

Geospatial Data Organization Methods with Emphasis on Aperture-3 Hexagonal Discrete Global Grid Systems

Ali Mahdavi Amiri

*School of Computing Science / Simon Fraser University / Burnaby / BC / Canada
Department of Computer Science / University of Calgary / Calgary / AB / Canada*

Troy Alderson

Department of Computer Science / University of Calgary / Calgary / AB / Canada

Faramarz Samavati

Department of Computer Science / University of Calgary / Calgary / AB / Canada

ABSTRACT

Digital Earth frameworks deal with data sets of different types collected from various sources. To effectively store, retrieve, and transmit these data sets, efficient multi-scale data representations that are compatible with the underlying structure of the Digital Earth framework are required. In this article, we describe several such techniques and their properties: namely, how to represent data in the multi-scale cell hierarchy of a discrete global grid system (DGGS) or in the multi-scale hierarchy of a customized wavelet transform. We also discuss how these techniques can be tuned to be applicable to the A3H DGGS.

Keywords: DGGS, data transmission, wavelet, regression

RÉSUMÉ

Les modèles de planète virtuelle traitent des ensembles de données de nature différente, colligées à partir de sources différentes. Le stockage, la consultation et la transmission efficaces de ces ensembles de données exigent des représentations multi-échelles des données qui sont compatibles avec la structure sous-jacente du modèle de planète virtuelle. Les auteurs décrivent plusieurs des techniques envisageables ainsi que leurs propriétés, plus précisément la façon dont elles permettent de présenter les données dans la hiérarchie des cellules multi-échelles d'un système de grille globale discrète (*discrete global grid system* – DGGS) ou dans la hiérarchie multi-échelles d'une transformée en ondelettes personnalisée. Ils traitent également de la façon dont ces techniques peuvent être adaptées aux DGGS A3H.

Mots clés : gestion des données, ondelette, régression, planète virtuelle, structures de données, système de grille globale discrète

Introduction

The Digital Earth provides a representation of the Earth on which data sets of different types and from different sources can be integrated, analyzed, and visualized (Mahdavi-Amiri, Alderson, and Samavati 2015a), and is commonly implemented using a discrete global grid system (DGGS; Sahr, White, and Kimerling 2003). In a DGGS, the Earth is typically approximated using a spherical polyhedron that is iteratively refined to generate a multiresolution hierarchy (Mahdavi-Amiri and others 2015a). Through this process, the surface of the Earth is discretized into a hierarchical set of cells, where the area of a refined/child

cell as compared with that of a parent cell is described by the factor (or aperture) of the refinement. These cells are then projected onto the sphere using one of a variety of spherical projection methods, although equal-area projections are usually more desirable when data analysis is emphasized (Mahdavi-Amiri, Bhojani, and Samavati 2013; White, Kimerling, and Overton 1992).

DGGSs consisting primarily of hexagonal cells are particularly popular (Sahr 2008; Tong and others 2013; Vince 2006) due to the superior sampling behaviour of hexagonal tilings. Especially common are those DGGSs, collectively known as aperture-3 hexagonal (A3H) DGGSs, that consist of hexagonal cells refined by a factor of 3

(aperture 3), which provides a smooth transition between resolutions (Mahdavi-Amiri, Harrison, and Samavati 2014; Peterson 2004; Vince and others 2009). See Figure 1 for an illustration.

As the core functionality of any Digital Earth lies in the efficient management of geospatial data (primarily imagery and elevation data, vector data, and quantitative data), techniques for the representation of geospatial data merit discussion. One of the core obstacles inherent in this topic lies in the immense scale of and storage requirements for many geospatial data sets, which continue to grow with improvements to data capturing technologies. As a result, geospatial data often cannot be stored locally on a single machine, and must be distributed across a set of servers. Client machines with limited memory may then connect to these servers to request data or initiate queries, ranging from requests for specific images in a given area to statistical analyses of quantitative data.

The issue of scale additionally requires that data representations be multiscale. The immense difference between the scope of a city neighbourhood and that of an entire country makes it often impractical to visualize or process a data set at its native resolution. While the cell hierarchy of a DGGS offers a natural multiscale representation for different data sets, alternative representations can offer additional benefits for certain operations. In particular, appropriate multiscale representations can be used to counteract the adverse effect of data size increases on transmission and query processing times and ensure that queries are handled quickly and accurately.

One such alternative representation is the wavelet transform. Wavelet transforms are a technique originating from approximation theory and signal processing that allow a signal to be decomposed into its low-frequency (coarse approximations) and high-frequency (wavelets or details) components and recombined to recreate the original signal. This technique has been adapted for use in computer graphics (Stollnitz, Deroose, and Salesin 1996), where it

allows numerous types of graphical data to be decomposed into a coarse approximation and associated high-frequency details. To provide a rough estimate of the data, it is possible to use the low-frequency or coarse approximation and later reconstruct the data using the complementary details. In data transmission, for example, low-frequency data that have the main trend of the data can be sent first to provide a rough estimate of the data, and then details can be sent and added to perfectly reconstruct the data without losing the information or long waiting time.

In the next section, we describe different methods that may be used to represent, in a multiscale manner, the three main types of geospatial data, with particular emphasis on client server Digital Earth architectures supported by an A3H DGGS. In essence, we would like to show how we can reduce the artifacts and issues of data size increases on transmission and query processing times using multiscale representations of geospatial data such as wavelet transforms. We present a number of existing and novel methods that represent data with wavelet transforms or convert data into a wavelet-supported form, largely for the purpose of efficient data transmission or querying. Such methods allow an initial visualization or query result to be produced with approximate data, which may then be refined to the correct result using loss-less wavelet techniques as additional data arrives via transmission.

In the following section, we briefly overview how geospatial data may be stored in and retrieved from a DGGS. The primary mechanism underlying these processes is a DGGS indexing system. Two such indexing systems for A3H DGGSs, which assign a unique index to each cell of the DGGS, are described in detail.

We then describe how imagery and elevation data sets can be extracted from the cell hierarchy of an A3H DGGS for the purposes of transmission or rendering. This is then followed by discussions of how these data may be losslessly compressed for transmission to client machines using, for example, the Haar wavelet transform. A novel wavelet

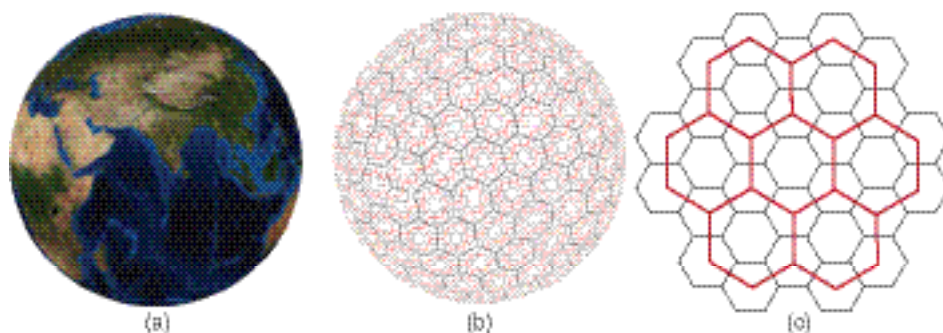


Figure 1. (a) and (b) The PYXIS Digital Earth, which is an A3H DGGS, and its cell structure; (c) the 1-to-3 hexagonal refinement (aperture 3)

Source: Blue Marble.

transform that is compatible with an A3H DGGS, the integer ternary Haar wavelet, is additionally described.

Following this, we describe approaches to representing vector data in a DGGS. These approaches include rasterization, hierarchical cell representation, and a geometry-based spherical vector representation. We introduce here a novel generalization of the hierarchical cell representation (which works with quad trees; Samet 2005) to more general types of DGGS.

Finally, we discuss the management of quantitative data sets. Statistics pertaining to each cell may be transmitted and analyzed before the transmission completes (with a known, progressively decreasing error) using wavelet histograms and range trees. We present a novel modification of nLT trees that allows storage requirements to be reduced, and a modification of the wavelet histogram that approximates the data using a piecewise linear function. By applying the wavelet histogram to the difference between this function and the original data, the error from analyzing the incomplete data can be controlled and reduced.

Related Work

Digital Earths, their applications (data analysis being an important example), and the different methods used to build, visualize, and index them have been studied extensively in the literature (Mahdavi-Amiri and others 2015a). Three important types of data sets for Digital Earths are considered in this article: imagery/elevation, vector, and quantitative data. In the following, we describe each of these types of 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 for the visualization and analysis of locations and are often used as textures for 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 colour or height are assigned to these cells (Sahr 2008).

DGGSs induce a natural multiresolution representation for images, as each resolution of a DGGS contains a set of images that represent the Earth or specific regions at that particular resolution. However, alternative multiresolution representations for images can also be beneficial. For instance, mip-mapping (Williams 1983) has been used (for example, in Han and Hoppe 2010) to establish a continuous transition between different correlated images. In addition, wavelet transforms (see Stollnitz and others 1996) can be used to produce a low-resolution version of an image that can be efficiently transmitted through a network and

gradually restored to its original resolution on a client as data arrive. A simple and useful wavelet transform, which we make use of throughout the article, is the Haar wavelet (refer to *Image Data Transmission: The Haar Wavelet* for a description).

Elevation data sets are very similar to images, as data elevation models are uniform grids with values for heights instead of colours. As a result, their rasterization and wavelet transforms are very similar to those employed for images, the only difference being that there is no need to use integer wavelets for elevation data sets. This is because they are often represented by floating point values as opposed to integers.

VECTOR DATA SETS

Vector data sets are available in the form of points, polylines, or polygons. These points may be connected by spherical or ellipsoidal arcs and represent networks (such as road or river networks) or region boundaries (such as those belonging to continents, cities, etc.). Vector data can be produced via ground surveying, LIDAR, and photogrammetry.

Features in LIDAR data sets can be detected and vectorized using different techniques (Boyko and Funkhouser 2011; Mena 2003; Rottensteiner 2003), although imagery data alone can also be used to extract vector data sets (Auclair-Fortier, Ziou, and Armenakis 1999; Mayer, Laptev, and Baumgartner 1998; Mena 2003).

Three main methods are available to visualize vector data sets: texture-based, geometry-based, and shadow-volume-based (Zhou, Chen, and Gong 2015). Texture-based approaches rasterize the vector data into textures that are then mapped onto the terrain surface (Keresting and Dollner 2002). Geometry-based approaches consider the geometry of the vector data separately from the geometry of the terrain (Agrawal, Maringati, and Joshi 2006; Schilling, Basanow, and Zipf 2007; Schneider, Guthe, and Klein 2005), and can be used with wavelet transforms (Alderson, Mahdavi-Amiri, and Samavati 2016) to provide multiscale rendering. 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 (Lambers and Kolb 2012).

As the individual grids of a DGGS each provide a rasterization of the Earth, vectors are often converted into raster images for storage in a DGGS, particularly on the Web, as in Bing and Google Maps. Alternative approaches include the hierarchical cell representation, in which a vector is represented as an ordered set of cells at each resolution of the DGGS. Treating the cells of the DGGS as "buckets" allows data storage techniques similar to those used by quad trees to be employed (Samet 2005) and can be used to store the actual vector geometry to address rasterization artifacts. Example works that use the approach include

Peterson (2004), which uses the cells of an A3H DGGS to store vectors, and Zhou and others (2015), which employs a multiresolution vector pyramid to store vectors in a QQM (quaternary quadrangle model) DGGS.

QUANTITATIVE DATA SETS

Quantitative data sets provide statistical and other numerical 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; Zhang 2007) and are assigned to the cell of a DGGS as attributes related to that particular location.

To obtain useful information from these data sets, techniques from the field of information processing may be used. Though not specific to DGGS, of particular importance is that query processing times on such data sets can become prohibitively expensive in many situations (including data mining; Li and others 2002) and multidimensional aggregate computation (Vitter, Wang, and Iyer 1999), whether due to the complicated nature of the query or to the size of the data set. As a result, it can be useful to run the queries on simplified versions of the data to provide fast initial results, with the possibility of refining the data. To accomplish this, wavelets can again prove useful, and have been used for this purpose by Chakrabarti and others (2001); Garofalakis and Gibbons (2002); Li and others (2002); Vitter and others (1999); and Vitter and Wang (1998).

Data Storage and Retrieval

In a DGGS, geospatial data are assigned to and retrieved from cells. To support this functionality, each cell requires a unique identifier, called a cell index. Using these indices as references into a database, folder structure, or file (e.g., a spreadsheet) allows cells to be associated with data stored in any of these formats (Mahdavi-Amiri, Samavati, and Peterson 2015b). For instance, one possibility is to utilize

a nested folder structure, in which folders are associated with cells and contain files for data sets belonging to the cell (e.g., textures, elevation data). The folder for a child cell is contained within the folder for its parent, allowing the folder structure to be traversed similarly to the DGGS cell structure. Of course, this is not the only method of storing geospatial data sets on disk. Interested readers may refer to (Southern Terra Cognita Laboratory 2017) for a DGGS implementation.

Hence, an essential component of any DGGS is its associated cell indexing system. Three main types of indexing systems are employed within different DGGSs: hierarchy-based, coordinate-based, and space-filling curve (SFC)-based (Mahdavi-Amiri and others 2015b). We describe in this section two indexing systems that have been proposed for A3H DGGSs: one hierarchy-based and one coordinate-based.

To define a hierarchy-based indexing system on the hexagonal cells of an A3H DGGS, a hierarchical relationship among the cells is needed (Peterson 2004; Sahr 2008; Vince and Zheng 2009). However, as hexagonal refinements are not congruent, this hierarchical relationship is not straightforward to define. The PYXIS hierarchy presents one possibility for defining hierarchical relationships between cells at different resolutions (Peterson 2004; Vince and Zheng 2009). In this hierarchy, cells are categorized into two types (A and B) generating different fractal shapes called tiles throughout the resolutions that fit together to cover the entire surface of a sphere (see Figure 2). Type B cells with index b have children with indices bi ($0 \leq i \leq 6$; $i \in \mathbb{N}$), while a type A cell has only one child with index $a0$. $a0$ and $b0$ are considered to be of type B, while the other bi cells are of type A (Mahdavi-Amiri and others 2014; Vince and Zheng 2009). We have used this type of indexing system and the tree structure resulting from the parent-child relationships between cells to define a hierarchical cell representation for feature vectors (as discussed in *Hierarchical Cell Representation*).

Hierarchy-based indexing systems are efficient at addressing hierarchical queries, but neighbourhood queries

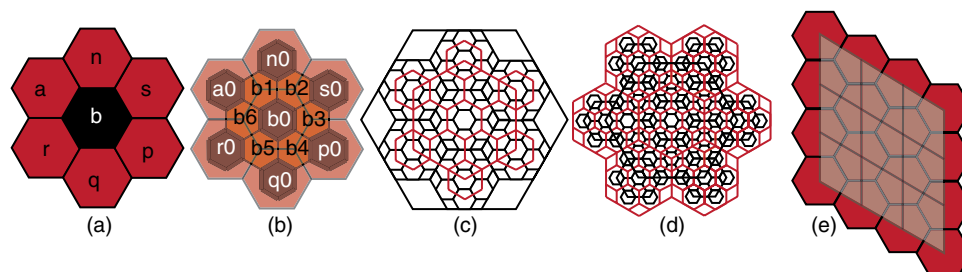


Figure 2. (a) and (b) PYXIS hierarchical indexing method; (c) and (d) fractal tiles created from type A and B cells; (e) a rhombus covering a set of hexagonal cells. The vertices of the refined rhombus are aligned with the centroids of the hexagonal cells.

(finding the neighbours 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, coordinate-based indexing system can be defined on the hexagonal cells of a given A3H DGGS (Mahdavi-Amiri and others 2014). One such system is based on the duality relationship between hexagons and triangles (see Figure 2). Once the centroids of the hexagons are connected to form triangles, and pairs of triangles are combined into rhombuses (Mahdavi-Amiri, Harrison, and Samavati 2016), the axes of a coordinate system can be defined so that they align with the edges of the rhombuses. Using the resulting coordinate system, the vertices of the rhombuses and their associated hexagonal cells may be indexed. This coordinate-based indexing system can be used to arrange hexagonal cells in a quadrilateral shape (i.e., rhombuses) that is compatible with standard graphics pipelines (e.g., OpenGL) and is easy to sample and render. For more details, see *Converting from DGGS Cells to Images*.

However, in to provide interoperability and retrieve data from an external DGGS, a conversion between the cell indices of the different DGGSs is required. In essence, given a cell in the destination DGGS, a corresponding cell that represents approximately the same area in the target DGGS must be determined. This requires a correspondence between the resolutions of the two DGGSs, which can be established based on, for example, the number of cells that each DGGS provides at specific resolutions. In the absence of an existing conversion, a cell index from the destination DGGS can be converted to an index in the target DGGS using a common intermediate DGGS or geospatial coordinate system (for example, latitude/longitude) for which an existing conversion is known. See Mahdavi-Amiri and others (2015b) for more details on converting between two Digital Earth frameworks.

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 in order to be represented within the DGGS. In this section, we describe methods to assign imagery data to the cells of an A3H DGGS and to transmit it through a network. The benefits to visualization offered by the mechanism used to convert the data (namely, the dual conversion) are also discussed. Due to similar structure, elevation data sets are handled similarly.

CONVERTING FROM DGGS CELLS TO IMAGES

Assigning imagery data to DGGS cells requires a establishing a correspondence between the cells of the DGGS

and the pixels of the image. This can be accomplished using standard cartographic projection techniques, such as Snyder's projection (Snyder 1992). However, for some purposes, we may wish to convert the DGGS cells back into imagery. For this to be done, every pixel should correspond to a single cell. This can be accomplished using a conversion method described by (Mahdavi-Amiri and others 2016) in which hexagonal cells can be efficiently converted to triangles and quads that are more compatible with standard libraries such as OpenGL. Using this conversion, satellite images and elevation data may be extracted from a DGGS more naturally for use as textures or displacement maps, respectively (see Mahdavi-Amiri and others 2014 for more details on how to efficiently send these data to the GPU), or for transmission to clients.

The basic idea behind this method is to use a dual conversion, forming triangles that connect the centroids of neighbouring hexagonal cells (Mahdavi-Amiri and others 2016). In essence, the vertices of a triangle each correspond to a hexagonal cell. When these triangular cells are created, they are packed into congruent quadrilaterals, or rhombuses (see Figure 3), which can be used for transmission or to render cells efficiently on the GPU (Mahdavi-Amiri and others 2014; Mahdavi-Amiri and Samavati 2012, 2014a, 2014b). Note that cracks or gaps can appear between rhombuses at different resolutions, but can be connected by a set of "zippers" to patch the final surface of the Earth (Mahdavi-Amiri and Samavati 2014b).

Given this conversion of hexagonal cells into rhombuses compatible with standard compression and rendering techniques, we can serialize hexagonal cells into quadrilateral images by sampling the DGGS at each vertex of a rhombus. As the aperture of the cells in an A3H DGGS is three, the resulting sampled images have dimensions of $3^n \times 3^n$ (where n depends on the resolution of the DGGS cells). Figure 4 illustrates an example of image sampling using rhombuses.

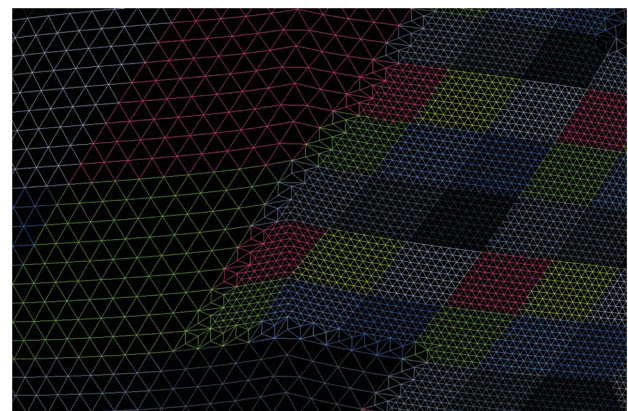


Figure 3. A zoomed-in view of the Earth rendered using rhombuses



Figure 4. (a) A globe textured using rhombuses where one of the rhombuses has been textured differently to display its relationship to the original hexagonal cells; (b) A view of neighbouring textured rhombuses where the red lines highlight the boundary of one of the rhombuses; (c) A close-up look at the hexagonal sampling of the textures
Source: Blue Marble.

IMAGE DATA TRANSMISSION: THE HAAR WAVELET

With a client-server DGGs architecture, imagery data sets are sampled and assigned on a server, then sent over a network to clients. However, sending the entire data set can be quite slow due to the potentially large volume of the data.

Wavelets can be used to design lossless compression techniques, which send an initial approximation to the data that may be corrected as additional information arrives over time. One of the simplest, yet most 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} ,



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

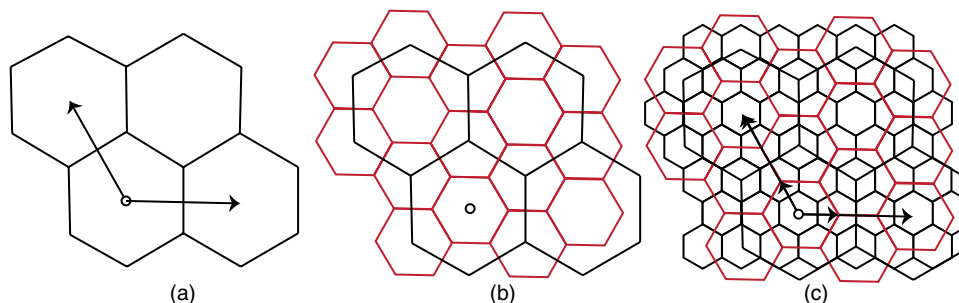


Figure 6. (a) A set of hexagons and a coordinate system defined on them; (b) The coarse hexagons in (a) are refined using 1-to-3 refinement; (c) After two iterations of 1-to-3 refinement, the fine hexagons are aligned with the coarse hexagons in (a)

in a process called decomposition, which may be applied to images by performing the decomposition both rowwise and columnwise. The details corresponding to the decomposition are found as the differences 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 $f_{2i+1} = c_i + d_i$ (see Figure 5). As is readily apparent, the dimension of the coarse data and their associated details is the same as that of the original high-resolution data set (i.e., no additional information/storage space is needed).

The Haar wavelet is a binary wavelet, as the dimension of the low-resolution data is half that of the high-resolution data. However, this transform is not readily compatible with an A3H DGGs, for which 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 (Minasyan, Stankovic, and Astola 2009).

TERNARY HAAR WAVELET

In the ternary Haar wavelet transform (Minasyan and others 2009), 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; that is,

$$c_i = \frac{f_{3i} + f_{3i+1} + f_{3i+2}}{3} \text{ associates two details } d_{2i} \text{ and } d_{2i+1}$$

with c_i instead of just one, which are individually denoted as $d_{2i} = c_i - f_{3i}$ and $d_{2i+1} = c_i - f_{3i+1}$. In the reconstruction process, high-resolution data values are reconstructed via $f_{3i} = c_i - d_{2i}$, $f_{3i+1} = c_i - d_{2i+1}$, and $f_{3i+2} = c_i + d_{2i} + d_{2i+1}$. Algorithms 1 and 2 outline the decomposition and reconstruction processes, respectively, of the ternary Haar wavelet. In general, much as the ternary Haar wavelet is appropriate for an A3H DGGS, an n -ary Haar wavelet ought to be used with a DGGS of aperture n . A description (and proof of correctness) for general n -ary Haar wavelets is provided in the Supplementary Material.

Algorithm 1. Decomposition of a single row of an image f (of n pixels) into c and d via the ternary Haar wavelet

for $i = 0$ to $\frac{n}{3}$, step 3 **do**

$$c_i = \frac{f_{3i} + f_{3i+1} + f_{3i+2}}{3}$$

$$d_{2i} = c_i - f_{3i}$$

$$d_{2i+1} = c_i - f_{3i+1}$$

end for

Algorithm 2. Reconstruction of a single row of an image f (of n pixels) using c and d via the ternary Haar wavelet

for $i = 0$ to $\frac{n}{3}$, step 3 **do**

$$f_{3i} = c_i - d_{2i}$$

$$f_{3i+1} = c_i - d_{2i+1}$$

$$f_{3i+2} = c_i + d_{2i} + d_{2i+1}$$

end for

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 after the averaging process. While this is not a problem for elevation data (as such data are naturally available in floating point format), imagery data sets are represented in integer format. Therefore, when the Haar wavelet transform is applied, the data must be truncated 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.

INTEGER TERNARY HAAR WAVELET

As proposed in (Wendler 1987), integer wavelets restrict the high-resolution data, low-resolution data, and details

to remain as integers throughout the entire multiresolution process. To define an integer ternary Haar wavelet, we modify the ternary Haar wavelet in such a way that all details and low-resolution data values become integers. Consider $d_{2i} = c_i - f_{3i}$ and $d_{2i+1} = c_i - f_{3i+1}$. If we substitute

$$c_i = \frac{f_{3i} + f_{3i+1} + f_{3i+2}}{3} \text{ into these relationships, we get}$$

$$d_{2i} = \frac{-2f_{3i} + f_{3i+1} + f_{3i+2}}{3} \text{ and } d_{2i+1} = \frac{f_{3i} - 2f_{3i+1} + f_{3i+2}}{3}.$$

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 integers, so we save

$$\tilde{d}_{2i} = -2f_{3i} + f_{3i+1} + f_{3i+2} \text{ and } \tilde{d}_{2i+1} = f_{3i} - 2f_{3i+1} + f_{3i+2}.$$

In the reconstruction process, f_{3i} and f_{3i+1} are reconstructed

$$\text{quite simply as } f_{3i} = \tilde{c}_i - \left\lfloor \frac{\tilde{d}_{2i}}{3} \right\rfloor \text{ and } f_{3i+1} = \tilde{c}_i - \left\lfloor \frac{\tilde{d}_{2i+1}}{3} \right\rfloor.$$

Finally, given f_{3i} , f_{3i+1} , and \tilde{d}_{2i} , f_{3i+2} is reconstructed as

$$\tilde{d}_{2i} + 2f_{3i} - f_{3i+1}.$$

These decomposition and reconstruction processes are presented in Algorithms 3 and 4, respectively. Using this wavelet transform, we can compress imagery data sets in a manner compatible with an A3H DGGS (see Figure 7).

Algorithm 3. Decomposition of a single row of an image f (of n pixels) into c and d via the ternary Haar wavelet

for $i = 0$ to $\frac{n}{3}$, step 3 **do**

$$\tilde{c}_i = \left\lfloor \frac{f_{3i} + f_{3i+1} + f_{3i+2}}{3} \right\rfloor$$

$$d_{2i} = -2f_{3i} + f_{3i+1} + f_{3i+2}$$

$$d_{2i+1} = f_{3i} - 2f_{3i+1} + f_{3i+2}$$

end for

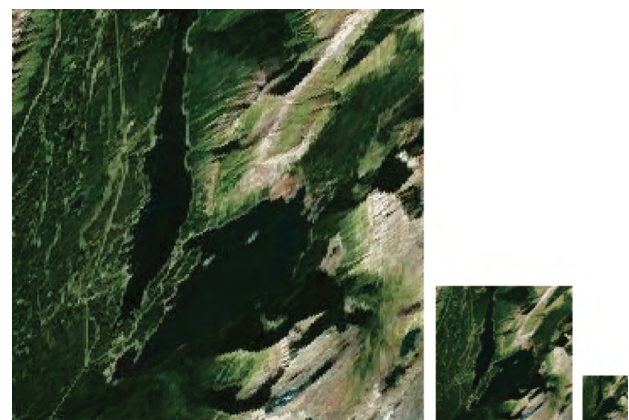


Figure 7. Applying the ternary Haar wavelet to an image two times

Source: Microsoft® Bing™ Maps. Microsoft product screen shot(s) reprinted with permission from Microsoft Corporation.

Algorithm 4. Reconstruction of a single row of an image f (of n pixels) using c and d via the ternary Haar wavelet

for $i = 0$ to $\frac{n}{3}$, step 3 **do**

$$f_{3i} = \tilde{c}_i - \left\lfloor \frac{\tilde{d}_{2i}}{3} \right\rfloor$$

$$f_{3i+1} = \tilde{c}_i - \left\lfloor \frac{\tilde{d}_{2i+1}}{3} \right\rfloor$$

$$f_{3i+2} = d_{2i} + 2f_{3i} - f_{3i+1}$$

end for

To examine the behaviour 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 80% of the original data for reconstruction. The comparison shows that the integer ternary Haar wavelet is comparable with the integer binary Haar wavelet, as the PSNRs of both methods are very close. Table 1 provides the PSNRs of both methods for the images shown in Figure 8.

It is possible to extend the ternary integer Haar wavelet to an arbitrary n -ary integer Haar wavelet for use with DGGs of aperture n . The extension to n -ary integer Haar wavelets and their proof of correctness are provided in the Supplementary Material.

Vector Data Sets

Vector (or feature) data sets are defined as polylines, points, or polygons that describe geospatial features, such as road networks or the boundaries of countries and cities.

Typically, the points making up these vectors exist on a sphere and are connected by geodesic arcs (i.e., great circle arcs), as geodesic arcs traverse the shortest paths between two points on the surface of a sphere. Since these data sets may be very large, consisting, for example, of millions of points, it is necessary to utilize a representation of them 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 rasterizing the vector into an image (rasterization), by association with cells at different resolutions (hierarchical cell representation), or by using wavelet transforms on the feature curves (spherical vector representation). In the following, we describe each method in the context of an A3H DGGs and its benefits for different applications.

RASTERIZATION

One approach that can be used to represent, store, transmit, and visualize vector data sets is to rasterize them into an image. This is similar to texture-based approaches for attaching vector data to a terrain (Kersting and Dollner 2002). Like the images described under *Converting from DGGs Cells to Images*, the pixels of these images each share a one-to-one correspondence with a cell in the DGGs, and

Table 1. PSNRs for the images in Figure 8 under integer ternary and binary Haar wavelets

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

Note: The three rows per method provide the PSNR of each colour channel: red, green, and blue.

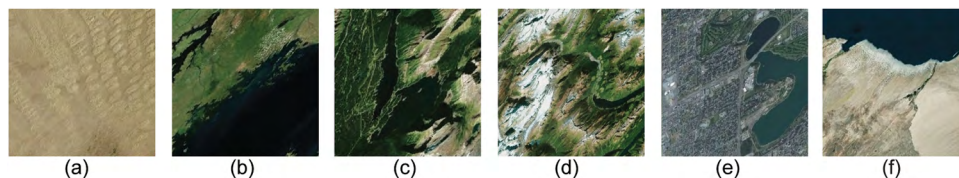


Figure 8. Test pictures used in Table 1

Source: Microsoft® Bing™ Maps. Microsoft product screen shot(s) reprinted with permission from Microsoft Corporation.

can be assigned to the DGGS using the dual conversion. Furthermore, as in *Imagery and Elevation Data Sets*, the resulting images can then be overlaid on the globe for visualization and can be effectively compressed and transmitted through the network using known techniques for image compression and transmission (Shukla, Alwani, and Tiwari 2010).

In order to generate images that they have the same dimensions as the cells in an A3H DGGS, the vectors are sampled using hexagons in a $3'' \times 3''$ rhombus-shaped tiling that is large enough to contain the features (where n depends on the resolution of the corresponding DGGS cells). The cells that contain parts of the feature receive a colour, while other cells are made transparent (see Figure 9). This technique can be used with DGGSs with different apertures if the dimensions of the images are modified accordingly.

While the cells representing a feature can 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.

HIERARCHICAL CELL REPRESENTATION

In a hierarchical cell representation, the cell structure of a DGGS is used directly to obtain a representation for vector data sets. The basic approach can be illustrated using quad trees, which are widely used to approximate features on an image (Samet 2005). Given a quad tree, its set of cells may be refined recursively until a good approximation of the feature is achieved. A cell is refined if it is intersected by the feature, such as the grey cell in Figure 10. If the cell is fully inside or outside of the feature, or if the cell size shrinks past a particular threshold, it is not refined any more.

An important distinction between quad trees and A3H DGGSs lies in the relationship between cells at different resolutions. Cells in an A3H DGGS are hexagonal, and therefore parent cells and their children are incongruent. To solve this issue, we use a hierarchical tree structure compatible with PYXIS indexing (refer to *Data Storage and Retrieval*). While the construction of the nodes of the tree is very similar to the quad tree method of Samet

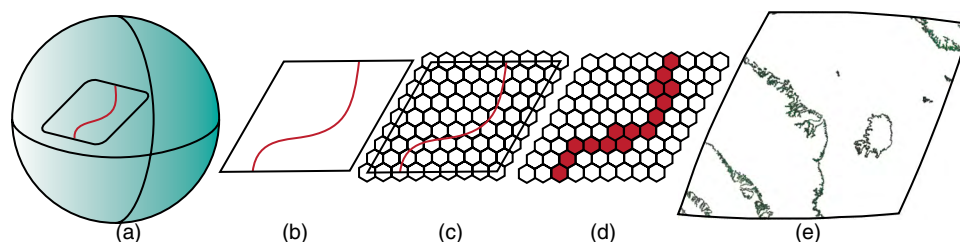


Figure 9. (a) Red feature and the rhombus enclosing this feature on the sphere; (b) Projecting the feature and the rhombus to the 2D rhombus; (c) Sampling the feature using hexagonal cells; (d) Cells sharing a feature are coloured; (e) A real example of a rasterized geospatial feature

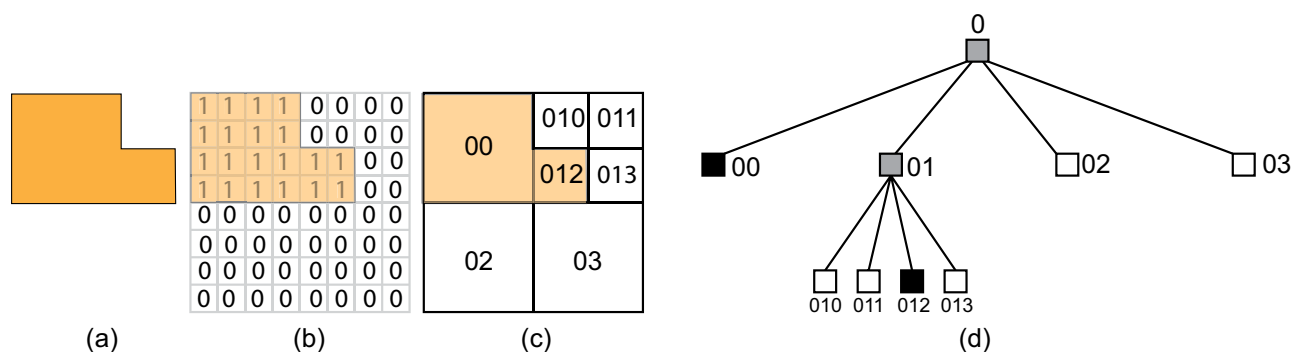


Figure 10. (a) A simple feature; (b) Cells that are inside or outside of the feature are assigned a number, 1 or 0, respectively; (c) Refinement of cells is used to approximate the feature; (d) Trees and colouring of the nodes. Source: Image is a reproduction of an image in Samet (2005).

(2005), the refinement is instead a hexagonal 1-to-3 re-refinement (see Figure 11) with incongruent parent and child cells.

Deciding whether a cell is completely inside or outside a feature is 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 is approximated using two circles with different radii (see Sahr 2008 for more details). Figure 12 illustrates the maximum and minimum radii for type B cells. If the circle associated with a cell lies completely outside or inside the feature, the cell is not refined anymore. Otherwise, it is considered to be a grey cell and refinement continues (refer to Algorithm 5).

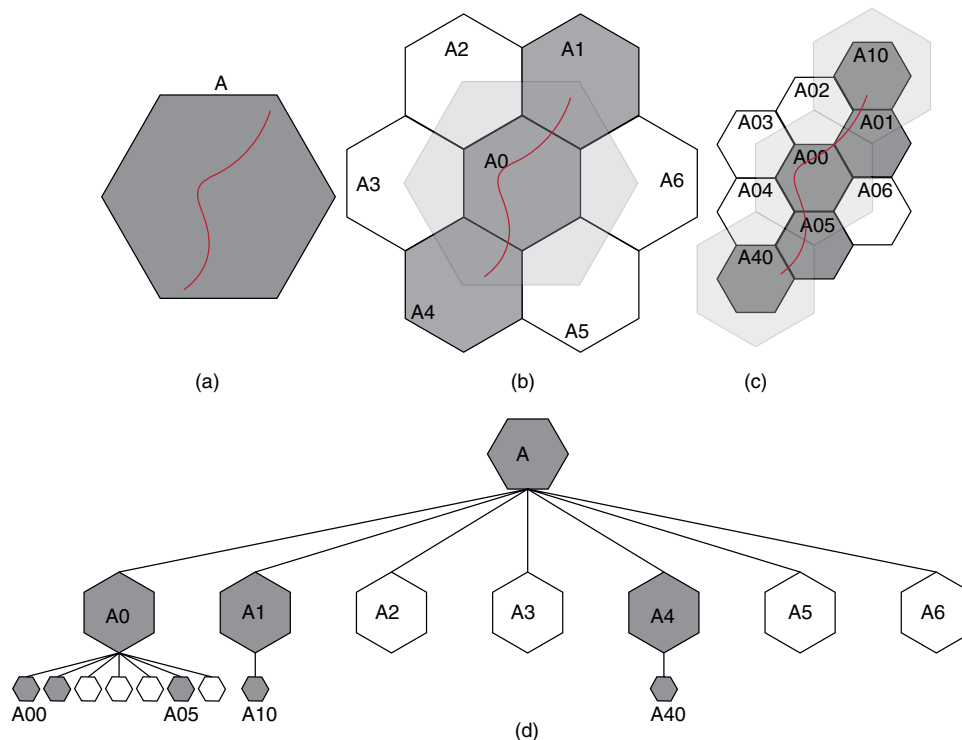


Figure 11. (a) Red feature on a coarse hexagon with index A; (b) and (c) Cells are refined to approximate the feature; (d) Hierarchical tree associated with the refinement process and leaves store the geometry of the curves if they contain points less than a threshold

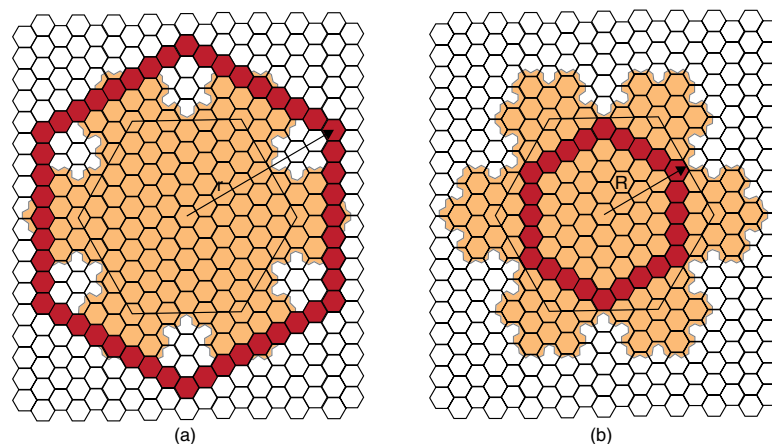


Figure 12. Minimum and maximum radius for the fractal coverage of type B cells

Algorithm 5. Hierarchical cell rasterization of a feature F into the cells of an A3H DGGS

The maximum and minimum radii associated with a cell c are denoted as $r(c)$ and $R(c)$, respectively.

```

for each cell  $c_0$  at the coarsest resolution do
   $cellsToCheck = \{c_0\}$ ,  $doubleCheck = \{\}$ 
  for each cell  $c$  in  $cellsToCheck$  do
    Remove  $c$  from  $cellsToCheck$ 
     $d =$  the distance from  $F$  to the centroid of  $c$ 
    if  $d > (c)$  then
      if the centroid of  $c$  is in  $F$  then
        Mark  $c$  as black (inside the feature)
      else
        Mark  $c$  as white (outside the feature)
      end if
    else
      if  $d \leq R(c)$  then
        Mark  $c$  as grey (intersected by the feature)
      else
        Add  $c$  to  $doubleCheck$ 
      end if
      if  $c$  has children then
        Add the children of  $c$  to  $cellsToCheck$ 
      else
        (Optional) points = the points in  $F$  that are contained in  $c$ 
        (Optional) Store points in  $c$ 
      end if
    end if
  end for
end for
for each cell  $c$  in  $doubleCheck$  do
  if any descendant of  $c$  is not white then
    Mark  $c$  as grey (intersected by the feature)
  else
    Mark  $c$  as white (outside the feature)
  end if
end for

```

This method produces rasterizations of the vector data, similarly to the rasterization approach described above, but at all resolutions of the DGGS instead of at a single resolution. However, no raster approach can perfectly capture the original vector geometry; hence inaccuracies will still result. To avoid this, the geometry of the vectors may be stored at the leaves of the tree. Then, if a portion of a high-resolution curve needs to be visualized at its native resolution, only those feature points in the area of interest need to be retrieved, while the remaining portions can

be rendered in raster form by cutting the tree at a coarse depth.

MULTI-SCALE SPHERICAL REPRESENTATION

A third approach to representing a feature curve is to provide a multi-scale representation/wavelet transform of the curve itself, rather than build a separate hierarchical structure on top of the feature curve. This is a general approach that can be used with any type of DGGS as an independent data structure, and is not specific to A3H DGGS.

One natural approach is to project the feature curve into an intermediate 2D domain (e.g., a map addressed using latitude/longitude coordinates θ, φ) using a spherical projection χ (see Figure 13). In this way, a feature curve can be projected to a 2D curve with coordinates in (θ, φ) , on which we can apply a standard wavelet transform (e.g., the Haar wavelet; Samavati and Bartels 1999). Such a wavelet transform maps fine points f (the feature curve) to coarse points c , where $f_{i \in f}$ has coordinates (θ_i, φ_i) and $c_i \in c$ has coordinates $(\hat{\theta}_i, \hat{\varphi}_i)$. To obtain a coarse spherical curve, we can inverse project the coarse points c to the spherical domain using the inverse projection χ^{-1} .

However, as linear operations on geodetic coordinates (θ, φ) do not take into account the non-linear domain in which the coordinates reside, undesired distortions are produced. In particular, the shortest path between two points on the Earth generally maps to a curve in the (θ, φ) domain, and their midpoint on the Earth does not necessarily map to their midpoint in the (θ, φ) domain. This is especially true for features that cover large areas of the Earth, features whose points are located at the singularities of χ , and features that cross the boundaries of the (θ, φ) domain.

To avoid such artifacts, we can use spherical wavelet transforms to decompose and reconstruct feature curves directly in the spherical domain. A geometric construction for wavelet transforms on feature curves was introduced by Alderson and others (2016) based on a modified Lane-Riesenfeld algorithm (Lane and Riesenfeld 1980). As the construction is composed entirely of SLERP (spherical linear interpolation) operations, it is possible to increase or decrease the resolution of spherical curves without using intermediate domains (see Figure 14), and thus avoid any distortions due to projection mappings. A spherical Haar wavelet transform is described in Algorithms 6 and 7.

Algorithm 6. Decomposition of a feature curve f (of n points) into c and d via the spherical Haar wavelet

```

for  $i = 0$  to  $\frac{n}{2}$ , step 2 do
   $c_i = SLERP(f_{\epsilon >}, f_{\epsilon > \infty}, \frac{1}{2})$ 
   $d_i =$  the rotation from  $f_{2i}$  to  $c_i$ 
end for

```

Algorithm 7. Reconstruction of a feature curve f (of n points) into c and d via the spherical Haar wavelet

for $i = 0$ to $\frac{n}{2}$, step 2 **do**

$f_{2i} = \tilde{c}_i$ rotated by d_i^{-1}

$f_{2i+1} = \tilde{c}_i$ rotated by d_i

end for

As with other wavelet transforms, this approach is lossless and can be used to reconstruct a feature vector perfectly after decomposition. Furthermore, as it operates on local

neighbourhoods, one can build a good coarse approximation of the feature and then reconstruct portions of the curve in full detail on demand. It is therefore useful for rendering spherical curves on top of a textured globe, both for displaying the overall shape of the feature at a distance and for displaying a portion of the feature in detail when zoomed in on (see Figure 15). Such a multi-scale representation of geospatial vector curves has the potential to additionally prove beneficial in handling queries such as offsetting on the vector data (Alderson, Mahdavi-Amiri, and Samavati 2018).

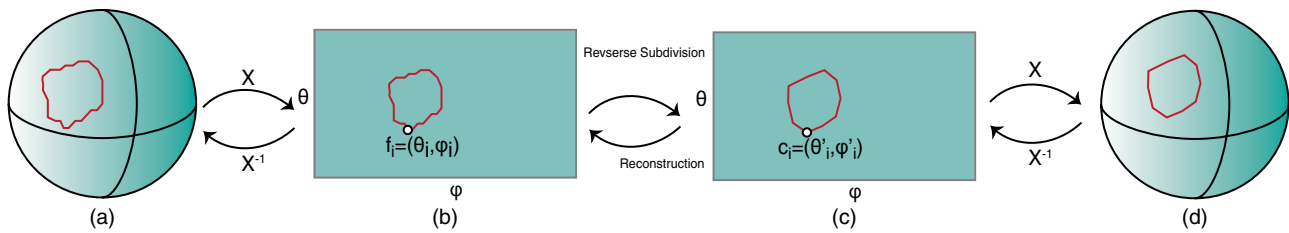


Figure 13. (a) A feature curve located on the sphere; (b) Projecting the sphere and the curve to the 2D domain (θ, ϕ) using the function χ ; (c) Using a wavelet decomposition, we can generate a low-resolution version of the curve with vertices c and using reconstruction, we can reconstruct the high-resolution curve f ; (d) The low-resolution curve can be inverse-projected to the sphere to obtain a coarse feature curve the using function χ^{-1}

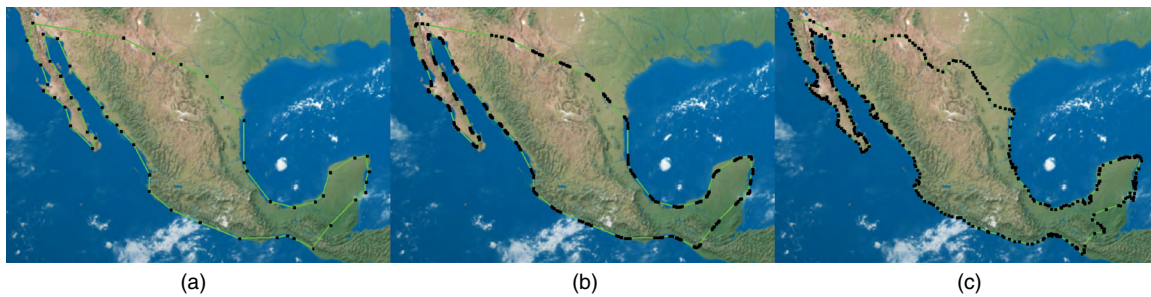


Figure 14. Progressive refinement of a geospatial feature curve: (a) coarse feature curve; (b) and (c) subdivision of the feature in (a) without and with details

Source: Courtesy of Tom Patterson, *Natural Earth III* (www.shadedrelief.com).



Figure 15. Adaptive refinement of a geospatial feature curve: (a) coarse feature curve and (b) the curve can be locally reconstructed to full detail as necessary

Source: Courtesy of Tom Patterson, *Natural Earth III* (www.shadedrelief.com).

SELECTING A REPRESENTATION

Each of these representations has its own advantages and disadvantages, and therefore selecting an appropriate vector data representation depends a great deal on the queries one expects to encounter. For instance, rasterization provides image representations of feature vectors, which are compatible with the cell structure of a DGGs and can make use of the efficient transmission and visualization techniques that exist for images. However, rasterized vectors have a fixed resolution, and performing geometric queries on these representations can be difficult and inaccurate. Specifically, the error can be as large as the radius of the largest cell at the target resolution.

Cell-based representations, in comparison, can efficiently support vectors up to any resolution supported by a given Digital Earth. Theoretically, this representation offers arbitrarily small error, but as the resolution of a Digital Earth is usually capped at a maximum in practice, the error equates to that of rasterization at the maximum resolution (unless the vector geometry is additionally stored). Furthermore, this representation is very sensitive, as a slight modification of the feature vector has the potential to completely alter the structure of the hierarchical cell tree.

The geometry-based multi-scale spherical representation of features is not sensitive to perturbations and is not resolution-dependent. Furthermore, spherical wavelet transforms support efficient data transmission, geometric query handling, and adaptive reconstruction. When fully reconstructed, this representation provides the same error as the original vector data; however, the coarse approximations are constructed based on a spherical shape for the Earth that is less accurate than conventional ellipsoidal representations such as WGS 84.

Geospatial Quantitative Data Sets

Quantitative data sets encompass a variety of data sets that are usually presented as numerical values. For instance, the average age of the population living in a region, the number of endangered species in a region, and the rates of rain in certain cities are examples of quantitative data sets.

While it is possible to represent these data sets as separate images (for visualization purposes, especially), there are some statistical queries one may wish to run over an entire data set that do not benefit from multi-scale representations over the spatial dimensions. One such query is the range query, in which the number of data values within a particular range is requested. Hence, it can be useful to consider multiscale representations for this additional non-spatial dimension of the data, which may be applied in conjunction with or separately from the spatial dimensions. While the Haar wavelet may be used for this purpose, an important goal in defining such a representation is to allow queries to be asked at the data's coarse

scale without significant loss in accuracy, so that a user may be presented with a fast estimate while awaiting the final result.

In this section, we discuss some methods for representing quantitative data sets in a multi-scale manner that are specifically tailored toward addressing range queries at different scales. These methods are general and can be used for any DGGs, but can be adapted for a DGGs with a particular refinement (e.g., an A3H DGGs) to establish a correspondence between the DGGs resolutions and the method structure. Note that as the methods outlined in this section are based on the Haar wavelet, using an n -ary Haar wavelet (see the Supplementary Material) allows us to extend this adaptation to DGGs of other apertures.

WAVELET HISTOGRAM

The first approach to managing quantitative data sets we discuss is the wavelet histogram, which has been used in data cubes (Matias, Vitter, and Wang 1998). Suppose that we are given a data set outlining how many people are of a certain age (ranging between one and eight) and we are interested in asking queries such as "How many people are between the ages of one and six?" To answer such queries, in (Matias and others 1998), 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 16). 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 address these queries efficiently. 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 the Haar wavelet is used, higher-level nodes are formed by averaging two consecutive nodes.

Given such a tree, range queries can be estimated without accessing the entire data set (for instance, if the data set has not finished transmitting to a client). In the wavelet histogram, each node at any level of the tree provides an estimate 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 i th node at the k th 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 16 and we want to answer the query above: "How many people are between the ages of

six and one?” Since age six belongs to the right node of the tree at the second level ($4 \leq 6 \leq 8$) and age 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 16c).

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 to 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 a specific 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 the error is not controllable.

RANGE TREES

An alternative method for classifying quantitative data sets and addressing range queries is known as the range tree, which is a modified versions of the nLT histogram (Chakrabarti and others 2001). At each node of an nLT histogram, 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 records the frequency of the data. As demonstrated in Figure 17a, the root of the tree describes the range of the entire data set and the sum of all frequencies. Each child node bisects the range of its parent and contains the frequency over its bisected range. Range trees differ from nLT histograms in one respect: in order to save memory, if the frequency of a node is zero or one, or if all of the numbers associated with a node are the same, bisection is not applied to the node (see Figure 17b). To further optimize the structure, if a node is missing either its left or right branch, the node is replaced by its child.

While this structure can support floating point numbers (unlike wavelet histograms), each node stores two floating point numbers and one integer, in comparison to the one float per node required by a wavelet histogram and associated detail tree. Hence, the number of data used by range trees is greater than that used by wavelet histograms. A comparison between the amounts of memory required by range trees and wavelet histograms is illustrated in Figure 18. For the data sets tested, with ranges from (0, 1024) to (0, 8192), the wavelet histograms required much less memory than the range trees.

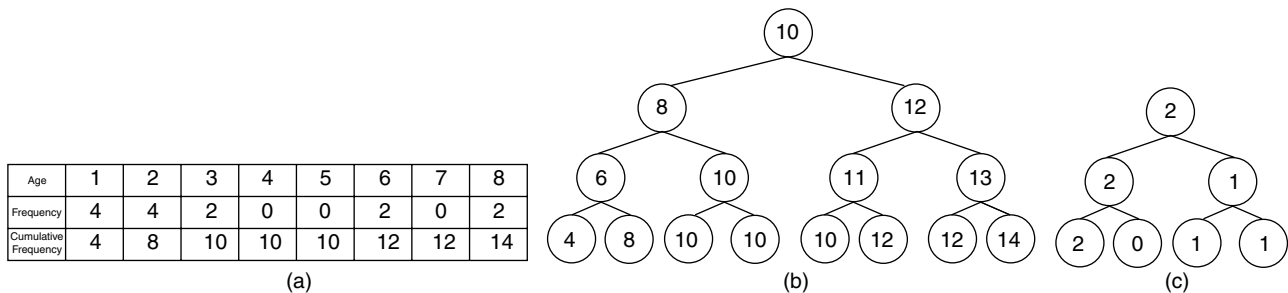


Figure 16. (a) Age data set and cumulative frequency; (b) Haar wavelet histogram; (c) Detail tree associated with the wavelet histogram in (b)

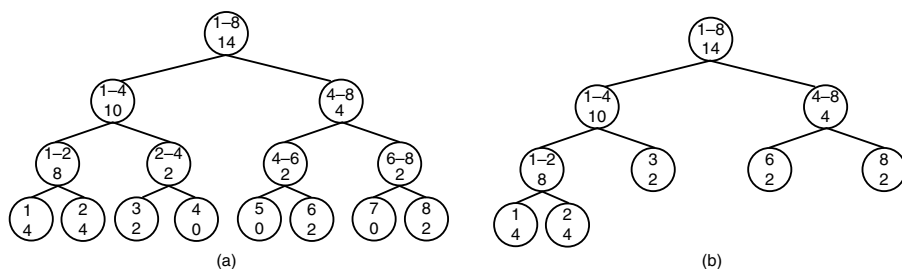


Figure 17. (a) Bisection in an nLT histogram and (b) range tree after optimization

MODIFIED WAVELET HISTOGRAM

With appropriate modifications, 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 $\text{bin_size} = \frac{\text{max} - \text{min}}{2^n}$

handled, and the number of bins is always a power of 2. 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 appropriate min and max based on the type of the data. For instance, given an age distribution 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 themselves. For instance, given a data set of heights (135.5, 157.6, 165, 190.5, 199.5) (see Figure 19), 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 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 in selecting min and max.

The choice of 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 2, so that a complete wavelet histogram can be built. For example, in Figure 19, $n = 3$ and bin_size = 8; hence 8 bins are created.

Although the Haar wavelet histogram is simple and can be improved upon in this way to better meet the needs of data representation in a Digital Earth framework, it does not provide any information about the error at any level of the tree,

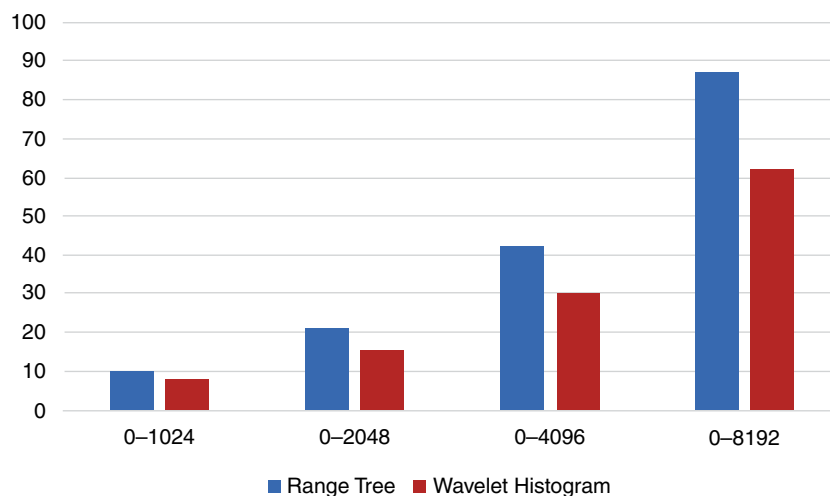


Figure 18. A comparison between the memory sizes of range trees and wavelet histograms for different data set ranges, which is clear that the wavelet histogram requires less space to store the entire data structure

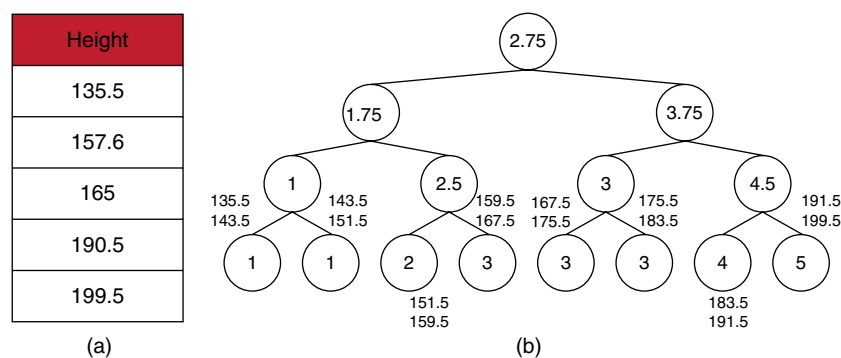


Figure 19. (a) Data set for the height of five people; (b) The wavelet tree created by forming the bins with size 8
Note: The range of each bin is shown beside each leaf node.

nor can the error be controlled. Here, the error, since each node of the tree summarizes all of its descendants, is the maximum difference between the value of the parent node and the values of its leaf descendants. (For instance, the error of the node with value 2.5 in Figure 19, whose children have values of 2 and 3, is 0.5.) In order to overcome this issue, we have developed a novel method called the least-squares wavelet.

LEAST-SQUARES WAVELET

In the least-squares wavelet, we first approximate data with a piecewise linear function 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 propose that these data points be approximated by piecewise linear functions (connected by control points $p_i, 0 \leq i < m$) by solving a least-squares system. 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 $(\Phi^T \Phi)^{-1} \Phi^T f$. Figure 20 illustrates a simple example, featuring a single-piece linear function to approximate the data from Figure 16.

The error (or residual) of this approximation is given by $r = f - \Phi p$. For example, in Figure 20, solving a least-squares system for two points ($m = 2$) produces $p_0 = 3.166$ at age 1 and $p_1 = 0.337$ at age 8, which form a line passing through the data set. To evaluate the error, we can subtract the data set values from corresponding values on the linear function $l(t) = (1 - t)p_0 + tp_1$. Therefore, the error for the i th data value, f_i ($0 \leq i \leq 7$), is $r_i = f_i - l(\frac{i}{7})$ (see Figure 20e).

To evaluate the total error, we can use different norms, but the least-squares solution minimizes norm-2, $\|r\|_2 = \sqrt{r_0^2 + \dots + r_n^2}$. By increasing the number of control points, m , we can reduce the total error and obtain a better approximation of the data. Therefore, by evaluating $\|r\|_2$, we can 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 we must determine 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 21). We can then gradually send nodes of the tree at different levels and improve the initial results obtained from the least-squares solution.

Formally, suppose that we are given the residual r and the operators of a known wavelet transform: the decomposition operators (a reverse subdivision operator A and detail

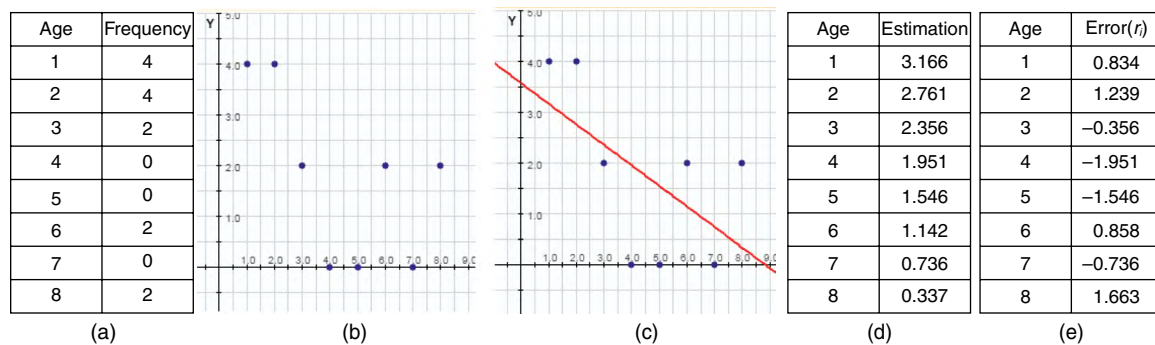


Figure 20. (a) Ages of eight people; (b) Plot of the data set in (a); (c) The least-squares line passing through the data set; (d) Estimation of the data set in (a) by evaluation of the linear function in (c); (e) The error, or residual, of the data evaluation

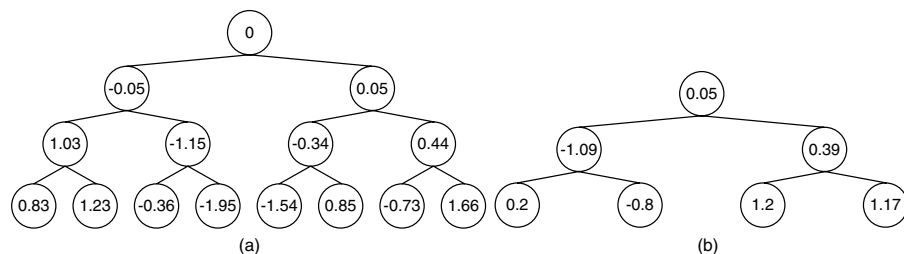


Figure 21. (a) Haar wavelet tree for residuals and (b) detail tree for residuals

calculation operator B) and the reconstruction operators (a subdivision operator P and detail restoration operator Q). For the sake of convenience, let S^i , Q^i , A^i , and B^i be, respectively, the matrices corresponding to i applications of the operators P , Q , A , and B . We can decompose r into a coarse residual $c^{[k]} = A^k r$ and details $d^{[i]} = B^i r$ ($i = k, k-1, \dots, 0$); see Algorithm 8 for the case of the Haar wavelet.

Note that the residuals can be approximated at different resolutions using $c^{[i-1]} = P c^{[i]} + Q d^{[i]}$ (with $c^{[0]} = r$); see Algorithm 9 for the reconstruction of r and f . Although the coarse residuals $c^{[i]}$ (for $i \neq 0$) have fewer entries than r , $c^{[i]}$ can be repeatedly subdivided to obtain a residual approximation $\hat{r} = P^i c^{[i]}$. As the root of the Haar wavelet histogram, $c^{[k]}$, is the average of the residuals (which are always zero), $c^{[k]}$ is not needed to reconstruct the residuals.

Algorithm 8. Decomposition of r into d_k, d_{k-1}, \dots, d_0 ($c_k = 0$)

```

for  $j = 0$  to  $k$ , step 1 do
   $temp = \text{new vector of size } \frac{\text{sizeof}(r)}{2}$ 
  for  $i = 0$  to  $\text{sizeof}(r)$ , step 2 do
     $temp\left(\frac{i}{2}\right) = \frac{r(i) + r(i+1)}{2}$ 
     $d_j\left(\frac{i}{2}\right) = r(i) - temp\left(\frac{i}{2}\right)$ 
  end for
end for

```

Algorithm 9. Reconstruction of f after receiving d_k, d_{k-1}, \dots, d_0

```

for  $j = k$  to  $0$ , step-1 do
   $temp = \text{new vector of size } 2 \times \text{sizeof}(r)$ 
  for  $i = 0$  to  $\text{sizeof}(r)$ , step 1 do
     $temp(2i) = r(i) - d(i)$ 
     $temp(2i+1) = r(i) + d(i)$ 
  end for
end for
 $f = \Phi p + r$ 

```

An important property resulting from using Haar wavelets to generate the residual tree is that the error of the approximation is bounded. That is, for a given $0 \leq i \leq k$ (with $\hat{r} = P^i c^{[i]}$), $\|r - \hat{r}\| \leq \|r\|$. To prove this, consider (without loss of generality) the first entry of $c^{[i]}$, which we will denote as c . By construction of Haar multiresolution, c is the average of the first 2^i entries of r , and the first 2^i entries of \hat{r} are duplicates of c . Now we can see that

$$c^2 \leq 2c^2,$$

$$c^2 \leq 2c \frac{r_0 + r_1 + \dots + r_{2^i-1}}{2^i},$$

$$2^i c^2 \leq 2cr_0 + 2cr_1 + \dots + 2cr_{2^i-1},$$

$$(c^2 - 2cr_0) + (c^2 - 2cr_1) + \dots + (c^2 - 2cr_{2^i-1}) \leq 0.$$

By adding $r_0^2 + r_1^2 + \dots + r_{2^i-1}^2$ to both sides, this becomes

$$(r_0 - c)^2 + (r_1 - c)^2 + \dots + (r_{2^i-1} - c)^2 \leq r_0^2 + r_1^2 + \dots + r_{2^i-1}^2$$

As this was found without loss of generality, it holds for all entries of $c^{[i]}$, and hence for all entries of \hat{r} and r . Therefore, $\|r - \hat{r}\| \leq \|r\|$. Table 2 illustrates an example in which the error has been reduced by adding the nodes of the Haar residual tree to the data set (the data are the same as in Figure 22).

REFINEMENT EXTENSION

The wavelet histogram tree and its associated binning method, as discussed above, are based on a binary Haar wavelet. Although a wavelet histogram can be built on the data independent of the underlying DGGS, it is also possible to adopt the tree and cell structure of the DGGS. In this case, the same factor of refinement as for the DGGS can also be used for the wavelet histogram tree. For instance, given an A3H DGGS, a ternary Haar wavelet (in which each coarse node is found as the average of its nine children) can be used to build a wavelet histogram tree or range tree that is compatible with the DGGS cell structure (see Figure 23). For DGGSs with different refinements (aperture n), it is also possible to use the appropriate n -ary Haar wavelet to construct the wavelet tree (see the Supplementary Material for the general Haar wavelet) or a range tree. Establishing a similar structure between these trees and the DGGS cell hierarchy supports the possibility of forming a correspondence between the resolution of the DGGS and the accuracy of the approximated data.

Table 2. Adding the nodes of the Haar residual tree at different resolutions reduces the error

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

Note: There is no change at the first resolution, since the average of the residuals is always zero.

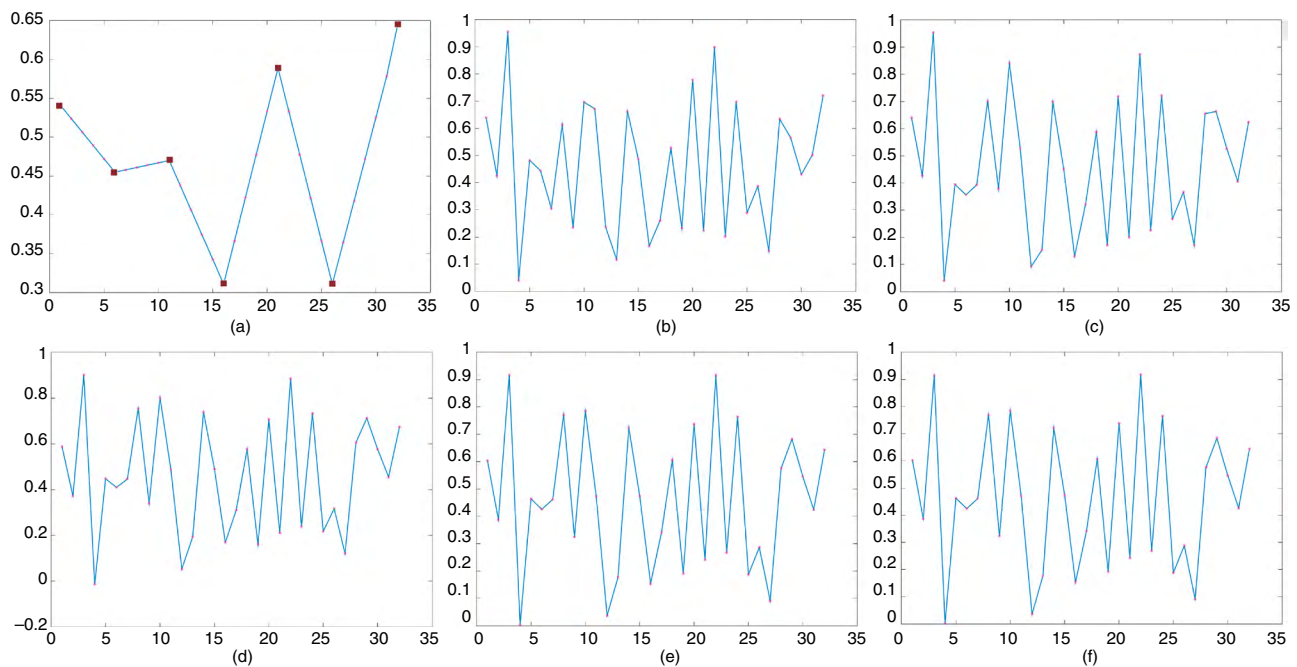


Figure 22. (a) Estimating the data set using piecewise linear functions. Control points are shown as squares; (b)–(e) Adding a portion of the residual to the piecewise linear functions using a Haar wavelet tree, until the data set is perfectly reconstructed in (f)

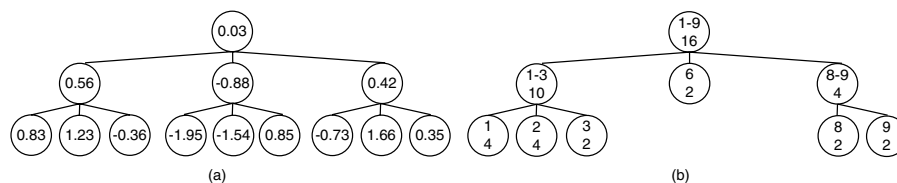


Figure 23. (a) Ternary Haar wavelet tree and (b) Ternary range trees

Conclusions and Discussion

We have quantified the performance of our Haar wavelet transform in data transmission for imagery data sets. We have selected a region located at (49.28°N, 123.11°W) that is composed of 81 rhombuses (see Figure 24). Each rhombus has dimensions 81×81 and is about 12 KB. Table 3 reports the time needed to transmit and receive the images associated with the region after two operations of Haar wavelet. The results clearly show that the smaller images need less time, which is expected, as fewer data are transmitted through the network. Figure 24 also illustrates a visual comparison after the wavelet is applied at two successive resolutions. To have a fair evaluation, we have simulated a client–server network in C# for this experiment to minimize the overhead and delays that might be available on a remote server. The same discussion is also valid for other types of data sets. When we need to transmit fewer

data, the time for transmission is usually reduced by the same factor by which the data sets have been compressed.

In addition to the use of our method in data transmission, we reviewed different techniques (both new and pre-existing) for representing data sets in an A3H DGGS in this article. Some of these methods (such as the least-square wavelet, modified wavelet histogram, and spherical vector representation) are general and can be applied to any DGGS. Our discussions of data representation were divided among different types of data (imagery/elevation, vector, and quantitative data sets) and began with an overview of data storage and retrieval using cell indices. We showed that using wavelets and related appropriate data structures, we can efficiently manage geospatial data sets in a DGGS framework. Additionally, we showed that depending the type of the data sets, we can use wavelets that respects the underlying structure of the data. For instance, if the data are spherical (e.g., vector data sets), we can keep them spherical,

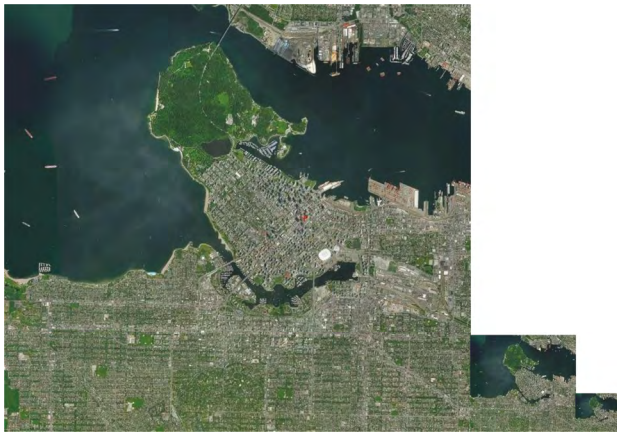


Figure 24. Tested area located at (49.28°N, 123°11 W) composed of 81 rhombuses compressed at two successive resolutions
Source: Map data © 2019 Google.

Table 3. Data transmission test on the imagery data set of a geospatial region

Compression level	Size	Time (send)	Time (receive)
0	1100 KB	2.36 s	2.98 s
1	360 KB	1.01 s	1.21 s
2	120 KB	0.45 s	0.53 s

Note: Compressing imagery data using the Haar wavelet results in a shorter time to send and receive the imagery data set.

or if they are integer (e.g., images), we can keep them in the same format. This is useful, as we can use the same type of data structures and algorithms after applying wavelet transforms. Additionally, we provided a wavelet-based data structure for quantitative data sets to be aware of the amount of error that we are reproducing due to compressing data, and we can always make it better through time by sending more accompanying details. In this article, we have mainly focused on the benefits of wavelets for data transmission. However, there are plenty of queries and applications related to DGGS in which wavelets can be beneficial. For instance, as discussed in (Alderson and others 2018), spherical wavelets can be useful to find the offset of a geospatial feature represented in vector or raster format.

Future Work

There are many potential directions that can be explored to improve upon the methods presented here. For instance, one may wish to determine the performance of these methods with different queries. Finding the intersection or union of two vector data sets as represented in any of the forms described above, or the use of wavelet histograms for

queries other than range queries, are two such examples. In addition, combining different representations of vector data sets, such as a combined multi-scale spherical representation and cell-based representation, is an interesting research path. Finally, designing new multiresolution frameworks besides Haar that can support the compression of quantitative data sets while preserving the quality of the data (i.e., with minimal error) could be interesting for future work.

Acknowledgements

We would like to thank the anonymous reviewers for their thoughtful comments in the review process. We also thank Idan Shatz, Amirhesam Moltaji, and Nader Hamekasi for their insightful discussions and early work on this article. This research was supported in part by the National Science and Engineering Research Council of Canada (NSERC) and the Global Grid System (formerly Pyxis Innovation Inc).

Author Information

Ali Mahdavi Amiri is currently a university research associate in the GrUVi lab at Simon Fraser University, Canada. He received his PhD from the University of Calgary in 2015. His research areas are machine learning, geospatial visualization and representation, geometric modelling, and computational creativity, design, and fabrication. He has received several outstanding grants and awards, including an NSERC Postdoctoral Fellowship, a J.B. Hyne Research Innovation Award, and a Michael A.J. Sweeney Award for the best paper at Graphics Interface, 2013. ORCID: 0000-0002-4693-3565. E-mail: amahdavi@sfu.ca.

Troy Alderson received his BSc and MSc in computer science from the University of Calgary, where he is currently working toward a PhD degree in computer science. His current research interests include subdivision, multiresolution, and geospatial representation. E-mail: tfalders@ucalgary.ca.

Faramarz Samavati is a professor in the Department of Computer Science at the University of Calgary. His research interests include geospatial visualization and representation, computer graphics, visualization, and geometric modelling. He has published more than 90 papers and one book and has filed three patents. E-mail: samavati@ucalgary.ca.

References

Agrawal, Anupam, Radhakrishna Maringanti, and R.C. Joshi. 2006. "Geometry-Based Mapping and Rendering of Vector Data over LOD Phototextured 3D Terrain Models." Paper read at WSCG' 06, 30 January–3 February, Pizen–Bory, Czech Republic.

Alderson, Troy, Ali Mahdavi-Amiri, and Faramarz Samavati. 2016. "Multiresolution on Spherical Curves." *Graphical Models* 86: 13–24. <https://doi.org/10.1016/j.gmod.2016.05.002>.

- Alderson, Troy, Ali Mahdavi-Amiri, and Faramarz Samavati. 2018. "Offsetting Spherical Curves in Vector and Raster Form." In CGI 2018 Conference. *Visual Computer* 34(6–8): 973–84. <https://doi.org/10.1007/s00371-018-1525-7>.
- Auclair-Fortier, Marie-Flavie, Djemel Ziou, and Costas Armenakis. 1999. "Survey of Work on Road Extraction in Aerial and Satellite Images." Center for Topographic Information Geomatics, Ottawa.
- Boyko, Aleksey, and Thomas Funkhouser. 2011. "Extracting Roads from Dense Point Clouds in Large Scale Urban Environment." *ISPRS Journal of Photogrammetry and Remote Sensing* 66(6): S2–S12. <https://doi.org/10.1016/j.isprsjprs.2011.09.009>.
- Chakrabarti, Kaushik, Minos Garafalakis, Rajeev Rastogi, and Kyuseok Shim. 2001. "Approximate Query Processing Using Wavelets." *Vldb Journal* 10(2–3): 199–223. <https://doi.org/10.1007/s007780100049>.
- Garofalakis, Minos, and Phillip B. Gibbons. 2002. "Wavelet Synopses with Error Guarantees." In Proceedings of the ACM SIGMOD International Conference on Management of Data, 3–6 June, Madison, WI, ed. Michael J. Franklin, Bongki Moon, and Anastasia Ailamaki, 476–87. N.p.: ACM.
- Han, Charles, and Hugues Hoppe. 2010. "Optimizing Continuity in Multiscale Imagery." *ACM Transactions on Graphics* 29(6): article 171. <https://doi.org/10.1145/1882261.1866193>.
- Kersting, Oliver, and Jurgen Dollner. 2002. "Interactive 3D Visualization of Vector Data in GIS." In *GIS 2002: Proceedings of the 10th ACM International Symposium on Advances in Geographic Information Systems*, 8–9 November, McLean, VA, 107–12.
- Lambers, Martin, and Andreas Kolb. 2012. "Ellipsoidal Cube Maps for Accurate Rendering of Planetary-Scale Terrain Data." In *Proceedings Pacific Graphics 2012: Short Papers*, ed. C. Bregler, P. Sander, and M. Wimmer, 5–10.
- Lane, Jeffrey M., and Richard M. Riesenfeld. 1980. "A Theoretical Development for the Computer Generation and Display of Piecewise Polynomial Surfaces." *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-2(1): 35–46. <https://doi.org/10.1109/tpami.1980.4766968>.
- Li, Tao, Qi Li, Shenghuo Zhu, and Mitsunori Ogiwara. 2002. "A Survey on Wavelet Applications in Data Mining." *ACM SIGKDD Explorations Newsletter* 4(2): 49–68. <https://doi.org/10.1145/772862.772870>.
- Mahdavi-Amiri, Ali, Troy Alderson, and Faramarz Samavati. 2015a. "A Survey of Digital Earth." *Computers and Graphics* 53(Part B): 95–117. <https://doi.org/10.1016/j.cag.2015.08.005>.
- Mahdavi-Amiri, Ali, Faraz Bhojani, and Faramarz Samavati. 2013. "One-to-Two Digital Earth." In *Advances in Visual Computing, 13th International Symposium, ISVC'13*, Las Vegas, 19–21 November, 681–92. N.p.: Springer.
- Mahdavi-Amiri, Ali, Erika Harrison, and Faramarz Samavati. 2014. "Hexagonal Connectivity Maps for Digital Earth." *International Journal of Digital Earth* 8(9): 750–69. <https://doi.org/10.1080/17538947.2014.927597>.
- Mahdavi-Amiri, Ali, Erika Harrison, and Faramarz Samavati. 2016. "Hierarchical Grid Conversion." *Computer Aided Design* 79: 12–26. <https://doi.org/10.1016/j.cad.2016.04.005>.
- Mahdavi-Amiri, Ali, and Faramarz Samavati. 2012. "Connectivity Maps for Subdivision Surfaces." In *GRAPP & IVAPP 2012: Proceedings of the International Conference on Computer Graphics Theory and Applications and International Conference on Information Visualization Theory and Applications*, 24–26 February, Rome, 26–37.
- Mahdavi-Amiri, Ali, and Faramarz Samavati. 2014a. "Adaptive Atlas of Connectivity Maps." In *Mathematical Methods for Curves and Surfaces 8th International Conference, MMCS 2012*, 28 June–3 July, Oslo. Lecture Notes in Computer Science, Vol. 8177, 304–20. N.p.: Springer.
- Mahdavi-Amiri, Ali, and Faramarz Samavati. 2014b. "Atlas of Connectivity Maps." *Computers and Graphics* 39: 1–11. <https://doi.org/10.1016/j.cag.2013.09.003>.
- Mahdavi-Amiri, Ali, Faramarz Samavati, and Perry Peterson. 2015b. "Categorization and Conversions for Indexing Methods of Discrete Global Grid Systems." *ISPRS International Journal of Geo-Information* 4(1): 320–36. <https://doi.org/10.3390/ijgi4010320>.
- Matias, Yossi, Jeffrey Vitter, and Min Wang. 1998. "Wavelet-Based Histograms for Selectivity Estimation." In *SIGMOD 1998: Proceedings ACM SIGMOD International Conference on Management of Data*, 2–4 June, Seattle, ed. Laura M. Haas and Ashutosh Tiwary, 448–459. N.p.: ACM.
- Mayer, Helmut, Ivan Laptev, and Albert Baumgartner. 1998. "Multi-scale and Snakes for Automatic Road Extraction." In *Computer Vision – ECCV'98*, ed. H. Burkhardt and B. Neumann, Lecture Notes in Computer Science, Vol. 1406, 720–33. Berlin/Heidelberg: Springer. <https://doi.org/10.1007/BFb0055700>.
- Mena, Juan B. 2003. "State of the Art on Automatic Road Extraction for GIS Update: A Novel Classification." *Pattern Recognition Letters* 24(16): 3037–58. [https://doi.org/10.1016/s0167-8655\(03\)00164-8](https://doi.org/10.1016/s0167-8655(03)00164-8).
- Minasyan, Susanna, Radomirand Stankovic, and Jaakko Astola. 2009. "Ternary Haar-Like Transform and Its Application in Spectral Representation of Ternary-Valued Functions." In *Computer Aided Systems Theory – EUROCAST 2009: 12th International Conference*, 15–20 February, Las Palmas de Gran Canaria, Spain, Lecture Notes in Computer Science, Vol. 5717, pp. 518–25. Berlin/New York: Springer.
- Peterson, Perry 2004. "Close-Packed, Uniformly Adjacent, Multi-resolutional, Overlapping Spatial Data Ordering." US Patent PCT/CA2004/001507, March 19.
- Rottensteiner, Franz 2003. "Automatic Generation of High-Quality Building Models from LIDAR Data." *IEEE Computer Graphics and Applications* 23(6): 42–50. <https://doi.org/10.1109/mcg.2003.1242381>.
- Sahr, Kevin 2008. "Location Coding on Icosahedral Aperture 3 Hexagon Discrete Global Grids." *Computers, Environment and Urban Systems* 32(3): 174–87. <https://doi.org/10.1016/j.compenvurbsys.2007.11.005>.
- Sahr, Kevin, Denis White, and Jon A. Kimerling. 2003. "Geodesic Discrete Global Grid Systems." *Cartography and Geographic Information Science* 30(2): 121–34. <https://doi.org/10.1559/152304003100011090>.

- Samavati, Faramarz, and Richard Bartels. 1999. "Multiresolution Curve and Surface Representation: Reversing Subdivision Rules by Least-Squares Data Fitting." *Computer Graphics Forum* 18(2): 97–119. <https://doi.org/10.1111/1467-8659.00361>.
- Samet, Hannan. 2005. *Foundations of Multidimensional and Metric Data Structures*. San Francisco: Morgan Kaufmann Publishers Inc.
- Schilling, Arne, Jens Basanow, and Alexander Zipf. 2007. "Vector Based Mapping of Polygons on Irregular Terrain Meshes for Web 3D Map Services." In *Web Information Systems and Technologies, 3rd International Conference*, 3–6 March, Barcelona, 198–205.
- Schneider, Martin, Michael Guthe, and Reinhard Klein. 2005. "Real-Time Rendering of Complex Vector Data on 3D Terrain Models." In *VSM 2005: Proceedings of the 11th International Conference on Virtual Systems and Multimedia*, 3–7 October, Ghent, 573–82.
- Shukla, Jaya, Manoj Alwani, and Anil Kumar Tiwari. 2010. "A Survey on Lossless Image Compression Methods." Read at the 2nd International Conference on Computer Engineering and Technology, 16–19 April, Chengdu, China.
- Snyder, John P. 1992. "An Equal Area Map Projection for Polyhedral Globes." *Cartographica* 29(1): 20–21. <https://doi.org/10.3138/27h7-8k88-4882-1752>.
- Southern Terra Cognita Laboratory. 2017. "DGGRID." Available at <http://www.discreteglobalgrids.org/software/>.
- Stollnitz, Eric J., Tony D. Deroose, and David H. Salesin. 1996. *Wavelets for Computer Graphics: Theory and Applications*. San Francisco: Morgan Kaufmann Publishers.
- Tong, Xiaochong, Jin Ben, Ying Wang, Youngshen Zhang, and Tao Pei. 2013. "Efficient Encoding and Spatial Operation Scheme for Aperture 4 Hexagonal Discrete Global Grid System." *International Journal of Geographical Information Science* 27(5): 898–921. <https://doi.org/10.1080/13658816.2012.725474>.
- Vince, Andrew. 2006. "Indexing the Aperture 3 Hexagonal Discrete Global Grid." *Journal of Visual Communication and Image Representation* 17(6): 1227–36. <https://doi.org/10.1016/j.jvcir.2006.04.003>.
- Vince, Andrew, and Xiqiang Zheng. 2009. "Arithmetic and Fourier Transform for the PYXIS Multi-resolution Digital Earth Model." *International Journal of Digital Earth* 2(1): 59–79. <https://doi.org/10.1080/17538940802657694>.
- Vitter, Jefferey, and Min Wang. 1999. "Approximate Computation of Multidimensional Aggregates of Sparse Data Using Wavelets." *ACM SIGMOD Record* 28(2): 193–204. doi.org/10.1145/304181.304199.
- Vitter, Jefferey, Min Wang, and Bala Iyer. 1998. "Data Cube Approximation and Histograms via Wavelets." In *Proceedings of the 7th International Conference on Information and Knowledge Management*, 2–7 November, Washington, DC, 96–104. New York: ACM.
- Wendler, Thomas. 1987. "Verfahren für die hierarchische Codierung von Einzelbildern in medizinischen Bildinformationssystemen." PhD thesis, Technische Hochschule, Aachen.
- White, Denis, Jon A. Kimerling, and Scott Overton. 1992. "Cartographic and Geometric Components of a Global Sampling Design for Environmental Monitoring." *Cartography and Geographic Information Science* 19(1): 5–22. <https://doi.org/10.1559/152304092783786636>.
- Williams, Lance. 1983. "Pyramidal Parametrics." *ACM SIGGRAPH* 17(3): 1–11.
- Zhang, Chunlong. 2007. *Fundamentals of Environmental Sampling and Analysis*. Hoboken, NJ: Wiley.
- Zhou, Mengyun, Jing Chen, and Jianya Gong. 2015. "A Virtual Globe-Based Vector Data Model: Quaternary Quadrangle Vector Tile Model." *International Journal of Digital Earth* 9(3): 230–51. <https://doi.org/10.1080/17538947.2015.1016558>.