formed by averaging two consecutive nodes.

With such a detail tree, range queries can be estimated without accessing the entire data set. In the wavelet histogram, each node at any level of the tree provides an estimation for a range of data. Consider a data set with range $0 \leq i \leq n$ for which a wavelet histogram has been built. In general, the ith node at the kth level of this wavelet histogram ($0 \leq k \leq \log_2 n$ and $0 \leq i \leq 2^k - 1$) provides an estimate for all values in the range $(\frac{i \times n}{2^k}, \frac{(i+1) \times n}{2^k})$. For example, consider a situation in which we have only the second level of the tree in Figure 15 and we want to answer the query above: "How many people are between the ages of six and one?". Since six belongs to the right node of the tree at the

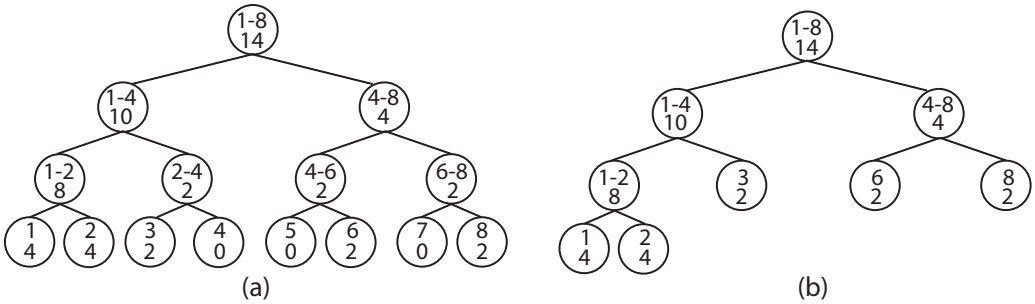


Figure 16: (a) Bisection for the range trees and its nodes. (b) Range tree after optimization.

second level; $(4 \leq 6 \leq 8)$ and one belongs to the left node $(0 \leq 1 \leq 4)$, we can consult the second level of the tree and estimate the answer as $12 - 8 = 4$. Naturally, estimates carry a certain amount of error; in this example, for which the correct answer is $12 - 4 = 8$, the error is equal to 4.

Note that it is not necessary to store the whole tree in order to reconstruct the data, as we can store the detail tree associated with the Haar wavelet decomposition, whose dimension is equal to the dimension of the initial data set. The detail tree is built using the nodes of the wavelet tree. If N is a non-leaf node of the wavelet tree, it has two children: N_l on the left and N_r on the right. The nodes of a (Haar) detail tree are defined as $N - N_l$. Since the leaves of a wavelet tree do not have any children, detail trees are one level shorter than wavelet histograms. Given only the root of the wavelet histogram alongside the complete detail tree, the entire data set can be reconstructed (see Figure 15 (c)).

Although wavelet histograms are powerful, their usefulness is limited by four main factors. First, if the data set consists of floating point numbers, the wavelet histogram cannot be applied on the data set. Second, if the range of the data is not a power of two, a complete tree cannot be made. Third, since wavelet histograms are created based on an unestablished range of numbers, combining two wavelet histogram is not possible without making a new tree. Fourth, the tree does not provide any information about the error and error is not controllable.

Range Trees: The first alternative method to classify quantitative data sets and address range queries is to build range trees. In each node of a range tree, three numbers are saved. Two of the numbers record the range of the data (i.e. the min and max of the range) and the last node records the frequency of the data. As demonstrated in Figure 16 (a), the root of the tree consists of the range of the entire data set in addition to the sum of all frequencies. The next level of the tree is built by bisecting the range and then counting the frequency of each range. If the frequency of a node is zero or one, or all of the numbers associated with a node are the same, bisection is not applied to the node (see Figure 16 (b)).

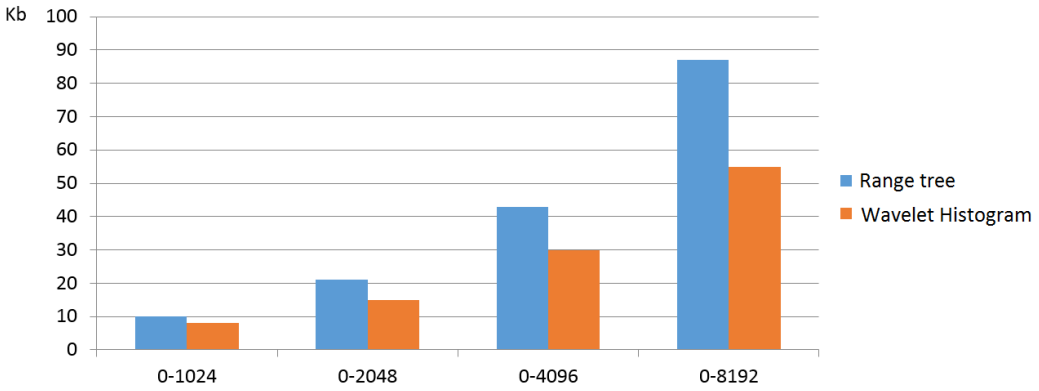


Figure 17: The comparison between range trees and wavelet histograms for different data set ranges. It is clear that the wavelet histogram requires less storage space to store the whole data structure.

To further optimize the structure, if a node does not have a left or right branch, the node is replaced by its only child. This structure can support floating point numbers and is hence advantageous in comparison to the wavelet histogram. However, for each node we need to store two floating point numbers and one integer, in comparison to only one float per node in a wavelet histogram or associated detail tree. Hence, the amount of data used by range trees is higher than that used by wavelet trees. We have tested and compared the amount of memory that is needed by range trees to that needed by wavelet histogram detail trees for data sets with ranges (0,1024) to (0,8192). As illustrated in Figure 17, detail trees require much less memory in comparison to range trees.

Modified Wavelet Histogram: By modifying the wavelet histogram, we can address the four issues that reduce the usefulness of wavelet histograms. The first two issues can be solved by binning. Suppose that the range of a data set is between Min and Max. We can distribute the data set into 2^n bins with $bin\_size = \frac{Max-Min}{2^n}$. This way, both integer and floating point numbers can be handled, and the number of bins is always a power of two. In order to solve the third problem, we can fix Min, Max, and bin_size based on the properties of the data set.

There are two strategies for choosing Min and Max. One is to choose an an appropriate Min and Max based on the type of the data. For instance, for the age data set, a sensible Min and Max could be zero and 100, as the majority of people fall within this age range. The second strategy is to set the maximum and minimum based on the data itself. For instance, given a data set of heights — $(135.5, 157.6, 165, 190.5, 199.5)$ — (see Figure 18), we can take $Max = 199.5$, $Min = 135.5$, and $n = 3$.

Both approaches suffer from some problems, however. For the first approach, it is possible to select an unnecessarily large range beyond what the actual data requires. For

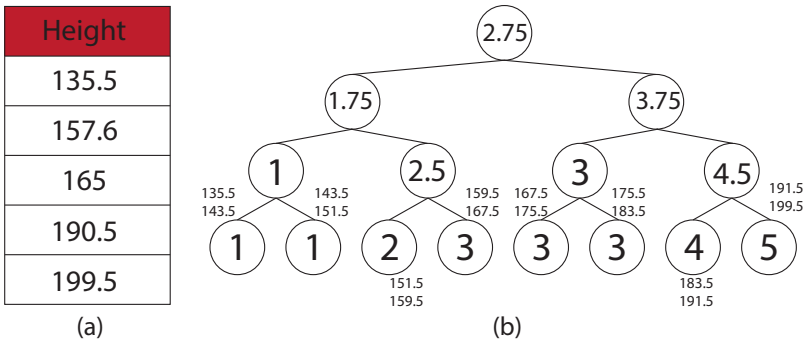


Figure 18: (a) Data set for the height of five people. (b) The wavelet created by the forming the bins with size 8. The range of each bin is shown beside each leaf node.

the second approach, if the data set changes, the bins may need to be resized and the data redistributed. As a result, additional properties of the data, such as how dynamic/prone to change it is, should be considered when selecting Min and Max.

Choosing a value for bin_size should also be based on properties of the data set. As a rule of thumb, bin_size should be selected in such a way that all the data located in a single bin can be processed and transmitted efficiently without the need for a multiresolution approach. However, it must always be a power of two, so that a complete wavelet histogram can be built. For example, in Figure 18, $n = 3$, $bin\_size = 8$, hence 8 bins are created.

Our wavelet histogram tree and its associated binning method is based on a binary Haar wavelet. Wavelet histogram can be built on the data and independent from the underlying DGGS. It is also possible to adapt the tree and the cell structure of the DGGS. In this case, the same factor of refinement as the DGGS should be also used for the wavelet histogram tree. For instance, for an A3H DGGS, a ternary or a 9-ary Haar wavelet in which its coarse nodes are the average of their nine children can be used to build the wavelet histogram tree compatible with the DGGS cell structure.

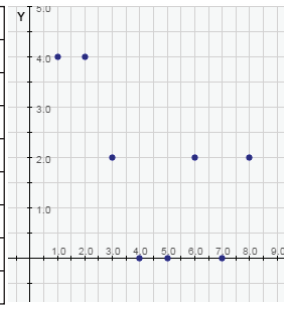
Although the Haar wavelet histogram is simple and has been improved upon to better meet our needs in data representations for Digital Earth frameworks, it does not provide any information about the error at any level of the tree. In addition, we cannot control the error before constructing the tree. As a result, we provide a method called Least square Wavelet to overcome this issue.

Lease Squares Wavelet: In the least squares wavelet, we first approximate data with a known and reasonable error and then progressively improve the estimate by reconstructing the data using the known error. This methodology is especially useful in data transmission (an important process in any Digital Earth), allowing queries to be estimated with a known maximum error and then refined as the individual error values are transmitted.

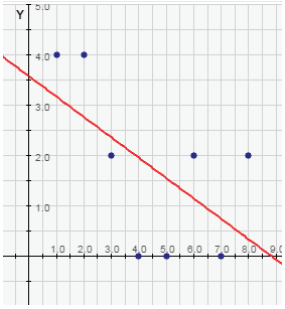
Consider n data points (f_i, $0 \leq i < n$) that need to be retrieved or transmitted. We

Age	Frequency
1	4
2	4
3	2
4	0
5	0
6	2
7	0
8	2

(a)



(b)



(c)

Age	Estimation
1	3.166
2	2.761
3	2.356
4	1.951
5	1.546
6	1.142
7	0.736
8	0.337

(d)

Age	Erro(r_i)
1	0.834
2	1.239
3	-0.356
4	-1.951
5	-1.546
6	0.858
7	-0.736
8	1.663

(e)

Figure 19: (a) Data set for the age of eight people. (b) Plot of the data set in (a). (c) The least squares line passing through the data set. (d) Estimation of the data set in (a) by evaluating the linear function in (c). (e) The error, or residual, of the data evaluation.

propose to approximate these data points with piecewise linear functions (connected by points p_i, $0 \leq i < m$) by solving a least squares system (see Figure 21). Consider a system of equations $\Phi P = F$ in which P and F are vectors composed of p_i and f_i and Φ is an $n \times m$ matrix with linear basis functions. The solution of this system can be found using the normal equation: $P = (\Phi^T \Phi)^{-1} \Phi^T F$ [10]. Figure 19 illustrates a simple example, featuring a single-piece linear function to approximate the data set from Figure 15.

The error of this approximation (or residual) is given by $\mathbf{r} = F - \Phi P$. For example, in Figure 19, solving the least squares system provides two points $P_0 = (1, 3.166)$ and $P_1 = (8, 0.337)$ that form the line passing through the data set. To evaluate the error, we can subtract the data set values from corresponding values on the linear function $l(t) = vt + P_0$ where $v(t) = P_1 - P_0$. Therefore, the error for the f_i, which is the ith data value $(0 \leq i \leq 7)$, is $r_i = f_i - l(\frac{i}{7})$ (see Figure 19 (c)). To evaluate the total error, we can use different norms, but the least squares solution minimizes norm-2, $\|\mathbf{r}\|_2 = \sqrt{r_0^2 + ... + r_n^2}$. By increasing the number of control points, we can reduce the total error and obtain a better approximation of the data. Therefore, by evaluating $\|\mathbf{r}\|_2$, we gauge the error of our approximation and decrease it as necessary, as opposed to setting up the histogram on the actual data without control over the error.

Now the question becomes how to gradually add more information to the data set and reduce the error. To do so, we propose applying the wavelet histogram to the residuals instead of the data itself (see Figure 20). We can then gradually send nodes of the tree at different levels and improve the initial results obtained from the least squares solution. Formally, if we have the residual r, we can decompose it to $C_n, D_n, D_{n-1}, ..., D_0$ in which $C_n = A^n \mathbf{r}$ and $D_i = B^i \mathbf{r}$ (see Algorithm 1 for the case of Haar multiresolution). Note that $C_{n-1} = PC_n + QD_n$ and A, B, P, Q are the matrices of a known multiresolution framework (e.g Haar). When $C_n, D_n, D_{n-1}, ..., D_0$ is available, r can be reconstructed as $\mathbf{r} = P^n C_n + P^{n-1} Q D_{n-1} + P^{n-2} Q D_{n-2} + ... + Q D_0$ (see Algorithm 2 for the reconstruction

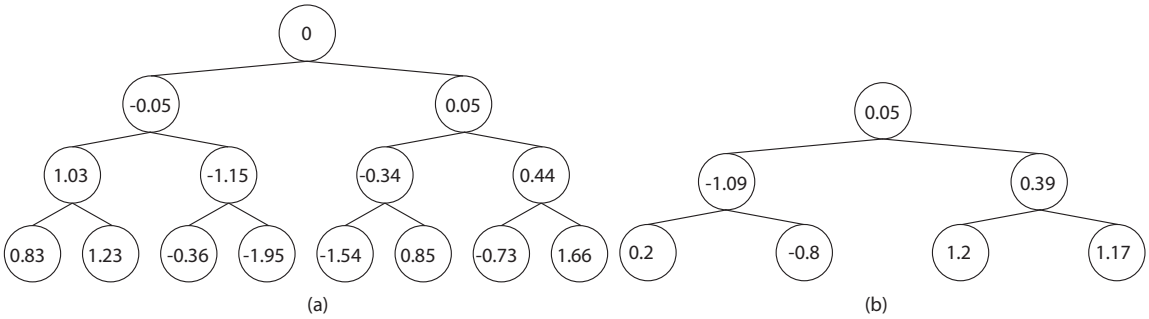


Figure 20: (a) Haar histogram wavelet for residuals. (b) Detail tree for residuals.

of r and F). As the root of the Haar wavelet histogram (C_n) is the average of the residuals and the average of the residual is always zero, C_n is not needed to reconstruct residuals.

Algorithm 1 Decomposition of r into $D_n, D_{n-1}, ..., D_0$ ($C_n = 0$).

for $j = 0$ to n, step 1 **do**
  for $i = 0$ to sizeof(r), step 2 **do**
    $Temp(\frac{i}{2}) = \frac{r(i)+r(i+1)}{2}$.
    $D_j(\frac{i}{2}) = r(i) - Temp(\frac{i}{2})$.
  end for
  $r = Temp$
end for

We intend to prove that if we add the nodes resulting from the Haar wavelet tree to the residuals, we can always reduce the residual (i.e. $\|\mathbf{r} - r_{send}\| \leq \|\mathbf{r}\|$). Once the $\mathbf{r}$ is perfectly reconstructed, it is clear that the approximations get better, as the initial data is completely reconstructed. However, an important property of the Haar wavelet is that applying Haar wavelets on residuals and gradually sending the results guarantees that the error decreases. Assume that all the components of r_0 are equal to each other, i.e. $r_{send} = (\,{}_{R,R,...,R}\,)^T$. We denote the entries of r as r_i and, therefore, $\mathbf{r} = (\,{}_{r_0,r_1,...,r_n}\,)^T$. Using these notations, we can rewrite the error equation as:

$$\|\mathbf{r} - r_{send}\| \quad = \quad (r_0 - R)^2 + (r_1 - R)^2 + \ldots + (r_n - R)^2.$$

We want to show that $\|\mathbf{r} - r_0\| \leq \|\mathbf{r}\|$ holds. We can rewrite this equation as:

$$(r_0 - R)^2 + (r_1 - R)^2 + \ldots + (r_n - R)^2 \quad \leq \quad {r_0}^2 + {r_1}^2 + \ldots + {r_n}^2.$$

Opening the left side of the equation, we have

$$(r_0^2 + 2r_0 + R^2) + \ldots + (r_n^2 + 2r_n + R^2) \quad \leq \quad {r_0}^2 + {r_1}^2 + \ldots + {r_n}^2.$$

Algorithm 2 Reconstruction of F after receiving $D_n, D_{n-1}, ..., D_0$.

$F = \Phi P$
for $s = n$ to $s \leq 0$ **do**
  for $j = 0$ to $j < 2^{n-s}$ **do**
    for $i = 0$ to $i < 2^s$ **do**
      $r_{i+2j} = r_{i+2j-1} + D_s(j)$.
      $r_{i+2j+2^s} = r_{i+2j+2^s} - D_s(j)$.
    end for
  end for
end for
$F = F + r$

Canceling out the same terms from the left and the right side of the equation, we can deduce:

$$\begin{aligned} nR^2 &\leq 2R(r_0 + \ldots + r_n) \Rightarrow \\ R &\leq 2\frac{\sum_{i=0}^{n}}{n} \end{aligned} \tag{5.1}$$

One possible value for R to satisfy this inequality condition is the average of the residuals resulting from the Haar wavelet. Since each node of the any Haar wavelet (e.g. binary or ternary) tree is the average of its sub-tree, the values of the Haar wavelet tree always reduce the error. Table 2 illustrates an example in which the error has been reduced by adding the nodes of the Haar wavelet tree to the data set (the data is the same as the data illustrated in Figure 21).

Table 2: Adding the nodes of the Haar wavelet tree at different resolutions reduces the error. There is no change at the first resolution since the average of the residuals is always zero.

Resolution	Error
0	1.3461
1	1.3461
2	1.3391
3	1.3181
4	1.2497
5	0

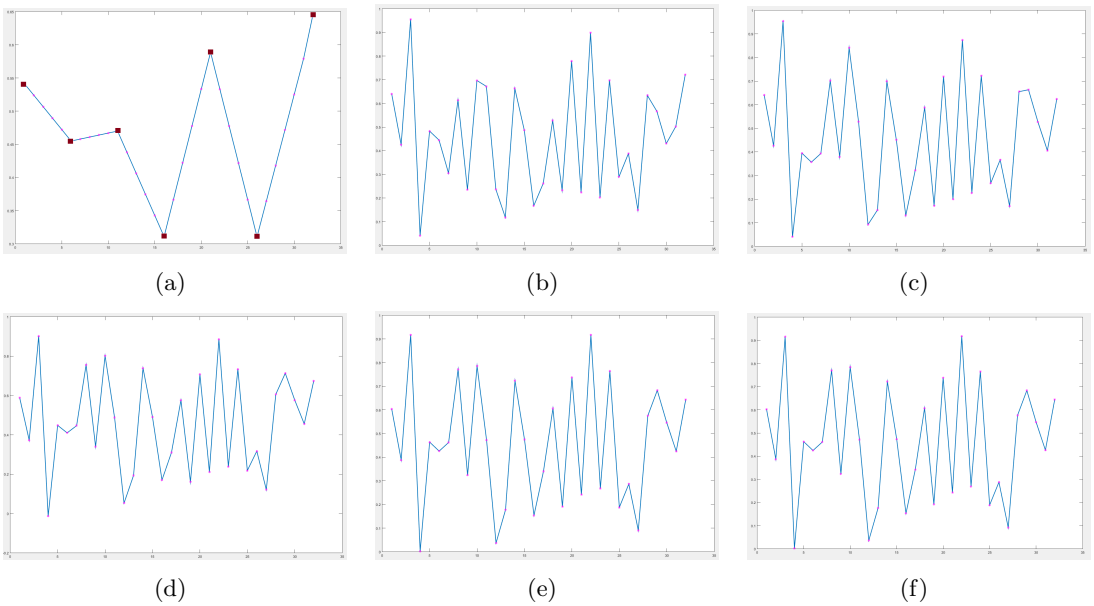


Figure 21: (a) Estimating the data set using piecewise linear functions and points illustrated as squares. (b), (c), (d), (e) Adding a portion of the residual to the piecewise linear functions using a Haar wavelet tree, until the data set is perfectly reconstructed in (f).

6 Conclusion and Future Work

In this paper, we introduced different techniques to represent data sets in an A3H DGGS. Some of these methods such as least square wavelets, modified wavelet histogram, and spherical vector representations are general and can be applied to any DGGS. We also discussed data management of three important types of data sets: imagery, vector, and quantitative data sets. For each method that we described, we discussed its advantages and disadvantages over other methods. There are many directions that can be explored to improve the methods. For instance, there are different queries that these methods should be tested on to determine their performance. Finding the intersection or union of two vector data sets as represented in any of these forms described above is one such example. Designing new multiresolution frameworks instead of Haar that can support compression of the quantitative data sets while preserving the quality of the data and producing small errors is another interesting future work. For quantitative data sets, in addition to range queries, examining other queries can be considered as a future work.

References

[1] Anupam Agrawal, M. Radhakrishna, and R. Joshi. Geometry-based mapping and rendering of vector data over LOD phototextured 3D terrain models. In *Proc. of WSCG '06*, 2006.

[2] Troy Alderson, Ali Mahdavi-Amiri, and Faramarz F. Samavati. Multiresolution on spherical curves. *Graphical Models*, 2016.

[3] Aleksey Boyko and Thomas Funkhouser. Extracting roads from dense point clouds in large scale urban environment. *ISPRS Journal of Photogrammetry and Remote Sensing*, 66(6):S2–S12, 2011.

[4] M. A. Brovelli, M. Cannata, and U. M. Longoni. Managing and processing LIDAR data within GRASS. In *Proc. of the GRASS Users Conference*, volume 29, 2002.

[5] Kaushik Chakrabarti, Minos Garofalakis, Rajeev Rastogi, and Kyuseok Shim. Approximate query processing using wavelets. *The VLDB JournalThe International Journal on Very Large Data Bases*, 10(2-3):199–223, 2001.

[6] Simon Clode, Franz Rottensteiner, Peter Kootsookos, and Emanuel Zelniker. Detection and vectorization of roads from lidar data. *Photogrammetric Engineering & Remote Sensing*, 73(5):517–535, 2007.

[7] A Fortier, D Ziou, C Armenakis, and S Wang. Survey of work on road extraction in aerial and satellite images. Technical report, Département de mathématiques et dínformatique, Université de Sherbrooke, 1999.

[8] M.-F. Auclair Fortier, D. Ziou, C. Armenakis, and S. Wang. Automated correction and updating of road databases from high-resolution imagery. *Canadian Journal of Remote Sensing*, 27(1):76–89, 2001.

[9] Minos Garofalakis and Phillip B Gibbons. Wavelet synopses with error guarantees. In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, pages 476–487. ACM, 2002.

[10] Gene H. Golub and Charles F. Van Loan. *Matrix Computations (3rd Ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.

[11] Charles Han and Hugues Hoppe. Optimizing continuity in multiscale imagery. *ACM Transactions on Graphics*, 29(6):171:1–171:10, 2010.

[12] Oliver Kersting and Jürgen Döllner. Interactive 3D visualization of vector data in GIS. In *Proceedings of the 10th ACM International Symposium on Advances in Geographic Information Systems*, pages 107–112. ACM, 2002.

[13] M. Lambers and A. Kolb. Ellipsoidal cube maps for accurate rendering of planetary-scale terrain data. In *Proc. of the Pacific Conference on Computer Graphics and Applications*, PG '12, pages 5–10, 2012.

[14] Jeffrey M Lane and Richard F Riesenfeld. A theoretical development for the computer generation and display of piecewise polynomial surfaces. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (1):35–46, 1980.

[15] Tao Li, Qi Li, Shenghuo Zhu, and Mitsunori Ogihara. A survey on wavelet applications in data mining. *ACM SIGKDD Explorations Newsletter*, 4(2):49–68, 2002.

[16] P. Longley. *Geographic Information Systems and Science*. Wiley, 2nd edition, 2005.

[17] Ali Mahdavi-Amiri, Troy Alderson, and Faramarz Samavati. A survey of digital earth. *Computers & Graphics*, 53, Part B:95 – 117, 2015.

[18] Ali Mahdavi-Amiri, Faraz Bhojani, and Faramarz F. Samavati. One-to-two digital Earth. In *Proc. of the International Symposium on Visual Computing*, ISVC '13, pages 681–692, 2013.

[19] Ali Mahdavi-Amiri, Erika Harrison, and Faramarz F. Samavati. Hexagonal connectivity maps for digital Earth. *International Journal of Digital Earth*, pages 1–20, 2014.

[20] Ali Mahdavi-Amiri, Erikal Harrison, and Faramarz F. Samavati. Hierarchical grid conversion. *Computer Aided Design*, 2016.

[21] Ali Mahdavi-Amiri and Faramarz F. Samavati. Connectivity maps for subdivision surfaces. In *GRAPP/IVAPP*, pages 26–37, 2012.

[22] Ali Mahdavi-Amiri and Faramarz F. Samavati. Adaptive atlas of connectivity maps. In *Proc. of the 8th International Conference on Curves and Surfaces*, Lecture Notes in Computer Science. Springer, 2014.

[23] Ali Mahdavi-Amiri and Faramarz F. Samavati. Atlas of connectivity maps. *Computers & Graphics*, 39:1 – 11, 2014.

[24] Ali Mahdavi-Amiri, Faramarz F. Samavati, and Perry Peterson. Categorization and conversions for indexing methods of discrete global grid systems. *ISPRS International Journal of Geo-Information*, 4:320–336, 2015.

[25] Yossi Matias, Jeffrey Scott Vitter, and Min Wang. Wavelet-based histograms for selectivity estimation. In *ACM SIGMoD Record*, volume 27, pages 448–459. ACM, 1998.

[26] Helmut Mayer, Ivan Laptev, and Albert Baumgartner. Multi-scale and snakes for automatic road extraction. In *Computer Vision — ECCV'98*, volume 1407 of *Lecture Notes in Computer Science*, pages 720–733. Springer Berlin Heidelberg, 1998.

[27] J. B. Mena. State of the art on automatic road extraction for GIS update: a novel classification. *Pattern Recognition Letters*, 24(16):3037–3058, 2003.

[28] Susanna Minasyan, Radomir Stankovic, and Jaakko Astola. *Computer Aided Systems Theory - EUROCAST 2009: 12th International Conference, Las Palmas de Gran Canaria, Spain, February 15-20, 2009, Revised Selected Papers*, chapter Ternary Haar-Like Transform and Its Application in Spectral Representation of Ternary-Valued Functions, pages 518–525. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[29] Perry Peterson. Close-packed, uniformly adjacent, multiresolutional, overlapping spatial data ordering. US Patent 8,400,451 (issued March 19, 2013), 2004.

[30] Zhiyuan Qiao, Jingnong Weng, Zhengwei Sui, Heng Cai, and Xuzhao Zhang. A rapid visualization method of vector data over 3D terrain. In *Proc. of the 19th International Conference on Geoinformatics*, pages 1–5, 2011.

[31] F. Rottensteiner. Automatic generation of high-quality building models from lidar data. *IEEE Computer Graphics and Applications*, 23(6):42–50, 2003.

[32] Kevin Sahr. Location coding on icosahedral aperture 3 hexagon discrete global grids. *Computers, Environment and Urban Systems*, 32(3):174–187, 2008.

[33] Kevin Sahr, Denis White, and A. Jon Kimerling. Geodesic discrete global grid systems. *Cartography and Geographic Information Science*, 30(2):121–134, 2003.

[34] Hanan Samet. *Foundations of Multidimensional and Metric Data Structures (The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

[35] Arne Schilling, Jens Basanow, and Alexander Zipf. Vector based mapping of polygons on irregular terrain meshes for web 3D map services. In *Proc. of the 3rd International Conference on Web Information Systems and Technologies*, WEBIST '07, pages 198–205, 2007.

[36] Martin Schneider, Michael Guthe, and Reinhard Klein. Real-time rendering of complex vector data on 3D terrain models. In *Proc. of the 11th International Conference on Virtual Systems and Multimedia*, pages 573–582, 2005.

[37] A.K. Shackelford and C.H. Davis. Fully automated road network extraction from high-resolution satellite multispectral imagery. In *Proc. of the IEEE Internation Geoscience and Remote Sensing Symposium, 2003*, volume 1 of *IGARSS '03*, pages 461–463, 2003.

[38] Jaya Shukla, Manoj Alwani, and Anil Kumar Tiwari. A survey on lossless image compression methods. In *2010 2nd International Conference on Computer Engineering and Technology*.

[39] Eric J. Stollnitz, Tony D. Derose, and David H. Salesin. *Wavelets for computer graphics: theory and applications*. Morgan Kaufmann Publishers Inc., 1996.

[40] A. Vince and X. Zheng. Arithmetic and Fourier transform for the PYXIS multi-resolution digital Earth model. *International Journal of Digital Earth*, 2(1):59–79, 2009.

[41] Jeffrey Scott Vitter and Min Wang. Approximate computation of multidimensional aggregates of sparse data using wavelets. In *ACM SIGMOD Record*, volume 28, pages 193–204. ACM, 1999.

[42] Jeffrey Scott Vitter, Min Wang, and Bala Iyer. Data cube approximation and histograms via wavelets. In *Proceedings of the seventh international conference on Information and knowledge management*, pages 96–104. ACM, 1998.

[43] Zachary Wartell, Eunjung Kang, Tony Wasilewski, William Ribarsky, and Nickolas Faust. Rendering vector data over global, multi-resolution 3D terrain. In *Proc. of the Symposium on Data Visualisation*, VISSYM '03, pages 213–222, 2003.

[44] Thomas Wendler. *Verfahren fr die hierarchische Codierung von Einzelbildern in medizinischen Bildinformationssystemen*. PhD thesis, Aachen, 1987. Aachen, Techn. Hochsch., Diss., 1987.

[45] Denis White, Jon A. Kimerling, and Scott W. Overton. Cartographic and geometric components of a global sampling design for environmental monitoring. *Cartography and Geographic Information Science*, 19(1):5–22, 1992.

[46] Lance Williams. Pyramidal parametrics. *ACM SIGGRAPH Computer Graphics*, 17(3):1–11, 1983.

[47] C. Zhang. *Fundamentals of Environmental Sampling and Analysis*. Wiley, 2007.

[48] Mengyun Zhou, Jing Chen, and Jianya Gong. A virtual globe-based vector data model: quaternary quadrangle vector tile model. *International Journal of Digital Earth*, (ahead-of-print):1–22, 2015.