Entropy **2014**, *xx*, 1-x; doi:10.3390/——

OPEN ACCESS
entropy
ISSN 1099-4300
www.mdpi.com/journal/entropy

Article

CATEGORIZATION AND CONVERSIONS FOR INDEXING METHODS OF DISCRETE GLOBAL GRID SYSTEMS

Ali Mahdavi-Amiri, Faramarz Samavati, Perry Peterson

University of Calgary, PYXIS Innovation Inc.

* Author to whom correspondence should be addressed; amahdavi@ucalgary.ca

Received: xx / Accepted: xx / Published: xx

Abstract: Digital Earth frameworks provide a tool to receive, send and interact with large location-based data sets, organized usually according to Discrete Global Grid Systems (DGGS). In DGGS, an indexing method is used to assign a unique index to each cell of a global grid and the data sets corresponding to these cells are retrieved or allocated using this unique index. There exist many methods to index cells of DGGS. Toward facility, interoperability, and also defining a "standard" for DGGS, a conversion is needed to translate a data set from one DGGS to another. In this paper, we first propose a categorization of indexing methods of DGGS and then define a general conversion method from one indexing to another. Several examples are presented to describe the method.

Keywords: Digital Earth; Indexing Method; Hierarchy; Space Filling Curves

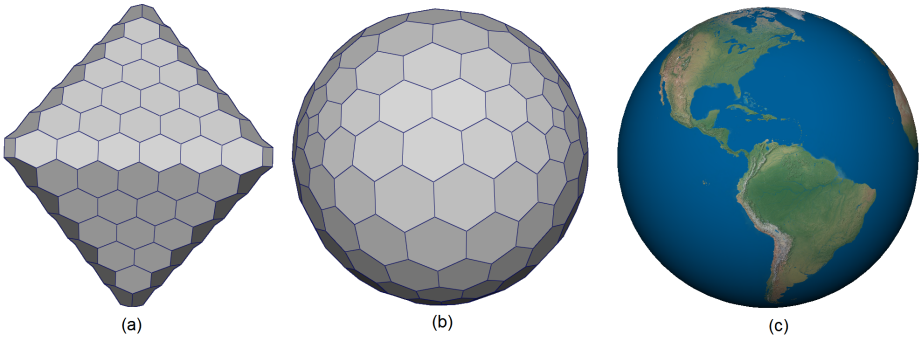
1. Introduction

Digital Earth frameworks provide a multiresolution representation of the Earth as a spatial reference model with the ability to visualize, retrieve, embed, and analyze data at different levels of detail [1]. To assign data to locations and establish a multiresolution representation, the surface of the Earth is discretized using different methods. The traditional method of discretizing the Earth is to use a latitude/longitude (lat-long) parametrization of the sphere. Taking equal length steps along the latitudes and longitudes parametrizes the Earth into quadrilateral cells. These cells have difference sizes and become smaller approaching the poles. In addition, poles are singularities in the lat-long parametrization and cells incident to the poles are triangular.

To obtain a more uniform cell structure with lower areal and angular distortions in order to simplify data analysis, Discrete Global Grid Systems (DGGS) have been proposed [1,2]. In DGGS, the Earth is

approximated by a spherical (or ellipsoidal) polyhedron. Faces of the spherical polyhedron are refined by a specific factor of refinement (aperture) to provide a multiresolution representation. The refined faces of the polyhedron are then projected to the sphere to create cells on the surface of the Earth (see Figure 1). To assign data sets to these cells, a data structure is needed. In DGGS, typically an indexing method is used to assign and retrieve and also handle essential queries. As a result, DGGS differ based on their initial polyhedron, cell type, projection, and indexing method. These DGGS need to communicate and receive, share, and integrate data coming from other DGGS. In essence, interoperability is an important property for these systems particularly in the context of Open Geospatial Consortium (OGC). **Conversion** between DGGS is a crucial requirement for supporting this property. In this paper, we introduce a general conversion method for DGGS. This conversion can be potentially used from a DGGS to a future standard of a DGGS or an OGC standard. Since data sets are associated to DGGS through an indexing method, the conversion is essentially defined on the indexing methods if both DGGS use the same projection. For the general case, the projections are also needed to find the conversion. In the following, we initially discuss DGGS and its elements, and then provide the general conversion for the indexing methods in Section 3.

Figure 1. (a) a refined polyhedron. (b) Projecting the polyhedron to the sphere. (c) Data is assigned to the spherical polyhedron.

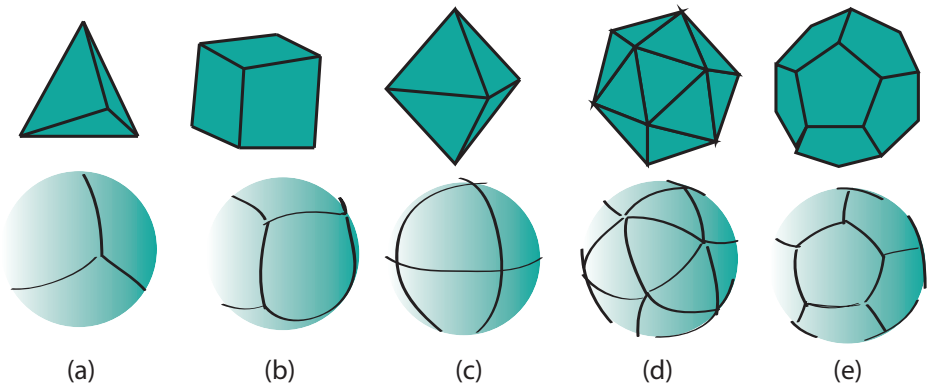


1.1. Polyhedron

Different polyhedrons have been used in DGGS. Among the proposed polyhedrons for DGGS, the icosahedron has been widely used as it can initially approximate the sphere with less angular and areal distortion [3–8] (see Figure 2 (d)). Cubes are also widely used as they provide quadrilateral cells that can be efficiently handled (see Figure 2 (b)) [9–12]. Although tetrahedrons may cause noticeable angular and areal distortions in their approximations of the Earth, they remained a popular choice due to their simplicity [13] (see Figure 2 (a)). Octahedrons are also very popular in representing the Earth since their faces can be associated with spherical octants when their singular vertices can be placed at the poles [14–16] (see Figure 2 (c)). Dodecahedrons have also been used (see Figure 2 (e)). However, since the faces of the dodecahedron are pentagons and a pentagonal refinement does not exist, it has not been widely used in DGGS although it introduces a reasonably low distortion [17].

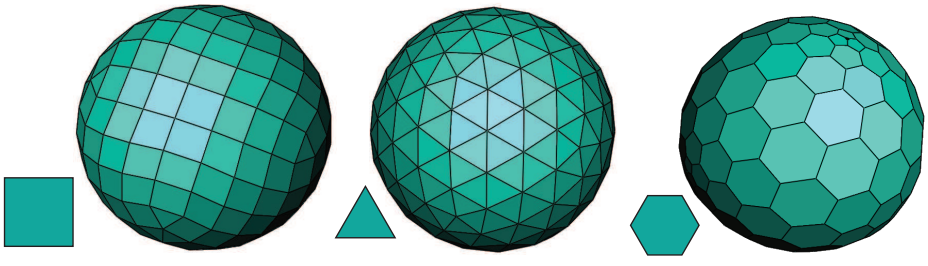
1.2. Cell Type

Figure 2. (Top) The tetrahedron, cube, octahedron, icosahedron and dodecahedron and (Bottom) each's corresponding spherical polyhedron.



Quadrilateral, triangular, and hexagonal cells are typically used in DGGS (see Figure 3). Quadrilateral cells are naturally used in DGGS using cubes as the polyhedron [9,10,12]. The congruency of quadrilateral cells paired with their adaptability to Cartesian coordinate systems, hardware devices, and available data structures (such as quadtrees) make it a popular choice. However, since the commonly used polyhedrons in DGGS initially have triangular faces, triangular cells are also widely used in DGGS [13,14,18,19]. These cells are also congruent and they can be rendered very efficiently, as they are supported by many built-in functions in rendering pipelines such as OpenGL. Conceptually, the triangular cells in polyhedrons with triangular faces, such as the octahedron and icosahedron, can also be paired into quadrilateral (diamond) cells [8]. Hexagonal cells are created on polyhedrons with triangular faces by applying a dual conversion to the triangular faces [20]. Hexagonal cells are also popular as they are efficient in sampling and they exhibit a uniform adjacency [5–8,15,16,20–22].

Figure 3. Quadrilateral, triangular, and hexagonal cells on spherical polyhedrons.

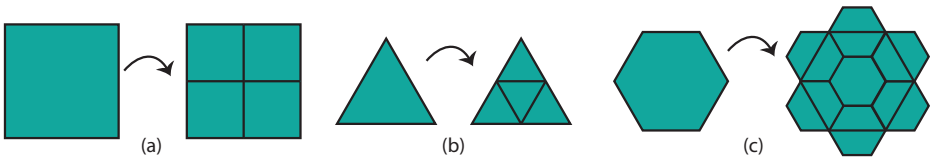


1.3. Refinement (Aperture)

Converting a set of coarse cells to finer cells through splitting edges or shrinking cells is called refinement. Consider the cells of a planar lattice. If a cell with area A is subdivided by refinement R to a set of cells with area $\frac{A}{i}$, refinement R is said to be a 1-to-i refinement with a factor (aperture) of i (see Figure 4). A refinement with a lower factor of refinement is usually desired, as it can generate a greater number of resolutions under a fixed number of maximum cells and, therefore, provides a smoother transition between resolutions [10]. Of the quadrilateral refinements, 1-to-2, 1-to-4, and 1-to-9 refinements have been used so far [9,10,12]. For triangular refinements, 1-to-4 refinement is the most

commonly used [13,14,18,19,23], while hexagonal refinements have employed in 1-to-3 [6,7], 1-to-4 [16,22] and 1-to-7 variations [24].

Figure 4. 1-to-4 refinements for quadrilateral (a), triangular (b), and hexagonal cells (c).



1.4. Projection

Spherical projections have been studied for a long time in the field of cartography [25]. When the faces of a polyhedron are projected to the sphere, two types of distortions may be created: angular distortion or areal distortion [26]. There exists a trade-off between these two distortions. This means that reducing one type of distortion causes greater distortion of the other. If a spherical projection preserves the area, it is equal area projection, and if it preserves the angles, it is angular preserving or conformal. Equal area projections are typically more desirable in DGGS as they simplify data analysis [2,6,10,11,22]. In the frameworks proposed in [2,6,22], Snyder equal area projection has been used, which is a popular projection due to its low angular distortion and the mapping of edges of the polyhedron to great circle arcs [27]. However, other types of equal area projection have also been used due to their own desirable properties, such as providing closed forms for both projection and inverse projection [10,28].

1.5. Indexing Methods

To assign data, traverse between resolutions, and handle essential queries in DGGS, a data structure is needed. Hierarchical data structures such as quadtrees may seem to be an obvious choice [29,30]. However, to interactively work with huge data on the fly, an indexing method is needed that can discard the tree structure requiring many pointers to maintain the connectivity. Many indexing methods have been proposed for DGGS. In the following sections, we first categorize the proposed indexing methods for DGGS and then, in Section 3, we provide a framework to convert an index from one DGGS to another. Note that if each of the elements listed above (i.e projection, factor of refinement, type of cell, and polyhedron) is different from a DGGS to another, the indices do not refer to the same point on the Earth. However, we can convert the indices to each other and anticipate the resulting error of the conversion process.

2. Indexing Categorization

Given an indexing method of a DGGS, it is desired that each cell at each resolution receives an index i that uniquely identifies the cell. From the index of a cell, its location (typically its centroid) on the Earth and also the resolution of the cell is determined. The index of a cell may also refer to a data structure or database to retrieve data associated with the cell. The indices of cells can be 1D (a string of letters and digits) or they can be nD, resulting from n axes defined on the faces of a polyhedron. Although various

types of indexing exist to index cells of DGGS, they are typically derived from three types of general indexing mechanisms: we call indexing methods that benefit from the hierarchy of cells provided by the refinement **Hierarchy-based** indexing methods. In some indexing methods, the parameterization provided by a **Space-filling curve** is used to index cells. Using the axes of a coordinate system defined on the cells of a polyhedron is also another common method that we name it **Axes-based** indexing. In the following, we describe each category and provide some existing examples.

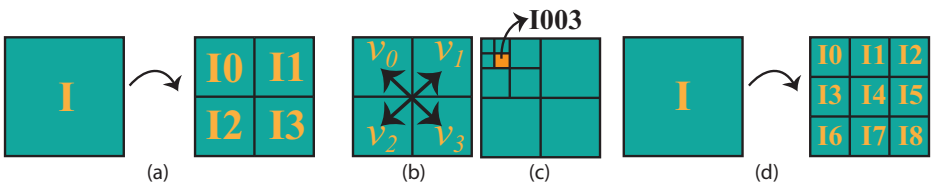
2.1. Hierarchy-based Indexing

Applying refinements on a polyhedron produces a useful hierarchy between cells that can be used to index cells. When a refinement is applied on a set of coarse cells C, a set of fine cells F is created. We can assign each cell $f \in F$ to a unique cell $c \in C$. In this case, f is the child of c.

As discussed earlier, it is possible to index the cells using the defined hierarchy. We can initially index the first resolution cells and then use this index as the prefix of the index at the following resolutions. Formally, if cell c has index I, its children f_i receive index Ii in which i is an integer digit appended to I. The range of i is denoted by b (i.e. $i \in [0, b-1]$) which is used as the base of this indexing method. This base can be used to define algebraic operations on indices, such as conversion to and from the Cartesian coordinate system, neighborhood finding, and Fourier transform [2,7,19].

An example of this indexing is proposed in [31] for quadrilateral cells resulting from a 1-to-4 refinement. As a result, the children resulting from 1-to-4 refinement of a quad with index I receive indices $I0$ for the NW cell, $I1$ for the NE cell, $I2$ for the SW cell, and $I3$ for the SE cell (Figure 5 (a)). An index with length r corresponds to a cell at resolution r whose location is determined using the vector obtained from the sequence of digits. Figure 5 (b) shows the vectors associated with each digit. The vector associated with an index is an addition of scaled vectors associated to each digit. For example, the cell with index $I003$ is obtained after applying 1-to-4 refinement three times on cell I. The vector associated with this index is obtained by $v_0 + \frac{1}{2}v_0 + \frac{1}{4}v_3$ (Figure 5 (b), (c)). SCENZ-Grid [12] also uses similar indexing method for 1-to-9 refinement of the quadrilateral faces of a cube in which the digit appended to the index of a fine cell varies between 0 and 8 based on its relative position to its parent (Figure 5 (d)).

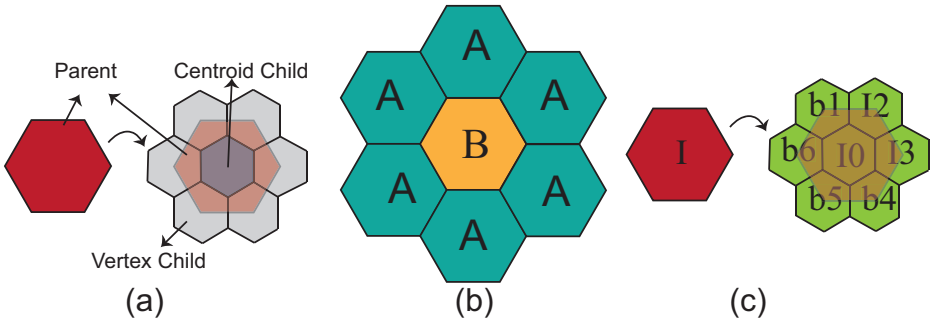
Figure 5. (a) Hierarchy-based indexing for 1-to-4 refinement. (b) Vectors corresponding to each digit. (c) 1D index I003 and its corresponding cell. (d) Hierarchy-based indexing method for quadrilateral 1-to-9 refinement.



For congruent refinements, this hierarchical relationship between the cells is trivial, as the parent encloses all its children (Figure 4 (a), (b)). However, for incongruent refinements, such as hexagonal refinements (Figure 4 (c)), assigning a set of fine cells to a unique coarse cell is not trivial . PYXIS indexing [7] is defined on hexagonal cells resulting from 1-to-3 refinement on an icosahedron. Under this

indexing, children are categorized into *centroid children* and *vertex children* (see Figure 6 (a)). Centroid children share a centroid with their parent while vertex children are covered by three different coarse hexagons, each of which could potentially be defined to be the parent. Coarse hexagons are categorized into two categories - A and B - in which each type B cell is surrounded by type A cells (see Figure 6 (b)). Each type B cell is treated as the parent of its centroid child and all its vertex children while type A cells are only considered to parent their centroid child. Afterwards, each centroid child is considered to be type B and each vertex child is considered to be type A, and the process continues and produces a fractal shape. This indexing is started from a truncated icosahedron (the refined icosahedron by the 1-to-3 refinement). In the truncated icosahedron, the initial pentagons are type B cells while hexagons are type A cells. If a coarse cell has index I, its centroid child gets indices $I0$ and its vertex children get index Ii in which $1 \leq i \leq 6$ (see Figure 6 (c)). Since each fine cell receives the index of its parent as the prefix, PYXIS indexing is a hierarchy-based indexing method.

Figure 6. (a) A coarse cell (red) is refined to finer cells. Children are categorized to vertex and centroid children. (b) Type B cells are surrounded by type A cells. (c) The children of a cell with index I get indices I0 to I6 based on their position relative to their parent.

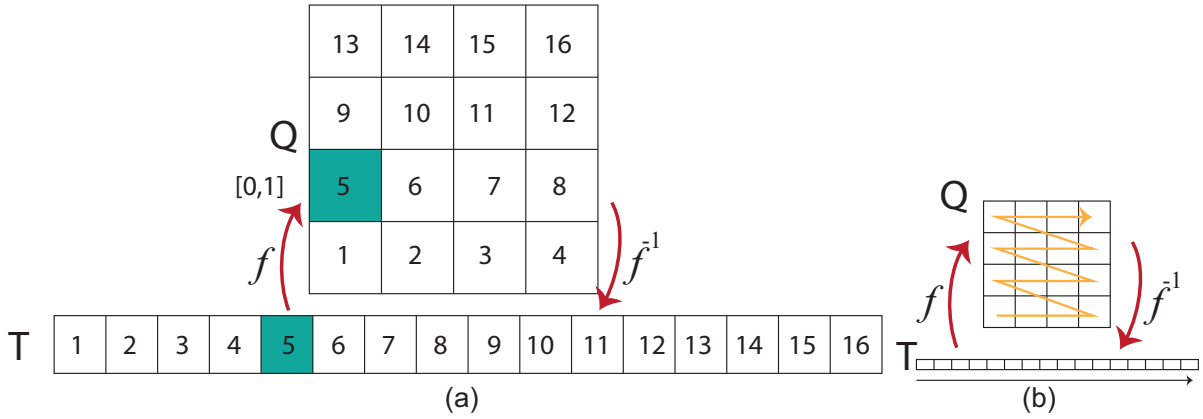


There exist many other hierarchy-based indexing methods in DGGS [11,14,19,22,23]. In [11], a hierarchy-based mechanism is used on the quadrilateral faces of a cube that are refined by factor of four. Triangular cells resulting from a 1-to-4 refinement are indexed similarly in [14,19,23] and, in [22], the hexagonal cells of an icosahedron resulting from a 1-to-4 refinement are indexed with a similar mechanism.

2.2. Space-filling Curves Indexing

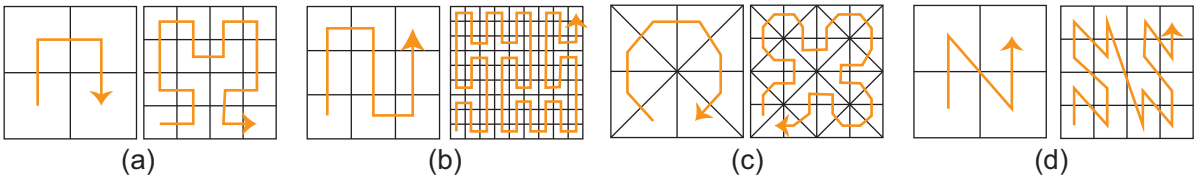
Another method of indexing cells in DGGS is to use space-filling curves (SFCs) as a reference. SFCs have been used in many applications, such as compression, rendering, and database management. SFCs are 1D curves that are created recursively to eventually cover a space. Space filling curve $f(t)$ typically provides a mapping from $T \subset R$ to $Q \subset R^2$ in which $t \in T$ (see Figure 7). Using a SFC that visits all the cells of Q after refinement, we may define an indexing on the cells of Q by discretizing T based on the number of cells. The 1D index I for cells in Q is defined on domain T (i.e. by taking a unit step on T, I is incremented). Each index I has a corresponding cell on Q that is returned by the mapping f ($f(I) \in Q$). Given mapping f from T to Q, f^{-1}, which maps the cells of Q to their unique index defined on T, can also be defined. Figure 7 illustrates an example of an indexing for Q based on a SFC, which is simply a row-major traversal.

Figure 7. (a) Function f maps parameters on T to domain Q. For instance, $t = 5$ is mapped to $[0, 1]$. Parameters on T are used as indices of Q. (b) A space filling curve that performs row major traversal is illustrated in orange.



Based on the properties that function f exhibits, different SFCs have been proposed. Some of the common SFCs are Hilbert, Peano, Sierpinski, and Morton (Z), as illustrated in Figure 8. As noticeable in Figure 8, these curves typically have a simple initial geometry defined on a simple domain. The domain is then refined and the simple geometry is repetitively transformed to cover the entire refined domain. For instance, in the case of the Hilbert curve, the simple geometry is defined on a simple two by two domain and the domain is then refined by a 1-to-4 refinement. Typically, if the initial geometry covers i cells, a 1-to-i refinement is suitable to get a refined domain. This way, each SFC is associated with a refinement.

Figure 8. (a) Hilbert, (b) Peano, (c) Sierpinski, and (d) Morton space filling curves.

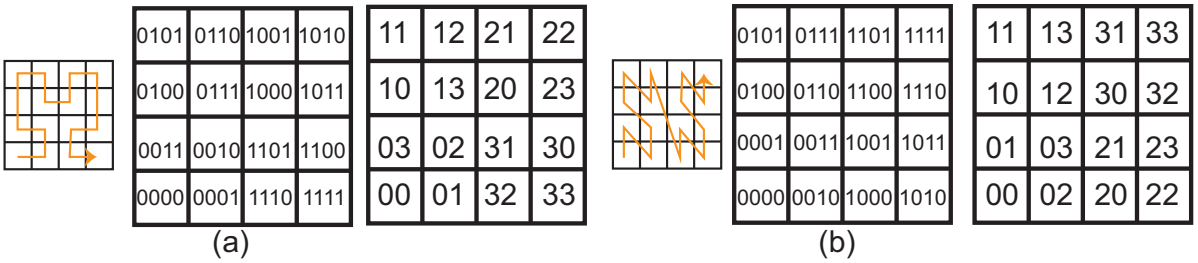


To index cells based on a SFC, decimal numbers may not be the best choice as their corresponding indices do not directly provide any information about the resolution. As a result, a base b for the indexing may be chosen to solve this issue. The base of the indexing method is usually chosen as i or $\sqrt{i}$ if refinement 1-to-i is associated with the SFC curve. In this case, given an index with base i, a cell at resolution n has an index whereas, given a base of $\sqrt{i}$, the cell receives an index of length $2n$. In addition, using these bases, redundant bits are not necessary to index cells, since whole cells resulting from these refinements can be covered by indices from $00...0$ to $(b-1)...(b-1)$.

For instance, the refinement associated with the Hilbert and Morton curves is 1-to-4. Therefore, base four or two for Hilbert and Morton is appropriate. After fixing a base for the index, the cell associated with the initial point of the curve gets index 0. As we move along the SFC, the index of each subsequent cell is incremented by 1 in base b. Figure 9 illustrates such an indexing for the Hilbert and Morton curves using bases two and four. Although base four seems to be more efficient, as it provides a shorter string to index cells, indices with base two can also be implemented efficiently as they are compatible with efficient binary operations in hardware.

Indexing methods derived from SFCs have been widely used in DGGS and terrain rendering. For instance, in [8,32], Morton indexing has been used to index cells resulting from 1-to-4 refinements on the icosahedron and octahedron while, in [18], the Sierpinski SFC has been used to index triangular cells. SFCs used in terrain rendering provide a 1D ordering of triangle strips and vertices suitable for GPU and out-of-core algorithms [33–36].

Figure 9. (a) (Left) Hilbert SFC. (Middle) Indexing with base 2. (Right) Indexing with base 4. (b) (Left) Morton SFC. (Middle) Indexing with base 2. (Right) Indexing with base 4.

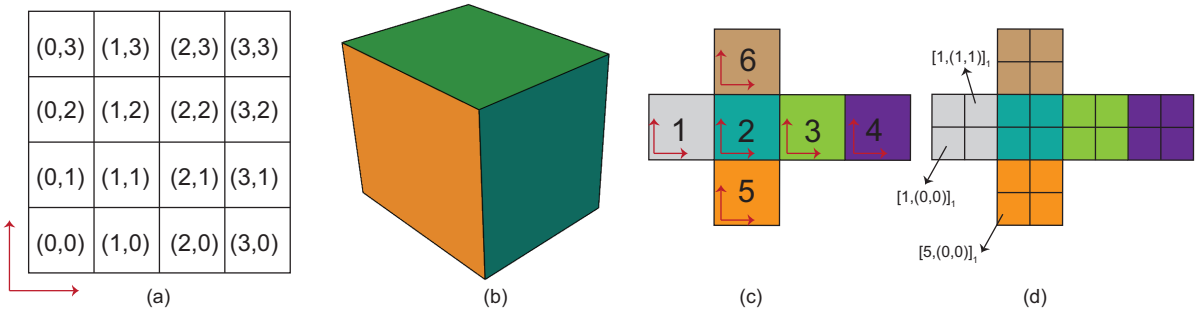


2.3. Axes-based Indexing

A natural way for indexing is to define a set of m axes U_1 to U_m to span the entire space on which the cells lie. Then the index would be an m dimensional vector $(i_1, i_2, ..., i_m)$ in which i_j are integer numbers indicating unit steps taken along the axes U_j. A simple example of such an indexing is to index a quadrilateral domain by Cartesian coordinates, as illustrated in Figure 10 (a). When a refinement is applied to the cells, a subscript r is appended to the index indicating the resolution [20,37,38]. In the available indexing methods for DGGS, m is typically two or three. For instance, 2D indexing has been used in [5,10,20] while 3D indexing has been used in [15,16] by taking the Barycentric coordinate of each cell to be its index. To apply a 2D indexing method on a polyhedron for DGGS, the polyhedron can be unfolded to a 2D domain and the axes defined for the entire 2D domain, or each face can be given its own coordinate system [20,37,38]. Figure 10 illustrates an indexing for the quadrilateral cells of a cube after 1-to-4 refinement wherein each face has its own coordinate system. To distinguish between the cells associated with each face, an additional component referring to the initial number of the polyhedron's faces can be added to the indices. As a result, index $[f, (a, b)_r]$ refers to cell (a, b) in face f at resolution r (Figure 10 (d)).

The provided categorization mostly reflects the core idea for constructing indexing methods. According to this categorization/construction, some of the operations can be naturally handled. For example, hierarchy-based indexing methods naturally lead to efficient parent-child operations. However, it is necessary to consider other properties and operations for well-designed indexing methods. Certainly, it is possible to handle neighborhood finding in hierarchy-based indexing methods but probably not as efficient/natural as axes-based techniques. In addition, based on the pattern of indices, some indexing methods can belong to two categories (e.g SFC or hierarchy-based). However, an indexing method is either constructed by a parametrized SFC or by inheriting the index of its parent. Although it is possible to use a parametrize SFC that indexes the children by indices that have the prefix of their parents, the indexing method is SFC since the construction of indexing is based on the parameterization of SFC.

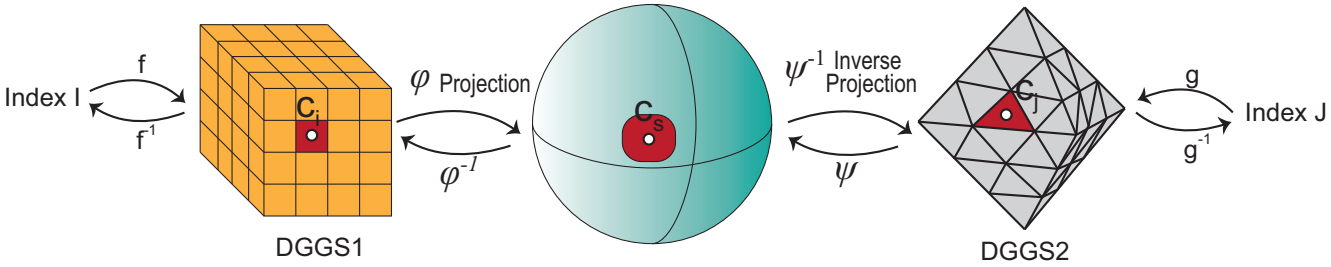
Figure 10. (a) Integer indexing of cells using Cartesian coordinates. (b) A cube. (c) The unfolded cube in (b) and coordinate systems for each face. (d) Indices of some cells after one step of 1-to-4 refinement.



3. General Conversion Method

As discussed earlier, there exist many different Digital Earth frameworks resulting from various types of DGGS. However, data in these frameworks are essentially assigned to the index of each cell. Indexing methods for different DGGS may be different from each other. In order to exchange, unify, or standardize data represented in different frameworks, a conversion is necessary that can convert an index in one type of DGGS to another. Consider index I in DGGS1, we would like to know the index J in DGGS2 that is referring to "almost the same area". Since each cell is composed of a point and an area that has been enclosed by that cell on the sphere (or ellipsoid), if any component of the DGGS (type of cell, polyhedron, indexing, aperture, and projection) differs between DGGS1 and DGGS2, the index of a cell in DGGS1 refers to a different area on the sphere (ellipsoid) compared to an index in DGGS2. To define such a conversion between different frameworks, we can benefit from a common domain for two DGGS. This domain can be a spherical/ellipsoidal or 2D Cartesian corresponding to faces of the polyhedron (see Figure 11). As a result, we discuss two cases for conversions in the following.

Figure 11. Index I in DGGS1 is mapped to the refined polyhedron by mapping f. The cell is then projected to the sphere by the projection φ used in DGGS1. Then the centroid of cell I is inverse projected by the inverse projection ψ^{-1} of DGGS2. The index J in DGGS2 is then found using inverse mapping g^{-1}.



Case 1 (General Case): Consider that, in addition to different indexing methods, at least another component of DGGS1 is different from DGGS2. In this case, we use a common spherical domain to convert index I in DGGS1 to index J in DGGS2. Consider I in DGGS1 and c_i the centroid of its corresponding cell in DGGS1. The face corresponding to this index on the refined polyhedron of DGGS1 is found by the mapping f. Note that functions f and f^{-1} are readily available in most DGGS since it is

an essential query to relate an index to its position on the polyhedron and vice versa. We can then project this point to point c_s on the sphere, using the projection φ used in DGGS1. c_s can be inversely projected to point c_j on DGGS2 using the inverse projection ψ^{-1} in DGGS2. The index J of the cell enclosing c_j in DGGS2 is the desired index, that can be obtained by the mapping g^{-1} (see Figure 11). Note that g maps an index to a point on a polyhedron while g^{-1} maps a point to an index of a cell enclosing the point at a specific resolution.

There exists an ambiguity here about the resolution of indices. The resolution of index I is given. However, the resolution of index J should be determined. To do this, we take the resolution for J at which the area of cells corresponding to I and J are as close as possible. For instance, if a cell has an index at resolution seven in PYXIS indexing, the corresponding index on the cube after 1-to-9 refinement (as in SCENZ-Grid) will be at resolution five, while the index on the cube after 1-to-4 refinement will be at resolution seven. Table 1 lists the number of cells in these frameworks, which is used to evaluate the areas of the cells at each resolution. Consequently, if cell I in DGGS1 has area A_i, resolution r in DGGS2 is chosen in which $Min_r\,|A_r - A_i|$ is satisfied (i.e. the resolution is chosen in which the areas of the cells (A_r) is closest to A_i).

Table 1. Number of cells at each resolution in three different frameworks. Note that under the PYXIS framework, the first resolution starts from 12 pentagons and 20 hexagons created on the icosahedron.

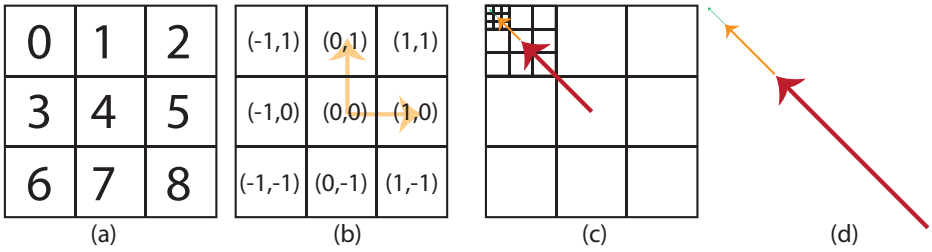
Resolution	PYXIS	SCENZ-Grid	Cube(1-to-4)
1	32	6	6
2	92	54	24
3	272	486	96
4	812	5374	384
5	2432	39366	1536
6	7292	354294	6144
7	21872	3188646	24576
8	65612	28697814	98304
9	196832	258280326	393216

Case 2: The conversion is easier when all components of DGGS1 and DGGS2 are the same except for the indexing methods, hence $\psi = \varphi$ and resolutions are the same. As a result, we can easily use functions f and g that map an index to a polyhedron. In this case, the indices are either in the first two categories (SFC or hierarchy-based) and as a result are 1D indices, or they are integer vectors defined in an axes-based indexing method. In both cases, we can convert index of one category to a vector and convert the vector to an index in another category.

Functions f and g are usually found by determining the corresponding 2D vectors of each index. To describe this conversion, we use an example. Consider the cells of 1-to-9 refinement on the cube with the hierarchy-based indexing used in SCENZ-Grid (see Figure 5), and an axes-based indexing using Cartesian coordinates (Figures 12 (a), (b)). A cell with index N000 is given and its corresponding index in the axes-based indexing system is desired. As illustrated in Figure 12, digit 0 in the hierarchy-based

indexing system corresponds to vector $(-1, 1)$ in the Cartesian coordinate system. Note that as cells get smaller by a factor $\frac{1}{3}$ as the resolution increases, the length of this vector is also scaled by $\frac{1}{3}$ (Figures 12 (c), (d)). As a result, 000 in the hierarchy-based indexing corresponds to the addition of vectors $(-1, 1)$, $\frac{1}{3}(-1, 1)$, and $\frac{1}{9}(-1, 1)$ which is equivalent to $(\frac{-13}{9}, \frac{13}{9})$. To get integer indices, we only need to scale the coordinate by 9, therefore index corresponding to N000 is $[N, (-13, 13)_2]$ (N refers to one of the initial faces of the cube).

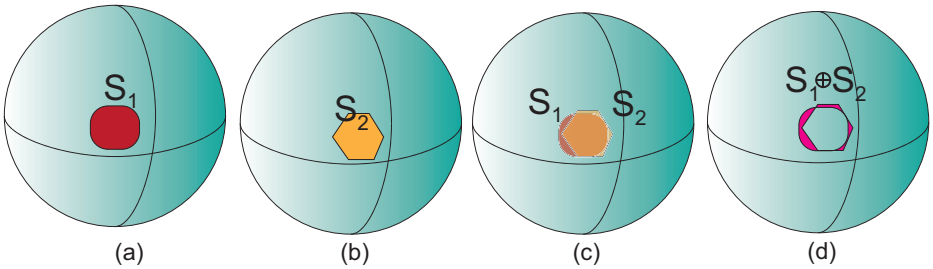
Figure 12. (a) Indices in the hierarchy-based indexing method used in SCENZ-grid and (b) The corresponding integer indices for each digit of (a) based on the defined coordinate system. (c) Vectors corresponding to index I000. (d) Magnified vectors from (c) for a better illustration.



The complexity of both of the conversion methods proposed in this paper is dependent on the employed functions: f, g, ϕ, and ψ and their inverses. These functions are typically computed in constant time or very efficiently as the performance of the DGGS is dependent on these functions. As a result, the performance of our proposed conversion is proportional to the performance of f, g, ϕ and ψ.

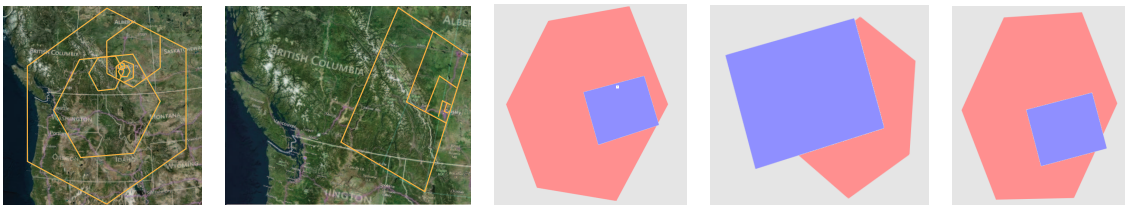
When the cells in these DGGS differ, any conversion introduces an error. Therefore, while no error while be introduced in the second case, there will be some error in the first case as the cells represented by two indices may be different in shape and size. Since each index represents an area on the surface of the Earth, this error can be measured as the difference between the areas that the two cells are covering (see Figure 13). Indeed, if S_1 and S_2 are, respectively, sets of points on the surfaces of DGGS1 and DGGS2, the error can be measured by the symmetric difference of these two subsets ($S_1 \oplus S_2$).

Figure 13. (a) S_1 is the subset of points on the sphere enclosed by DGGS1. (b) S_2 is the subset of points on the sphere enclosed by DGGS2. (c) S_1 and S_2 are overlapped. (d) The error is $S_1 \oplus S_2$ which is the difference of the areas that are enclosed by S_1 and S_2.



To provide an example of our proposed conversion, we have implemented it for converting from PYXIS to Cube (1-to-4). In Cube (1-to-4) framework, the equal area projection discussed in [28] has been used with a 1-to-4 refinement. As discussed earlier, we can determine resolutions that PYXIS cells

Figure 14. (a) Hexagonal cells of PYXIS framework at several successive resolutions. (b) Corresponding cells of (a) in Cube (1-to-4). (c,d,e) PYXIS cells at resolution three, six, and seven and its corresponding cells in Cube (1-to-4) at resolution four, six, and seven. Cells are uniformly scaled to provide a better visualizations.

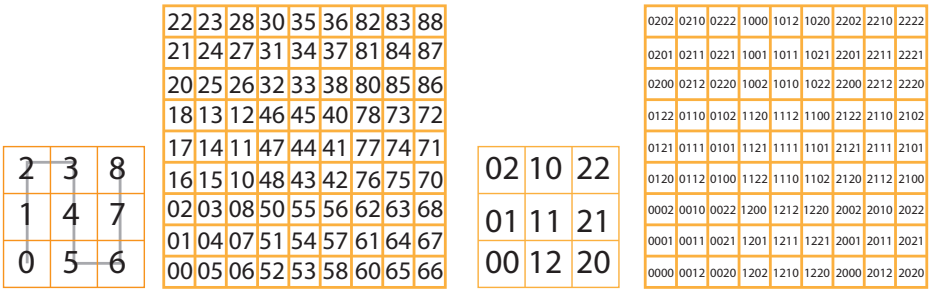


(a) (b) (c) (d) (e)

and Cube (1-to-4) cells are more compatible based on the area of cells. For instance, PYXIS cells at resolutions three, six, and seven, correspond to cells of Cube (1-to-4) at resolutions four, six, and seven respectively. Hexagonal cells in PYXIS framework are converted to quadrilateral cells in Cube (1-to-4) illustrated in Figure 14 (textures are removed to increase the visibility). After converting hexagonal cells to quadrilateral cells, it shows that hexagonal cells and quad cells share an area with factors of 15%, 48%, and 18% at resolutions three, six, and seven respectively. This conversion is performed real-time (less than mili-seconds) at any resolution as projections and inverse projections of both of the frameworks are very fast and efficient.

Using the conversions proposed in this section, we can simply convert indices from one type of DGGS to another. This work also clarifies that various types of indexing method can be proposed for a DGGS. However, most methods fall under one of the categories proposed in this paper. For instance, it is possible to define another indexing method for SCENZ-Grid using the Peano SFC considering two different bases, 3 and 9 (see Figure 15). As a result, our work can serve as useful material for researchers looking to design an indexing method for a potential OGC standard working based on DGGS.

Figure 15. (Left) Indexing of quadrilateral 1-to-9 refinement at two successive resolutions using the Peano SFC with base 9. (Right) The same indexing as Left but with base 3.



4. Conclusions

In this paper, we provide a categorization of indexing methods proposed for Digital Earth frameworks. We then proposed a conversion from one category to another. This shows that data allocated to cells can be represented in different ways using different types of DGGS, and that there exists a simple conversion

between these representations that can be used to unify the data available given any DGGS. This work also clarifies a method to index potentially new DGGS with a specific type of cell and aperture.

Using our proposed conversion, current DGGS that are employed in research or industrial settings can communicate, share, and receive data from each other or a potential OGC standard. As a result, our work facilitates the interoperability of DGGS. In addition, using our proposed categorization, it is possible to propose alternative indexing mechanisms for an established DGGS.

There remain some possible directions for future work regarding this subject. In this paper, we focus mostly on DGGS that employ an equal area projection. However, other types of projections may be used. In that case, the area of the cells are not equal even within a specific resolution. Hence, when converting the index of a cell, the conversion will need to locate the cell that has the closest area to the target cell. More research is needed to determine an efficient method for choosing the appropriate resolution at which to locate this cell.

Author Contributions

- Providing a categorization for indexing methods proposed for DGGS.
- A general conversion method from a specific type of index to another.
- A common ground for providing alternative indexing methods for a DGGS.

Conflicts of Interest

"The authors declare no conflict of interest".

References

1. Goodchild, M.F. Discrete Global Grids for Digital Earth. Proceedings of 1 st International Conference on Discrete Global Grids, March, 2000.
2. Sahr, K.; White, D.; Kimerling, A.J. Geodesic Discrete Global Grid Systems. *Cartography and Geographic Information Science* **2003**, *30*, 121–134.
3. Tong, X.; Ben, J.; Qing, Z.; Zhang, Y. The hexagonal discrete global grid system appropriate for remote sensing spatial data. Vol. 7146, pp. 71460J–71460J–10.
4. Tong, X.; Ben, J.; Wang, Y.; Zhang, Y.; Pei, T. Efficient Encoding and Spatial Operation Scheme for Aperture 4 Hexagonal Discrete Global Grid System. *Int. J. Geogr. Inf. Sci.* **2013**, *27*, 898–921.
5. Sahr, K. Location Coding on Icosahedral Aperture 3 Hexagon Discrete Global Grids. *Computers, Environment and Urban Systems* **2008**, *32*, 174–187.
6. Peterson, P. Close-packed, Uniformly Adjacent, Multiresolutional, Overlapping Spatial Data Ordering. United States Patent Application 20060265197, 2006.
7. Vince, A.; Zheng, X. Arithmetic and Fourier transform for the PYXIS multi-resolution digital Earth model. *Int. J. Digital Earth* **2009**, *2*, 59–79.

8. White, D. Global Grids from Recursive Diamond Subdivisions of the Surface of an Octahedron or Icosahedron. *Environmental Monitoring and Assessment* **2000**, *64*, 93–103.
9. Alborzi, H.; Samet, H. Augmenting SAND with a Spherical Data Model. First International Conference on Discrete Global Grids, 2000.
10. Mahdavi-Amiri, A.; Bhojani, F.; Samavati, F. One-to-Two Digital Earth. ISVC (2), 2013, pp. 681–692.
11. GÃşrski, K.M.; Hivon, E.; Banday, A.J.; Wandelt, B.D.; Hansen, F.K.; Reinecke, M.; Bartelmann, M. HEALPix: A Framework for High-Resolution Discretization and Fast Analysis of Data Distributed on the Sphere. *The Astrophysical Journal* **2005**, *622*, 759.
12. Gibb, R.; Raichev, A.; Speth, M. The rHEALPIX Discrete Global Grid System, 2012.
13. Cozzi, P.; Ring, K. *3D Engine Design for Virtual Globes*, 1st ed.; CRC Press, 2011.
14. Dutton, G. *A Hierarchical Coordinate System for Geoprocessing and Cartography: With 85 Figures, 16 Tables and 2 Foldouts*; Lecture Notes in Earth Sciences Series, Springer Verlag, 1999.
15. Vince, A. Indexing the Aperture 3 Hexagonal Discrete Global Grid. *J. Vis. Comun. Image Represent.* **2006**, *17*, 1227–1236.
16. Ben, J.; Tong, X.; Chen, R. A Spatial Indexing Method for the Hexagon Discrete Global Grid System. Geoinformatics, 2010 18th International Conference on, 2010, pp. 1–5.
17. Wickman, F.E.; Elvers, E.; Edvarson, K. A System of Domains for Global Sampling Problems. *Geografiska Annaler. Series A, Physical Geography* **1974**, *56*, 201–212.
18. Bartholdi, J.J.; III.; Goldsman, P. Continuous Indexing of Hierarchical Subdivisions of the Globe. *International Journal of Geographical Information Science* **2000**, *15*, 489–522.
19. Goodchild, M.F.; Shiren, Y. A hierarchical spatial data structure for global geographic information systems. *CVGIP: Graph. Models Image Process.* **1992**, *54*, 31–44.
20. Mahdavi-Amiri, A.; Harrison, E.; Samavati, F. Hexagonal Connectivity Maps for Digital Earth. *International Journal of Digital Earth* **2014**. To Appear.
21. Sahr, K. Hexagonal Discrete Global Grid Systems for Geospatial Computing. *Archives of Photogrammetry, Cartography and Remote Sensing* **2011**, *22*, 363–376.
22. Tong, X.; Ben, J.; Wang, Y.; Zhang, Y.; Pei, T. Efficient encoding and spatial operation scheme for aperture 4 hexagonal discrete global grid system. *International Journal of Geographical Information Science* **2013**, *27*, 898–921.
23. Szalay, A.S.; Gray, J.; Fekete, G.; Kunszt, P.Z.; Kukol, P.; Thakar, A. Indexing the Sphere with the Hierarchical Triangular Mesh. *CoRR* **2007**, *abs/cs/0701164*.
24. Gibson, L.; Lucas, D. Spatial Data Processing Using Generalized Balanced Ternary. Proceedings of the IEEE Computer Society on Pattern Recognition and Image Processing, 1982, pp. 566–572.
25. Kennedy, M.; Koop, S.; Environmental Systems Research Institute (Redlands, C. *Understanding Map Projections: Gis by Esri*; ArcGIS 8 concept guides, Environmental Systems Research Institute, 2000.
26. Hormann, K.; Lévy, B.; Sheffer, A. Mesh Parameterization: Theory and Practice Video Files Associated with This Course Are Available from the Citation Page. ACM SIGGRAPH 2007 Courses, 2007, SIGGRAPH '07.

27. Snyder, J.P. An Equal Area Map Projection for Polyhedral Globes. *Cartographica* **1992**, *29*, 10–21.
28. Roşca, D.; Plonka, G. Uniform spherical grids via equal area projection from the cube to the sphere. *Journal of Computational and Applied Mathematics* **2011**, *236*, 1033 – 1041.
29. Samet, H. *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.
30. Faust, N.; Ribarsky, W.; Jiang, T.; Wasilewski, T. Real-Time Global Data Model for the Digital Earth. Proceedings of the international conference on discrete global grid, 2000.
31. Gargantini, I. An Effective Way to Represent Quadtrees. *Commun. ACM* **1982**, *25*, 905–910.
32. Jianjun Bai, Xuesheng Zhao, J.C. INDEXING OF THE DISCRETE GLOBAL GRID USING LINEAR QUADTREE. *ISPRS Workshop on Service and Application of Spatial Data Infrastructure* **2005**, *7146*, 267–270.
33. Pharr, M.; Fernando, R. *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation*; Addison-Wesley Professional, 2005; chapter Terrain Rendering Using GPU-Based Geometry Clipmaps.
34. Hwa, L.M.; Duchaineau, M.A.; Joy, K.I. Real-Time Optimal Adaptation for Planetary Geometry and Texture: 4-8 Tile Hierarchies. *IEEE Trans. Vis. Comput. Graph.* **2005**, *11*, 355–368.
35. Lawder, J. The Application of Space-filling Curves to the Storage and Retrieval of Multi-dimensional Data. Phd thesis, University of London, 2000.
36. Lindstrom, P.; Pascucci, V. Terrain Simplification Simplified: A General Framework for View-Dependent Out-of-Core Visualization. *IEEE Transactions on Visualization and Computer Graphics* **2002**, *8*, 239–254.
37. Mahdavi-Amiri, A.; Samavati, F. Connectivity Maps for Subdivision Surfaces. GRAPP/IVAPP, 2012, pp. 26–37.
38. Mahdavi-Amiri, A.; Samavati, F. Atlas of connectivity maps. *Computers & Graphics* **2014**, *39*, 1 – 11.