Hierarchical Grid Conversion

Ali Mahdavi-Amiri, Erika Harrison, Faramarz Samavati

Department of Computer Science, University of Calgary, 2500 University Drive N.W.,
Calgary, Alberta, Canada, T2N 1N4, Phone: +1-403-890-7560

Abstract

Hierarchical grids appear in various applications in computer graphics such as subdivision and multiresolution surfaces, and terrain models. Since the different grid types perform better at different tasks, using simple conversions, we can switch between the grid types to take advantages of each grid for better supporting advanced applications. In this paper, we introduce some simple conversions between grids. To describe their usage, we define new regular and semiregular refinements. We also describe how patch-based data structures can be used for hexagonal cells and semiregular refinements.

Keywords: Refinements, Grid Conversion, Patch-based Data Structures, Transformations, Semiregular, Subdivision

1. Introduction

Triangular, quadrilateral, and hexagonal grids appear in many applications in computer graphics such as finite elements, subdivision and multiresolution surfaces, and terrain rendering. Triangular grids are common due to their application in many fundamental algorithms such as Delaunay triangulation and Loop subdivision [1, 2], and they are also optimized for processing on modern hardware. The simple parametric form of quadrilateral grids can be readily applied to tensor product surfaces, NURBS, B-Spline, and Catmull-Clark patches [3, 4]. Furthermore, quadtrees [5] exploit the simple boundaries of quadrilateral grids and their straightforward hierarchical shape. Hexagonal grids provide the best sampling of surfaces, support

Email address: `amahdavi@ucalgary.ca` (Ali Mahdavi-Amiri)
URL: `http://pages.cpsc.ucalgary.ca/~amahdavi/` (Ali Mahdavi-Amiri)

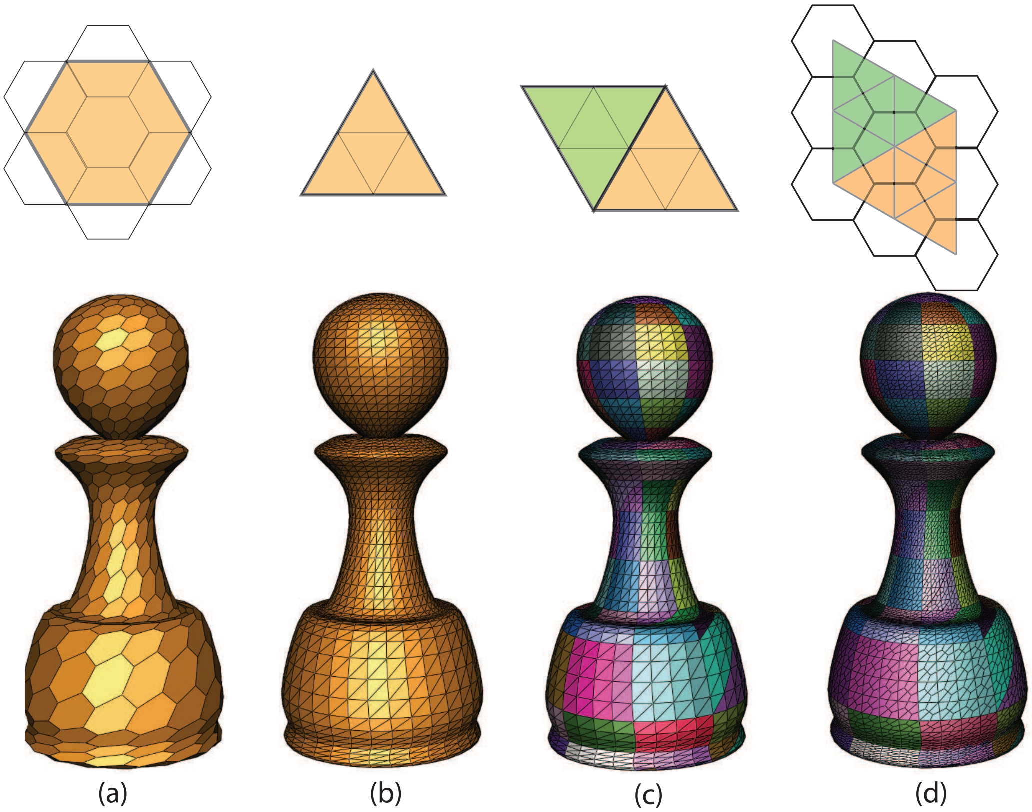


Figure 1: Hexagonal and triangular refinements are used to make hierarchical models (a), (b). There exists a conversion between regular grids that can be used to design an efficient hierarchical data structure for all types of regular grids (c), (d).

uniform neighborhood, and provide a reduced quantization error over other alternatives [6]. As a result, hexagonal grids appear in applications such as hierarchical representation of the Earth and subdivision surfaces [7, 8].

In this paper, we provide hierarchical grid conversions between triangular, quadrilateral and hexagonal grids. These conversions are basically simple modifications in the connectivity of vertices that convert a type of grid to another. Using these conversions, we can switch between the grids as the need dictates (see Figure 1). For example, hierarchical shapes resulting from a refinement of quads are very simple as opposed to hexagons that are not congruent (this means that it is not possible to completely cover a hexagon by a set of complete and disjoint smaller hexagons). As a result, we can convert hexagonal grids to quadrilaterals to design an efficient data structure for hexagonal grids and benefit from the simple hierarchical shape of quads and convert them back when cells with better sampling rate or a uniform neighborhood definition is desired.

Hierarchy among the cells is typically provided by refinements. Refinements introduce more cells and vertices into a model. When a refinement is applied to a cell with area A, it divides the cell into some smaller cells with area $\frac{A}{i}$. Such a refinement is called 1-to-i refinement or a refinement with the factor of i [9]. Refinements are useful when i is an integer number since after two levels of subdivision the cells are simply scaled by an integer number (although lattices may not be aligned). However, these refinements are typically specified for a particular grid. For instance, quadrilateral 1-to-3 refinement has not been defined while triangular 1-to-3 refinement has been successfully employed in $\sqrt{3}$ subdivision. Using hierarchical grid conversions, we propose a framework to define such refinements and study their properties.

1.1. Contributions

Our main contribution is to present hierarchical conversions between regular grids. To demonstrate the usefulness of these conversions, we use them to define new semiregular and regular refinements for grids and extend an existing patch-based hierarchical data structure - Atlas of Connectivity Maps (ACM)- [10, 11] to support hexagonal grids and more variety of regular and semiregular refinements.

2. Related work

As we present hierarchical grid conversions in this paper and use them to define new refinements and hierarchical data structures, we can categorize the work related to our method into three groups: conversions between regular grids, refinement and subdivision, and data structures proposed to support multiresolution (hierarchy) of semiregular models. In the following, we provide prior work of each group.

2.1. Conversion between Regular Grids

Grid conversion is already well explored within the Computer Graphics community, under the topic of remeshing. Triangulation [12, 13] and quadrangulation [14] convert arbitrary meshes to those with cells, of triangles and quadrilaterals respectively. This remeshing may improve rendering time, mesh quality, or fulfill geometric or aesthetic constraints. Hexagonal remeshing occurs for architectural or parametric reasons [15, 16, 17, 18]. Alternatively, conversions can occur through duality remeshing to achieve a specific cell type [8], or improve smoothness [19].

These cell conversions mostly take complicated geometric properties of the object into consideration for converting one type of grid to another as their applications need to satisfy a specific geometric property. However, we convert the grids on 2D domains by simple operations that only change the connectivity of vertices. Some of these conversions are very straightforward. However, we combine them with refinements to define new refinements and design efficient hierarchical data structures.

2.2. Refinements

Regular refinements in surface modeling are the process of splitting faces into a set of smaller faces. After refinements, more faces and vertices are created and a higher resolution model is obtained. As a result, refinements can create a hierarchy of objects at different resolutions (i.e. the level of refinement). Regular refinements have many usages in computer graphics such as subdivision in which faces are initially split by a regular refinement and then vertices are geometrically modified to obtain a smooth surface.

Regular refinements are defined differently in literature. Guskov et al. [20] consider only the dyadic refinement as a regular quadrilateral refinement in which a face is split into four faces (Figure 2 (b)). Weiss and De Floriani

[21] also consider the same definition for triangular faces. This type of refinement is the most common refinement as it is employed in designing popular subdivision methods such as Catmull-Clark and Loop [4, 2] and useful data structures such as quadtrees [5].

Velho in [22] defines regular refinements for quadrilateral meshes as a process that produces a finer set of similar faces that are only scaled. He then categorizes regular refinements as *primal* and *dual*. In a primal subdivision, the vertices of the coarse tessellation is preserved and old edges are divided and reconnected while in a dual subdivision, new vertices are inserted in the interior of faces and the old vertices and edges are discarded. With this definition Doo-Sabin refinement is a regular dual refinement. However, there still exists some regular refinements that are not included in this definition. Simplest refinement [23] in which new vertices are inserted at edges and the old vertices and edges are discarded is an example of a regular refinement that does not cover by Velho's definition (Figure 2 (d)).

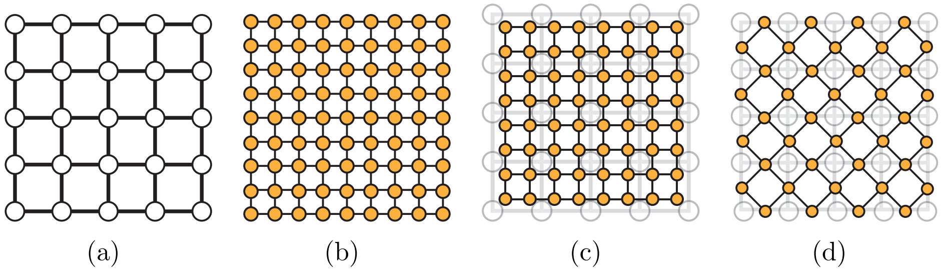

(a) (b) (c) (d)

Figure 2: (a) A set of coarse points. (b) Dyadic refinement. (c) Doo-Sabin refinement. (d) Simplest refinement.

Alexa [24] defines the definition of regular refinements for triangular meshes. He establishes the definition of regular refinements using 2D regular triangular meshes. He considers a coordinate system for triangle meshes using two edges of a equilateral triangle. To study the behavior of refinements, he establishes the hierarchical relationship between the coordinate system of triangular meshes before and after refinements.

As noted by Ivrissimtzis et al. [25], the definition of Alexa for 2D triangular meshes can be rephrased as 2D triangular lattices. If a 2D regular triangular lattices is named L_0, after an application of a regular refinement, a higher resolution lattice L_1 is created, and after r applications of a regular refinement L_r is made. Regular refinements studied by Alexa have three conditions as the following:

- All the lattices are similar (all triangles are equilateral).
- Lattice L_{n+1} can be obtained using a transformation in the Euclidean plane and the scale of the transformation is called *arity* of the refinement.
- Point sets of a lattice with higher resolution is the superset of the point set of a lower resolution (i.e. $L_0 \subset L_1 \subset L_2 ... \subset L_n$).

Ivrissimtzis et al. [25] use the same concept for categorizing refinements for regular quadrilateral lattices. However, the third condition in the Alexa's definition excludes Dual refinements such as Doo-Sabin since in such a refinement the old vertices are discarded ($L_r \not\subset L_{r+1}$). As a result, they replace the third condition with a looser one in which they consider both center-faces, and the vertices of L_{r+1}, as valid locations for the points of the coarser lattice L_r.

The usefulness of these refinements is later examined through a set of heuristics by Dodgson in [26]. Destelle et al. [27] also defined a set of subdivision operators with which regular refinements as well as other irregular refinements can be produced. However, there exists an intermediate set of refinements that are not completely irregular but they are not also regular. An example of such a refinement is the 1-to-2 refinement employed in 4-8 subdivision which carries good properties such as C^4 continuity at regular vertices. In this paper, we introduce new regular refinements and semiregular refinements by proposing a simple framework and study some of their properties. We hope that these refinements can later provide useful smooth subdivision schemes although subdivision schemes are not the only application of refinements. An alternative example of the refinements' application is Earth representation in which a hierarchical model is created by combining refinement and spherical projection [28, 7, 29].

2.3. Multiresolution Data Structures

Multiresolution surfaces have applications in mesh editing, compression, and morphing. One approach to construct multiresolution is to use surface subdivision (or refinement) [30, 31]. Various data structures support subdivision and multiresolution surfaces [9], with the most common keeping the connectivity information of vertices and faces into each edge [32, 33]. Unfortunately, these data structures require an excessive amount of memory and time to represent and maintain high resolution objects and their hierarchy.

To support subdivision and multiresolution surfaces, hierarchical data structures such as quadtrees are more useful [5]. In order to make quadtrees more efficient, indexing methods in which a unique index is assigned to each node have been proposed. Connectivity queries are then handled using a defined algebra on the indices themselves [5, 34]. Hierarchical indexing methods can be efficient in terms of both space and time.

In some multiresolution frameworks, meshes have to be semiregular or they must have subdivision connectivity [14, 31, 35]. These models are obtained by applying repetitive refinement on a mesh with an arbitrary topology. Recently, Mahdavi-Amiri and Samavati proposed an Atlas of Connectivity Maps (ACM) an indexing method to support semiregular quadrilateral and triangular multiresolution objects [36, 11]. Using conversions between regular grids, we can adapt ACM to additionally support hexagonal models. Conversions between grids not only enables us to adapt ACM but we can also adapt other indexing methods defined for quadtrees or similar structures featuring different properties to support all regular grids.

3. Conversion between Regular Grid Systems

By changing the connectivity of cells or applying simple operations, we can convert one type of a regular grid to another. These conversions can be then used to define novel refinements or extend hierarchical data structures to support cells with complicated hierarchy definitions such as hexagons.

An important property of a regular grid is that the connectivity of cells and vertices are implicit and there is no need to explicitly store the connection of cells or vertices. Each cell is mapped to a face with 3D vertices by mapping ϕ. A patch is a 2D regular grid mapped onto a 3D surface (see Figure 3). Similar to the coordinate systems of lattices [25], we define the coordinate systems on the regular grids (2D domains) by taking one vertex as the origin and two incident edges to be the coordinate axes [25, 24]. This coordinate system provides an integer indexing that can be used within a data structure to address a 2D array storing the 3D locations of each vertex (acting as ϕ).

Grid G_i, $i \in \{t, q, h\}$ denotes triangular, quadrilateral, or hexagonal when i is t, q, or h respectively. Given a grid G_i, we can convert it to grid G_j $j \in \{t, q, h\}$ by conversion $C^{i \to j}$. G_i and G_j have their own coordinate systems (O_i, U_i, V_i) and (O_j, U_j, V_j) in which O is the origin and U and V are axes. It is possible to find a transformation $T^{i \to j}$ mapping (O_i, U_i, V_i) to (O_j, U_j, V_j) (Figure 4). $T^{i \to j}$ is found by using the algebraic relationship

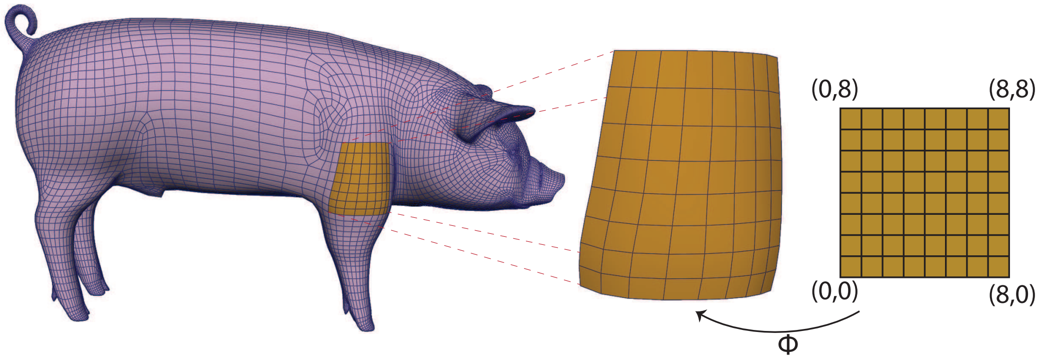


Figure 3: An eight by eight 3D patch on a mesh that is mapped to a 2D grid.

between axes of (O_i, U_i, V_i) and (O_j, U_j, V_j). Using $T^{i \to j}$, the coordinate of any point in the coordinate system of G_i can be found in G_j. Choosing coordinate systems on 2D domains can be arbitrary. However, our method remains valid for any arbitrarily defined coordinate system, although the applied transformations may be slightly different. In the following sections, we introduce some conversions as well as the notations used throughout the paper. These simple conversions enable one to switch between regular grids to benefit from the advantages offered by different grids.

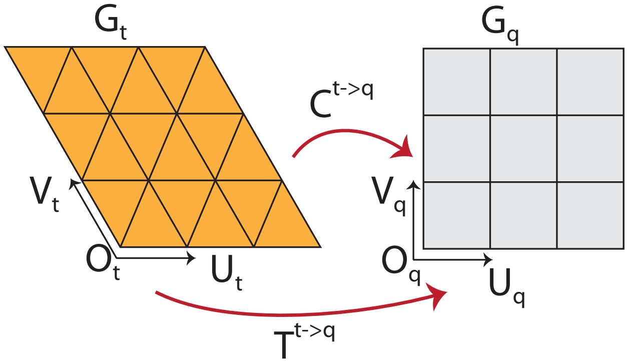


Figure 4: Grid G_t is converted to G_q by conversion $C_P^{t \to q}$ and (O_t, U_t, V_t) is mapped to (O_q, U_q, V_q) by $T^{t \to q}$.

3.1. Pairing Conversion

This conversion is basically a simple *pairing triangles* and denoted by $C_P^{t \to q}$ as the subscript refers to the name of conversion (pairing) and the superscript denotes that it converts triangular grids to quadrilaterals. Here, we first define the notation and then provide the conversion.

A quadrilateral grid G_q consists of a set C_q of quad cells with four vertices (Fig 5). A vertex O of G_q is chosen as the origin and edges U_q and V_q incident with O_q form axes of G_q. To distinguish between indexing cells and vertices, we use $q(a, b)$ as the index of a vertex and $q[a, b]$ as the index of a cell. A vertex v in P_q has index $q(a, b)$ where (a, b) is the integer coordinate of v in (O_q, U_q, V_q). A quad cell has index $q[a, b]$ if its vertices have indices $q(a, b)$, $q(a + 1, b)$, $q(a, b + 1)$, and $q(a + 1, b + 1)$.

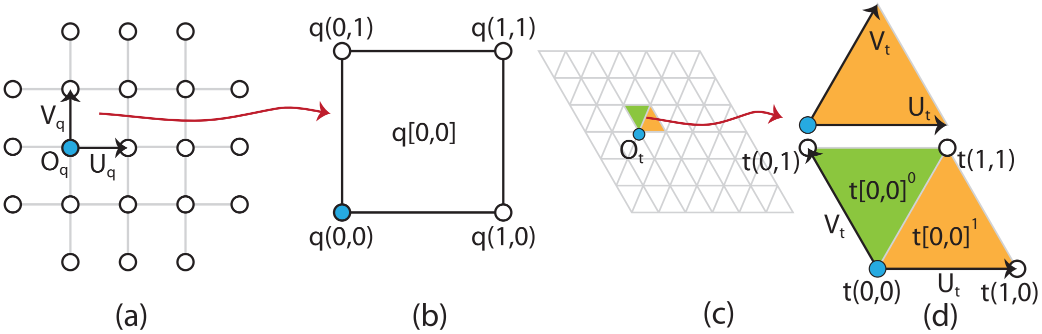


Figure 5: (a) A quadrilateral grid, its origin O, and two axes U_q and V_q. (b) Indices of vertices of a quadrilateral cell $q[0, 0]$. (c) A triangular grid and its origin O. (d) Top: two axes U_t and V_t with 60° difference. Bottom: U_t and V_t with 120° difference, and indices correspondent to these axes for cells and vertices. Green and orange cells have indices $t[0, 0]^0$ and $t[0, 0]^1$ respectively.

Similarly, to define a coordinate system for a triangular grid G_t, (O_t, U_t, V_t) illustrated in Figure 5 (d) is used. There exist two triangular cells associated with vertex $t(a, b)$. Hence, we add a superscript to differentiate between them: $t[a, b]^0$ and $t[a, b]^1$ illustrated in Fig 5. Using this notation, two triangular cells $t[a, b]^0$ and $t[a, b]^1$ can be paired to a quad $q[a, b]$ if the edge connecting $t(a, b)$, and $t(a + 1, b + 1)$ is removed (Figure 5 (d)). This conversion from triangular cells to quads is called the *pairing conversion*.

3.2. Unpairing Conversion

Unpairing conversion $C_U^{q\to t}$ is the inverse of $C_P^{t\to q}$. Given quadrilateral grid G_q, connect all vertices $q(a, b)$ to $q(a + 1, b + 1)$ and obtain a triangular grid G_t. We can also define this conversion by connecting vertices along $q(a+1, b)$ to $q(a, b+1)$. We refer to the first case by referring to $C_U^{q\to t}$ unless otherwise is indicated.

3.3. Dual Conversion

Hexagonal grids have also applications in parametrization, Earth representation, and surface modeling. A simple way to obtain hexagonal grids is taking the dual of triangular grids $C_D^{t \to h}$. To index hexagonal cells, we use the *hexagonal coordinate system* [37, 38] (Fig 6 (a)). The origin O_h is chosen as the midpoint of an arbitrary hexagonal cell h and axes U_h and V_h have 120° difference connecting O_h to the midpoints of two neighboring cells.

A hexagonal cell gets index $h[a, b]$, if its midpoint is a and b steps from O_h along U_h and V_v respectively. Using (O_h, U_h, V_h) to index vertices does not provide our desired integer indices. Therefore, a specific coordinate system for hexagonal vertices can be defined (see Fig 6 (b)). To convert hexagonal grid G_h to a triangular grid G_t ($C_D^{h \to t}$), we triangulate the midpoints of hexagonal cells by forming edges between the midpoints of all adjacent hexagons (Fig 6 (c)). Similarly, to define $C_D^{t \to h}$, the midpoints of triangular cells - are taken to be the vertices of the hexagonal cells, and edges are constructed by connecting the midpoints of adjacent triangles. After finding G_t from G_h by dual conversion, one can pair the triangles and obtain a quadrilateral grid with a reasonable computational cost (Fig 6 (d)). A complete pairing of triangles is possible and is computable in $O(Mlog^4 M)$ where M is the number of triangles [39, 40].

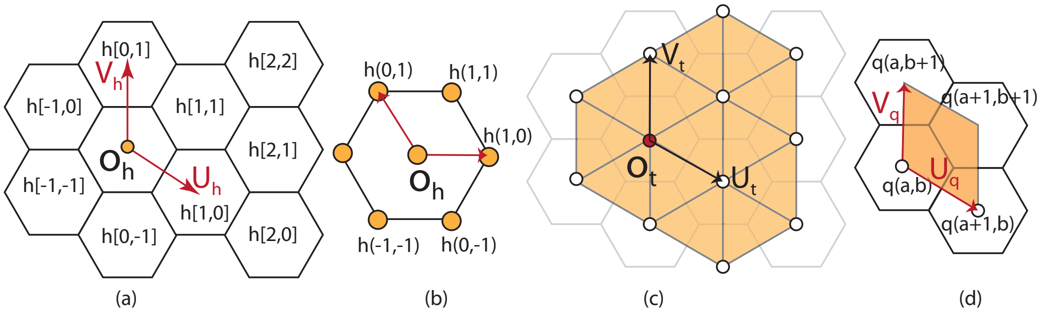


Figure 6: (a) Hexagonal coordinate system and its corresponding indices for hexagonal cells. (b) Coordinate system to index vertices and its corresponding indices. (c) Dual conversion of hexagons to triangles. (d) A quad corresponding to hexagonal grids, its axes and indices of its vertices.

In the dual conversion, the coordinates of cells in G_h correspond to the coordinate of vertices in G_t (Figure 7). In other words, for any vertex $t(a, b)$ in G_t, there exists a hexagonal cell with index $h[a, b]$ in G_h. In addition, each triangular cell in G_t corresponds to a vertex in G_h. Therefore, vertices of hexagonal cells can be indexed using the indices of six triangular cells (Fig 7 (d)).

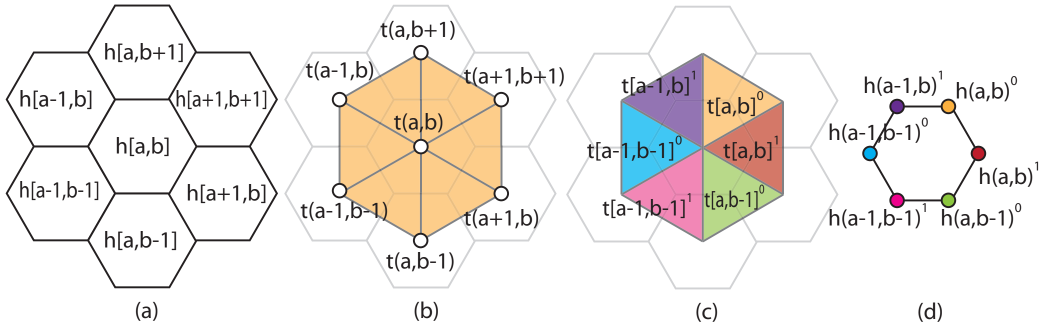


Figure 7: (a) Indices of hexagonal cells. (b) Corresponding triangular mesh of (a) and indices of its vertices. (c) Indices of triangular cells of the grid in (b). (d) Indices of vertices of a hexagonal grid. These indices correspond to triangular cells (c).

4. Hierarchical Grid Conversion

Various refinements can be applied to an object in order to obtain a more detailed or smoother object. To benefit from each grid at different levels of refinement, we combine conversions and refinements within the hierarchical grid conversion (Fig 8). Conversions and refinements may impose a transformation on the grid coordinates. Therefore, a total transformation T exists for the hierarchical grid conversion that is the composition of all involved transformations ($T = T_i^{a\to b} \circ T_R^n \circ T_j^{b\to c}$).

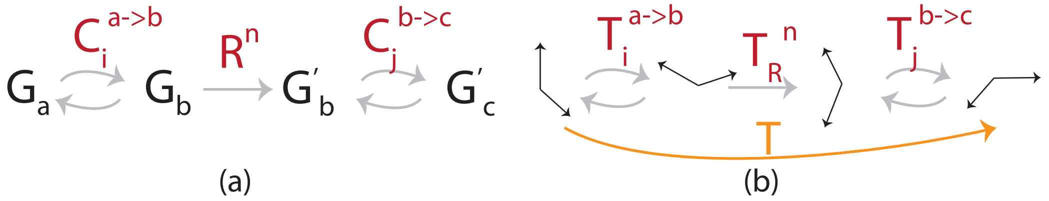


Figure 8: (a) General framework of having conversions and refinements. (b) Transformations imposed by grid conversions and refinements. Here, R is the refinement.

Using this framework, we can define useful operations and concepts. For instance, we can obtain new conversions, refinements, hierarchies, and data structures for grids. In the following, we discuss usability and applications of this framework and provide some examples.

4.1. Split and Aggregation Conversion

Our hierarchical grid conversions can be a base to define new conversions. For instance, if G_h is converted to G_t by dual conversion $C_D^{h \to t}$ and then a 1-to-3 refinement is applied to G_t to get $\acute{G}_t$, a high resolution grid (see Appendix B), the whole process converting G_h to $\acute{G}_t$ can be another type of conversion. We call this conversion *split conversion* ($C_S^{h \to t}$) (Fig 9). In fact, the set of all midpoints and vertices of G_h are triangulated by connecting the midpoint of each hexagonal cell to its vertices (Figure 10). It is possible to derive the transformation T imposed by the hierarchical grid conversion. As an example, we derive the transformation imposed by $C_S^{h \to t}$. Other transformations are fairly similar. $\acute{G}_t$ has the same origin as G_h and can take U_t and V_t as the basis vectors. We can find transformation mapping these grid coordinate systems by finding the equality of axes of each coordinate system. In this case, $U_h = U_t - V_t$ and $V_h = U_t + 2V_t$ (Fig 10 (c)). Then, a hexagonal cell with index $h[a, b]$ has a midpoint on vertex $t(c, d)$ in $\acute{G}_t$ where $(c, d) = a(1, -1) + b(1, 2) = (a + b, 2b - a)$.

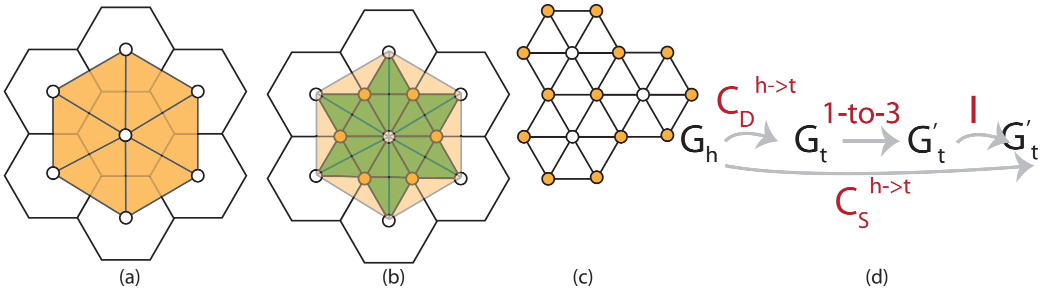


Figure 9: (a) Hexagonal grid G_h and triangular grid G_t obtained from dual conversion $C_D^{h \to t}$. (b) 1-to-3 refinement applied on G_t to obtain $\acute{G}_t$. (c) Split conversion $C_S^{h \to t}$. (d) $C_S^{h \to t}$ is made using the multiresolution framework.

The inverse of $C_S^{h \to t}$ is called *aggregation conversion* ($C_A^{t \to h}$) which reduces the resolution. $C_A^{t \to h}$ replaces six triangular cells sharing a common vertex into one hexagonal cell. Vertices of h are found by adding six vectors as shown in Fig 11(b). Note that choosing different origins in $\acute{G}_t$ results in different G_h. Considering the exact choice of h, only three main options are possible to completely cover $\acute{G}_t$, as illustrated in Fig 11 (c).

4.2. New Refinements

Refinements are generally defined for a specific type of grid. However, using hierarchical grid conversion, we can define new refinements for grid G_a

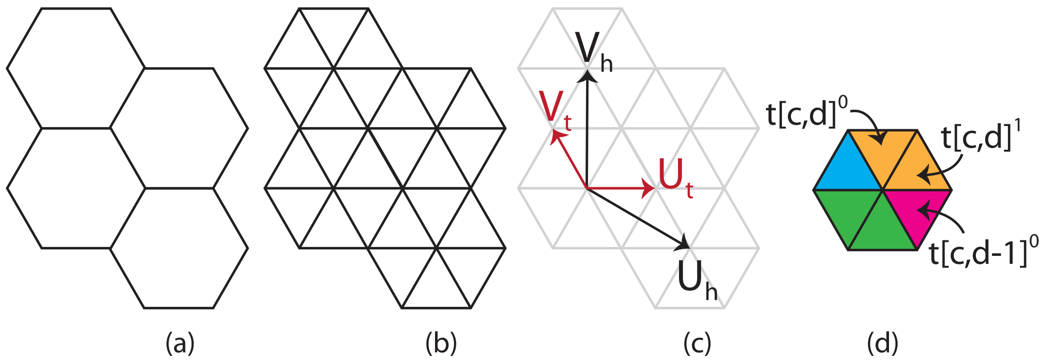


Figure 10: (a) Part of a hexagonal grid G_h. (b) Triangulation of (a) by split conversion to get $\acute{G}_t$. (c) Coordinate systems of G_h, and $\acute{G}_t$ illustrated by black and red arrows respectively. (d) A hexagonal cell is converted into six triangular cells. Indices of some triangular cells are shown.

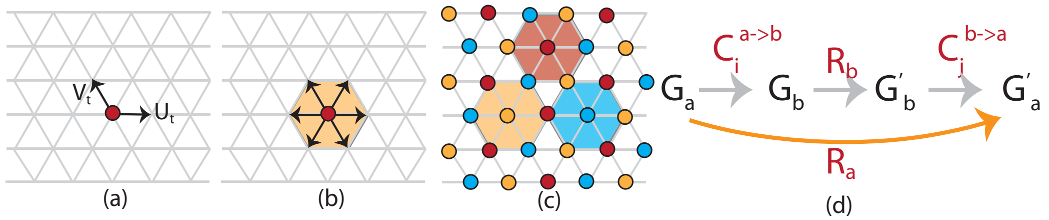


Figure 11: (a) G_t, its origin and coordinate system. (b) Vectors chosen to aggregate a hexagonal cell in G_t. (c) Midpoints of hexagonal grids cover the entire vertices of G_t. (d) Defining refinement R_a for G_a using refinement R_b defined for G_b.

using the existing refinement for grid G_b. This happens when G_a is converted to G_b by $C_i^{a\to b}$, G_b is refined and $\acute{G}_a$ obtained by conversion $C_j^{b\to a}$. The whole process to convert G_a to $\acute{G}_a$ is a refinement R_a defined for G_a (see Fig 11 (d)). Note that transformation R_a is the combination of all transformations $C_i^{a\to b}$, R_b, and $C_j^{b\to a}$. In the following, we discuss how to obtain refinements for grids using known regular refinements for a specific type of grid.

4.2.1. Quadrilateral Refinements

A variety of refinements have been proposed for quadrilateral grids. 1-to-4 refinement applied in Catmull-Clark subdivision is the most common one. However, 1-to-2 refinement used in $\sqrt{2}$ subdivision (see Appendix A) as well as 1-to-5 refinement has also been proposed for quadrilateral grids (see Appendix C) [41, 42]. It is possible to extend the variety of quadrilateral refinements using defined refinements of other grids. For example, using refinements defined for triangular grids, we show how to obtain a greater variety of quadrilateral refinements.

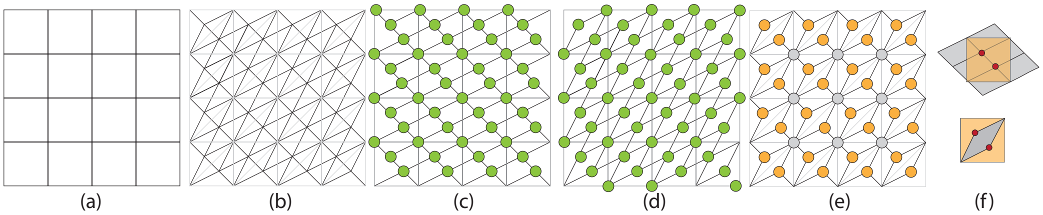


Figure 12: (a) G_q, (b) Refined G_t by 1-to-3 refinement. (c), (d), and (e) Three types of $\acute{G}_q$ (quadrilateral 1-to-3 refinements) can be made. Note that e is a semi-regular refinement. (f) Top/Bottom: Cell refinements for (c)/ (d),(e).

To define quadrilateral refinements using triangular refinements, we use the sequence of conversions and refinements illustrated in Figure 11 (d). For instance, if we apply $C_U^{q \to t}$ and use triangular 1-to-3 refinement on G_t and then apply $C_P^{t \to q}$, the whole process is a quadrilateral 1-to-3 refinement as illustrated in Figure 12. Note that depending on $C_P^{t \to q}$ three types of 1-to-3 refinements can be defined for quads as illustrated in Fig 12. These refinements are not regular since after refinement, quadrilateral cells are transformed to rhombic cells. However, there exists a strong regularity among the cells since all the cells have the same shape and area and the he area of cells is compressed with the same proportion of regular refinements. Note that the area of rhombic cells is equal to $\frac{A}{3}$ where A is the area of a regular quadrilateral cell. As some properties of such refinements are the same as regular refinements, we call them *semiregular* refinements.

Quadrilateral cells can be transformed to rhombic cells using a simple matrix transformation R. Matrix $R = \begin{pmatrix} \frac{2}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{2}{3} \end{pmatrix}$ is obtained for quadrilateral 1-to-3 refinement. Note that rows of matrix R are the coordinates of new inserted points after the refinement. The refinement illustrated in Figure 12 (e) produces vertices with valence three or six, so we call it *3-6* refinement. It is possible to define other quadrilateral refinements based on the hierarchical grid conversion. Similarly, we can define a 1-to-7 quadrilateral refinement (see Appendix D) if we replace 1-to-3 triangular refinement by 1-to-7 triangular refinement in the conversion (Fig

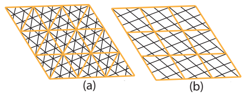


Figure 13: 1-to-7 refinement for triangular and quadrilateral grids.

13).

4.2.2. Triangular Refinements

Although triangular refinements are very well studied, we can define new refinements for triangular grids using quadrilateral refinements. Two instances of quadrilateral refinements are 1-to-2 and 1-to-5 refinements defined for quads (see Appendix A and Appendix C). To apply quadrilateral refinement R_q on a triangular grid G_t, we use pairing conversion $C_P^{t \to q}$ to obtain G_q, and then apply R_q on G_q to obtain $\acute{G}_q$. We can then apply the unpair conversion $C_U^{q \to t}$ to get $\acute{G}_t$ which is the refined version of G_t. Fig 14 and 15 illustrate this process when R_q is 1-to-2 or 1-to-5. Note that two types of $\acute{G}_t$ can be obtained: 4-8 refinement and triangular 1-to-2 refinement.

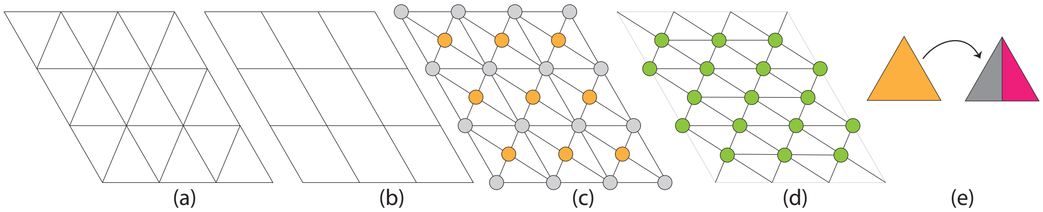


Figure 14: (a) G_t. (b) G_q. (c) $\acute{G}_t$ (4-8 refinement). (d) $\acute{G}_t$ (triangular 1-to-2 refinement). (e) 1-to-2 cell refinement for triangles.

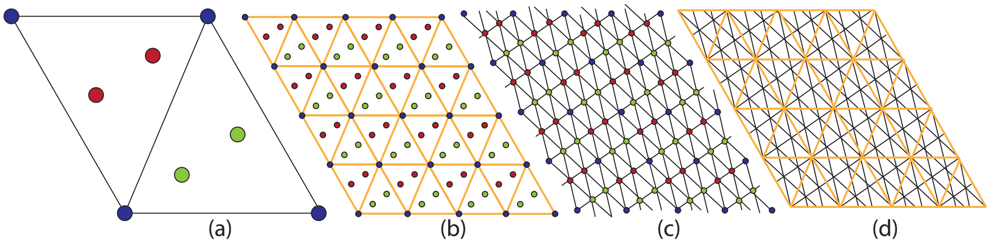


Figure 15: (a) The location of inserted vertices. (b) New vertices on a coarse grid. (c) new edges are drawn. (d) Triangular 1-to-5 refinement.

4.2.3. Hexagonal Refinement

Hexagonal refinements are often used in subdivision surfaces or image processing [8, 38]. Using existing triangular refinements and our conversion technique, the number of hexagonal refinements can be expanded. Given a hexagonal grid G_h, we convert G_h to G_t using split conversion ($C_S^{h \to t}$). We then apply triangular refinement on G_t (see Fig 16) and make a new hexagonal grid $\acute{G}_h$ using one of the aggregation conversions ($C_A^{t \to h}$). It is possible to define hexagonal refinements using a variety of triangular refinements. Fig 17 illustrate hexagonal 1-to-4, and 1-to-7 refinements. We have three possible aggregation conversions and each produces a different refinement.

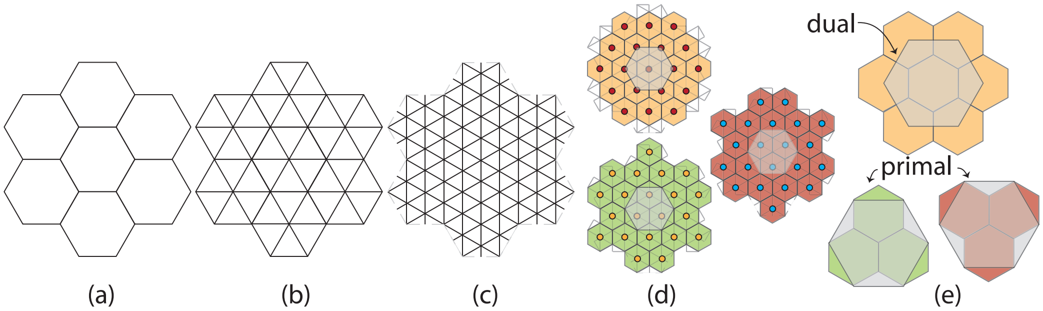


Figure 16: (a) Hexagonal grid G_h. (b) Split conversion is applied on G_h and G_t is obtained. (c) 1-to-3 on G_t results in triangular grid $\acute{G}_t$.

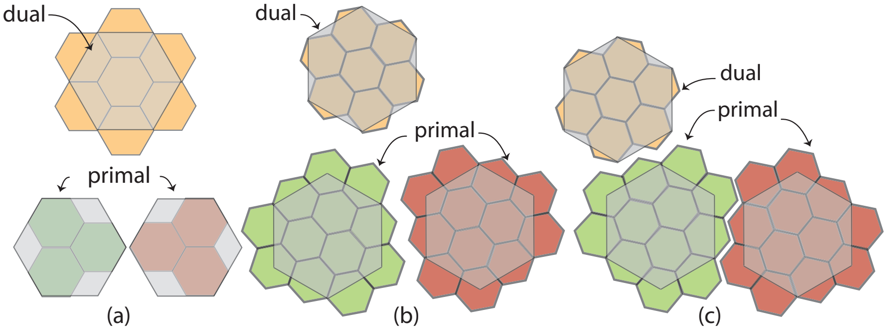


Figure 17: Dual and primal 1-to-4 hexagonal refinements (a), 1-to-7 hexagonal refinements with 19° (b) and −19° (c) rotations.

Our framework is general enough to introduce more variety of refinements by changing conversions and refinements. Here, we have provided a set of examples to describe the framework. Another instance of these refinements is produced when we use duality conversion and 1-to-2 refinement on the triangular grid, we can define a 1-to-2 refinement for hexagons as illustrated in Figure 18. In this paper, in addition to introducing a general framework to define refinements for grids, we achieved refinements that are novel to our knowledge. Examples of these novel refinements are primal 1-to-7, and 1-to-2 hexagonal refinements, 1-to-3 and 1-to-7 quadrilateral refinements as well as 1-to-2 and 1-to-5 triangular refinements.

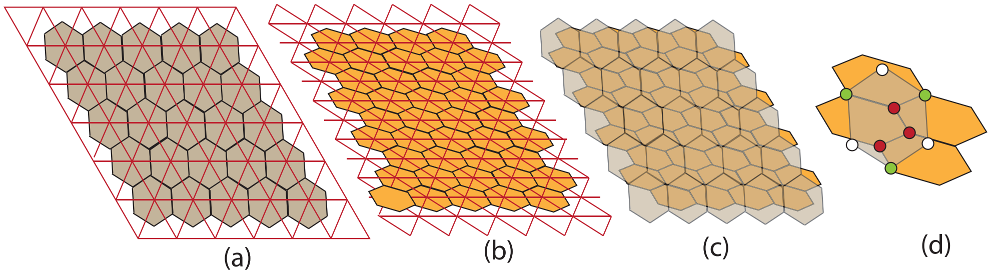


Figure 18: (a) G_h and its dual G_t (b) $\acute{G}_h$ and its dual $\acute{G}_t$ refined by 1-to-2 refinement. (c) G_h and $\acute{G}_h$ on top of each other. (d) Cell 1-to-2 hexagonal refinement. Red vertices are newly inserted, white vertices are removed, and green vertices are preserved after a refinement.

Although some of the refinements that are proposed seem skewed after one iteration of the refinement, it is possible to cancel out these effects after an additional application of refinements. This way, a perfect regular gird is obtained. For instance, Fig 19 illustrates how a perfect grid is obtained after two iteration of the refinement in Fig 12.

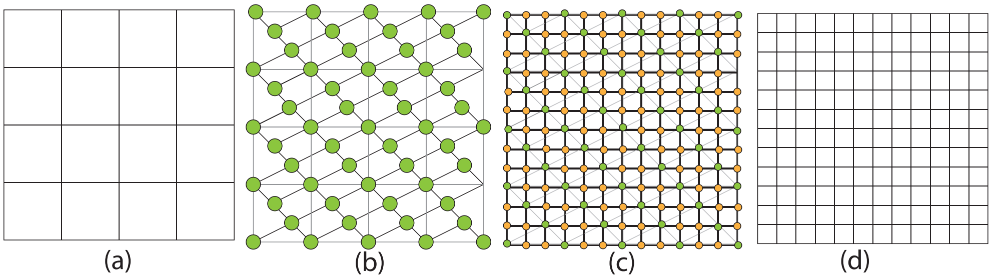


Figure 19: (a) A quadrilateral grid. (b) 1-to-3 refinement on the quadrilateral grid in (a). (c) An additional application of 1-to-3 refinement on the grid in (b). (d) The result of two applications of 1-to-3 refinement on a quadrilateral grid is a regular quadrilateral grid.

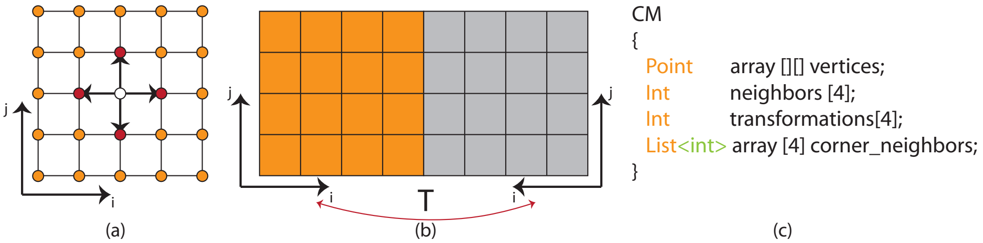
5. Indexing Semiregular Models

A semiregular model is created by applying refinements on a mesh with arbitrary connectivity. It is desired to have an efficient data structure for these meshes. In [10, 11], a data structure called Atlas of Connectivity Maps (ACM) has been proposed for triangular and quadrilateral semiregular models. In this section, we discuss how to use hierarchical grid conversions to extend ACM to hexagons and also some new refinements that have not been discussed in [10, 11]. Note that ACM is an example of a hierarchical data structure defined based on a particular grid (quads) and it can be extended to other cells thanks to the hierarchical grid conversions proposed in this paper. We can use the same concept to extend other hierarchical data structures such as quadtrees to support other types of cells and refinements.

5.1. Review of ACM

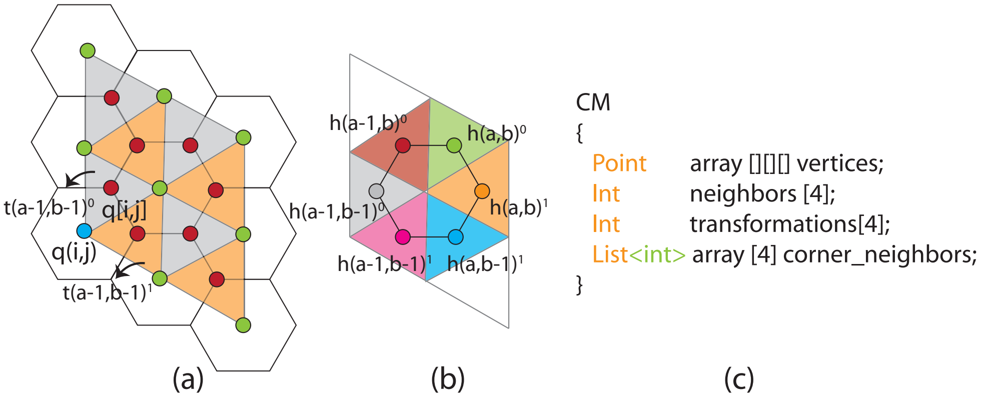
A semiregular model consists of a number of regular patches connected to each other. In ACM, the connectivity of each patch can be captured by a simple 2D grid with a 2D indexing method and then the geometry of vertices can be recorded in a 2D array. The indexing is based on a simple coordinate system assigned to each 2D grid.

Connections within each patch are implicit and therefore connectivity queries between internal vertices of each patch are addressed by simple neighborhood vectors. A transformation is used to traverse from one patch to another (Fig 20). These simple 2D grids and their interconnections are maintained through the resolutions (for all types of refinements) by applying a transformation (imposed by the refinement) to the coordinate system of each 2D grid. To capture this information, ACM has a set of elements illustrated in Figure 20 (c). ACM also supports triangular refinements using pairing conversion $C_P^{t \to q}$.



```
CM
{
  Point       array [][] vertices;
  Int         neighbors [4];
  Int         transformations[4];
  List<int> array [4] corner_neighbors;
}
```

Figure 20: (a) A connectivity map. Vertices are connected to their neighbors by neighborhood vectors. (b) Transformation T is used to traverse from one connectivity map to its neighbors. (d) Elements of a connectivity map. In ACM, we have a list of connectivity maps (CM) for the entire model.



```
CM
{
  Point       array [][][] vertices;
  Int         neighbors [4];
  Int         transformations[4];
  List<int> array [4] corner_neighbors;
}
```

Figure 21: (a) Vertices of a quadrilateral and hexagonal patch are drawn in green and red respectively. Vertex $q(i, j)$ (drawn in blue) is the bottom-left corner of cell $q[i, j]$. Cell $q[i, j]$ is split into two triangles $t[i, j]^0$, and $t[i, j]^1$ illustrated in grey and orange respectively (b) Vertices of a hexagon and their associated triangular cells. (c) Modified ACM for vertices of hexagonal cells. We use a 3D array instead of 2D arrays for storing vertices.

5.2. ACM for Hexagons

As ACM is an efficient data structure for semiregular models, it is desired to use it for hexagonal cells, despite its original formulation in terms of triangles and quads. Hierarchical grid conversions provided in this paper can help to extend ACM for supporting hexagons. To apply ACM to hexagons, we use dual conversion $C_D^{h \to t}$ and obtain a triangular patch with $n \times n$ vertices from an $n \times n$ patch of hexagonal cells (Fig 21 (a)). This way, for each triangular cell, a unique vertex of hexagonal cells exists (Figure 21 (b)). A quadrilateral patch with $n \times n$ vertices can be represented by a 2D array in which the (i, j) entry of the array refers to the bottom-left vertex of cell $q[i, j]$ (see Fig 21 (a)). As discussed earlier, each quad $q[i, j]$ may be split into triangular cells $t[i, j]^0$ and $t[i, j]^1$. Each triangular cell $t[i, j]^k$ ($k = 0$ or 1) corresponds to a vertex in a hexagonal grid. As a result, vertices of a hexagonal grid are represented in a 3D array in which entry (i, j, k) refers to triangular cell $t[i, j]^k$ obtained from dual conversion of the hexagonal grid.

We can modify the location array of vertices in ACM to a 3D array to support hexagonal grids (see Fig 21 (c)). Transformation between connectivity maps and handling the neighborhood queries are similar to the case of triangular cells. Figure 22 illustrates a hexagonal mesh at two successive resolutions and its connectivity maps. By extending ACM to support hexagonal

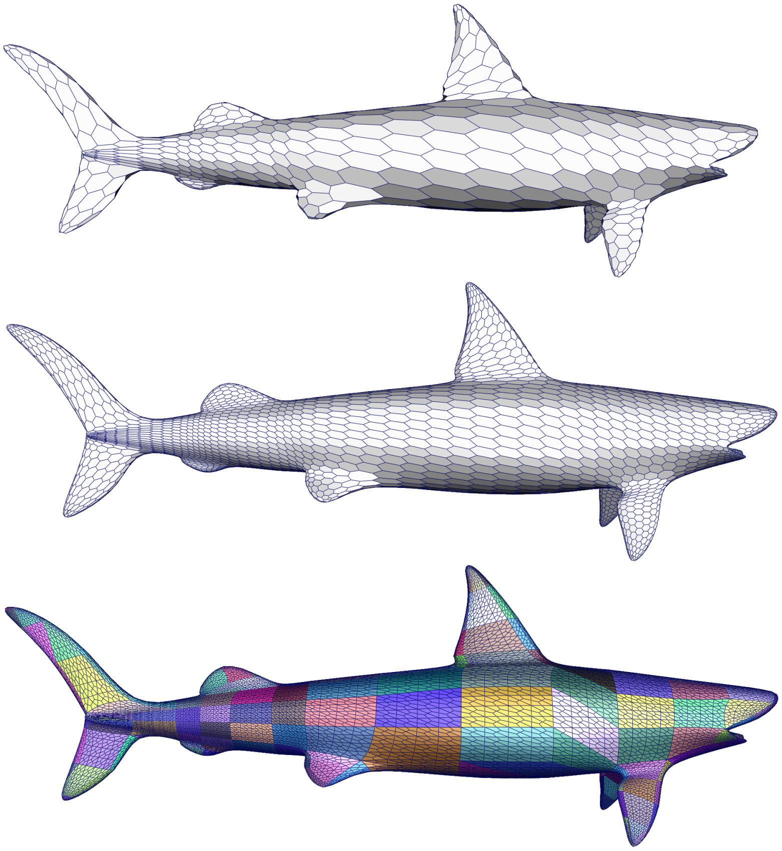


Figure 22: A shark at two successive resolutions (top) and its corresponding connectivity maps at the bottom.

grids, we can benefit from the advantages of ACM, such as speed in handling connectivity queries and the efficient support of hierarchical queries between the vertices and cells.

5.3. ACM Refinement Extension

In [10, 11], only regular refinements (with specific rotation) have been discussed. By providing more possible refinements through hierarchical grid conversions, the question is how to extend ACM to support these refinements. Here, we discuss how ACM can be modified to support more variety of refinements. In the following, we discuss two categories of refinements that have not been explored in [10, 11].

1-to-5 and 1-to-7: In [10], refinements are categorized based on their imposed transformation to the grid of subsequent resolutions. Based on these categorizations, both 1-to-5 and 1-to-7 refinements belong to the category of —scaling, rotation, no translation— that means these refinements impose only scaling and rotation to the subsequent grids. In ACM, rotation of refinements are eliminated in order to benefit from the connectivity information of the first resolution. In [10], 1-to-2 refinement is presented as an instance of this category by scaling the connectivity map by two. This way, at odd resolutions, some empty indices exist that are filled by new vertices at the next even resolution. In fact, after two resolutions, the connectivity maps are simply scaled by two without any rotation involved. This works due to the cancellation of the 45° rotation of 1-to-2 refinement after two resolutions.

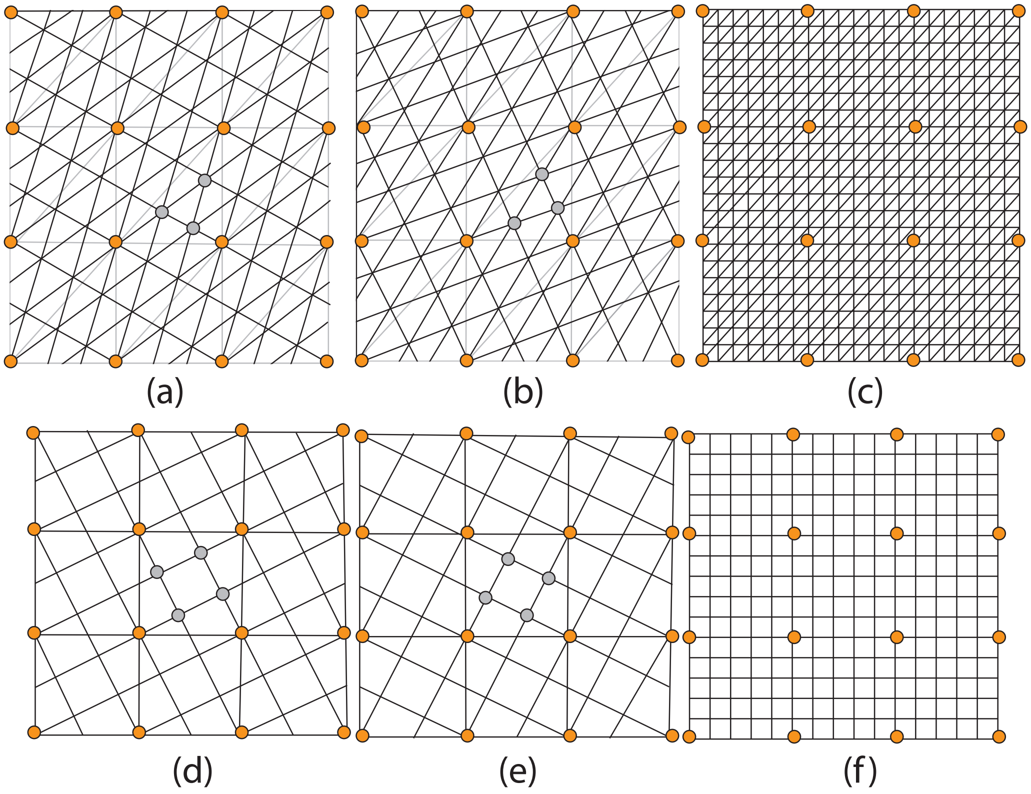


Figure 23: (a), (b) 1-to-7 refinement with +19° and −19° rotations. (c) After one level of refinement in (a) followed by the refinement in (b), the rotation is discarded. (c), (d) 1-to-7 refinement with +25° and −25° rotations. (e) Rotation is canceled out.

1-to-5 and 1-to-7 refinements are not aligned after two resolutions since the rotations are not canceled out. However, two versions of these refinements exist with θ and $-\theta$ rotations (see Fig 23). To have aligned connectivity maps for these refinements, we can apply one level of refinement with θ followed by one level of refinement with $(-\theta)$ rotation. This way, rotations are canceled out, and we get a refinement with the same factor but without a rotation (see Figure 23). Note that neighborhood vectors that connect a vertex to its neighbors are different for even and odd resolutions in refinements that impose rotations. Fig 24 (a) and (b) show neighborhood vectors for 1-to-5 and 1-to-7 refinements.

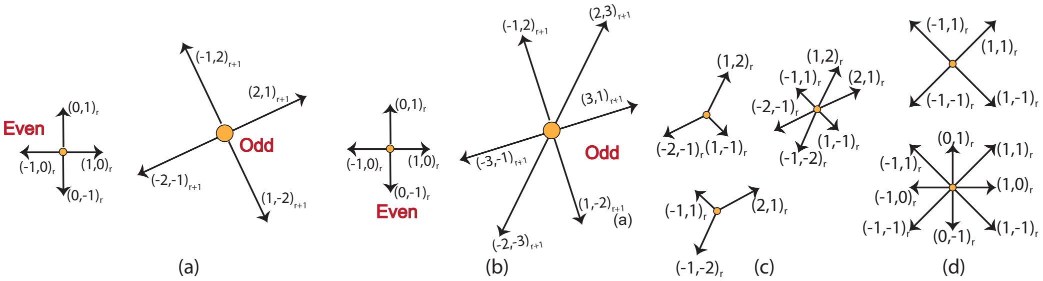


Figure 24: Neighborhood vectors for (a) 1-to-5, (b) 1-to-7, (c) 3-6 and (d) 4-8 refinements.

4-8 and 3-6: 4-8 and 3-6 semiregular refinements have two types of neighborhood vectors exist based on the type of vertex. Figure 24 (c) and (d) illustrate neighborhood vectors for vertices of 4-8 or 3-6 refinement. Connectivity maps are scaled by two and three every other resolution for 4-8 and 3-6 refinements respectively (see Fig 26).

6. Discussion and Results

Using the hierarchical grid conversions, we can define new refinements that may be applied on regular grids. By alternating between conversions and refinements, we can also apply subdivision methods that are defined for specific results. For example, we can apply 1-to-3 refinement on quadrilateral grids with the smooth filters defined for the $\sqrt{3}$ subdivision method for triangular grids (see Figure 25). We also extend ACM to support hexagonal meshes (see Fig 22) as well as more variety of refinements such as, 1-to-7, 1-to-5, and 4-8 subdivision (Figure 26). As a result, ACM, which efficiently handles connectivity and hierarchical queries on semiregular models, can be applied to a greater variety of refinements and grid types. Our proposed conversions can also be used to efficiently switch between grids as needed by the application. For example, the hexagonal meshes that are common in Earth representations [43] can be converted to a triangular mesh for efficient rendering.

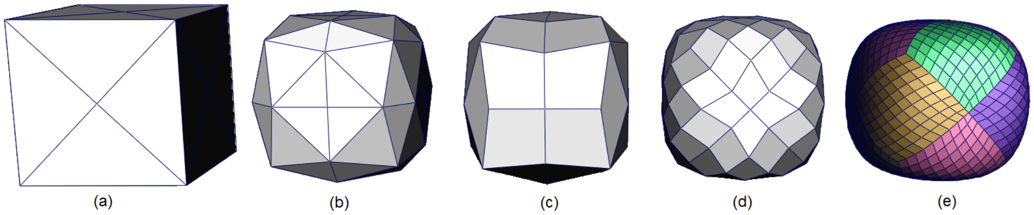


Figure 25: (a) A triangulated cube. (b) Applying $\sqrt{3}$ subdivision on (a). (c) Applying unpair conversion on (b) results a quadrilateral mesh refined by 1-to-3 and smoothed by $\sqrt{3}$ subdivision. (d) Applying the same process (unpair conversion and smoothing) on (c). (e) The cube in (a) after applying 1-to-3 refinement and $\sqrt{3}$ smoothing masks.

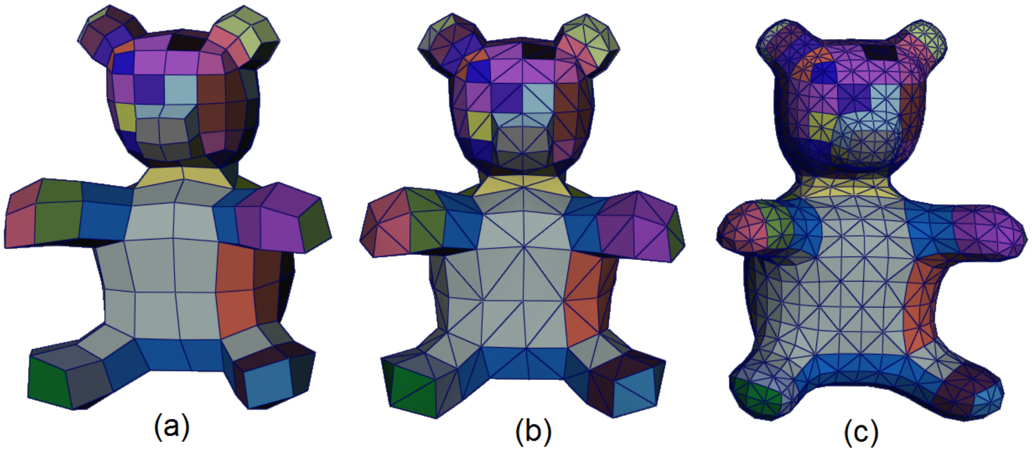


Figure 26: (a) A quadrilateral mesh. (b) Triangulating the mesh using pairing conversion. (c) 4-8 subdivision using ACM on the triangulated mesh.

7. Conclusion

In this paper, we present hierarchical grid conversions and use it to systematically define refinements. We extend an existing patch-based hierarchical data structure called ACM for handling connectivity queries of semiregular models, to support hexagonal semiregular models and some additional

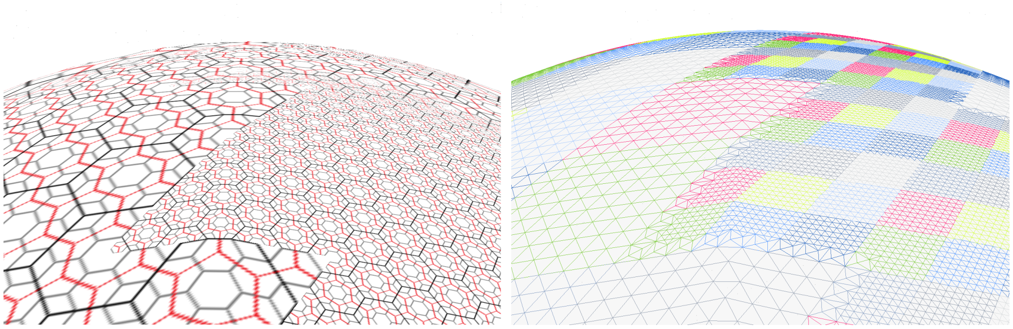


Figure 27: Left: A hexagonal globe with 1-to-3 refinement at three successive resolutions. Right: Triangulation of left using dual conversion using ACM.

refinements. From this enhanced support and newly defined conversions, we can best apply a given grid-type to the application-specific challenge.

8. Limitation and Future Work

Based on the conversions between regular grids, we can extend ACM to support hexagonal semiregular models as well as a wider range of refinements such as 4-8 and 1-to-7 refinements. However, ACM is specifically designed for semiregular models such as the meshes obtained from refinement, subdivision, multiresolution, or specific parametrization methods. A future work can be to modify this data structure to efficiently support arbitrary adaptive refinement. Additionally, we have proposed new types of refinements that may be used to generate smooth subdivision schemes.

Appendix A. 1-to-2 refinement

1-to-2 refinement used in quadrilateral $\sqrt{2}$ subdivision is composed of two stages of splitting and edge-flip. In the splitting stage, a cell is split by inserting the midpoint of quadrilateral cells. These vertices are then connected to the old vertices and old edges are flipped (see Figure A.28).

Appendix B. 1-to-3 refinement

1-to-3 refinement is used in $\sqrt{3}$ subdivision [44]. In this refinement, a vertex is inserted in the midpoint of each cell. These newly inserted vertices are connected to the old vertices by inserting new edges and old edges are flipped (see FigureB.29). Figure B.29 illustrates the steps of 1-to-3 refinement.

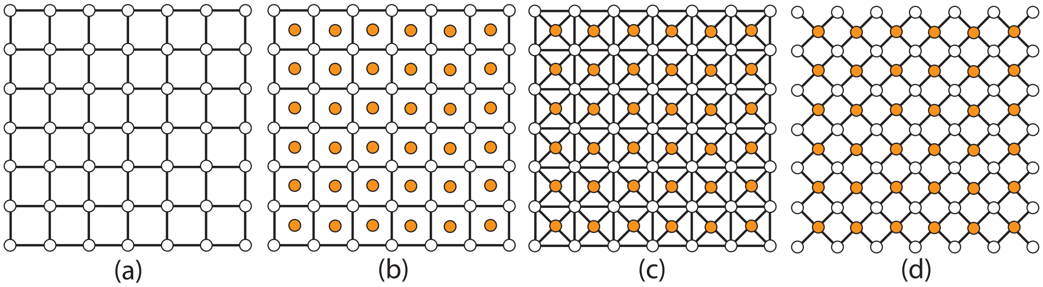


Figure A.28: (a) A quadrilateral grid. (b) Midpoints of cells are inserted. (c) New edges are drawn. (d) Old edges are removed.

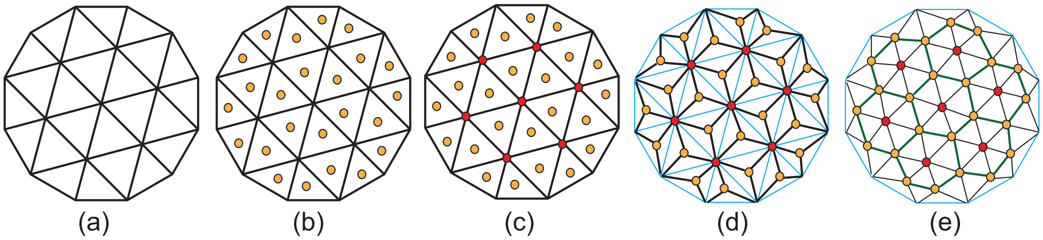


Figure B.29: (a) A triangular grid. (b) Midpoints of each triangular cell is inserted. (c) Old vertices are drawn in red. (d) New vertices are connected to old vertices. (e) Old edges are flipped.

Appendix C. 1-to-5 refinement

1-to-5 refinement has been introduced for quadrilateral grids. In this refinement, four vertices at locations $\left(\frac{2}{5}, \frac{1}{5}\right)$, $\left(\frac{4}{5}, \frac{2}{5}\right)$, $\left(\frac{1}{5}, \frac{3}{5}\right)$, and $\left(\frac{3}{5}, \frac{4}{5}\right)$ are inserted. The connectivity of vertices are changed afterwards as illustrated in Figure C.30.

Appendix D. 1-to-7 refinement

1-to-7 triangular refinement has been studied in $\sqrt{7}$ subdivision [42]. In this refinement, three vertices are inserted in a triangular cell, old edges are removed and new edges are formed. Consider a diamond with unit length edges. In triangle $t[0,0]^0$, three vertices with coordinates $\left(\frac{1}{7}, \frac{3}{7}\right)$, $\left(\frac{4}{7}, \frac{5}{7}\right)$, and $\left(\frac{2}{7}, \frac{6}{7}\right)$ are inserted and in $t[0,0]^1$, coordinates of three new vertices are $\left(\frac{5}{7}, \frac{1}{7}\right)$, $\left(\frac{3}{7}, \frac{2}{7}\right)$, and $\left(\frac{6}{7}, \frac{4}{7}\right)$. Figure D.31 illustrates the mask of 1-to-7 refinement and its topological changes.

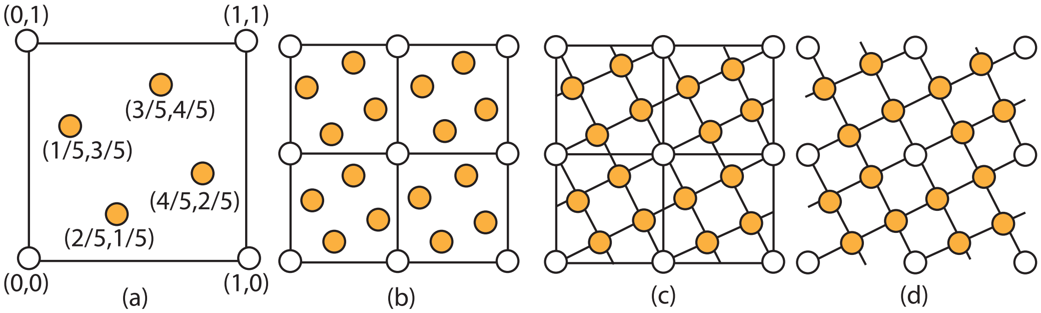


Figure C.30: (a) The location of inserted vertices. (b) New vertices on a coarse grid. (c) New edges are drawn. (d) Old edges are removed.

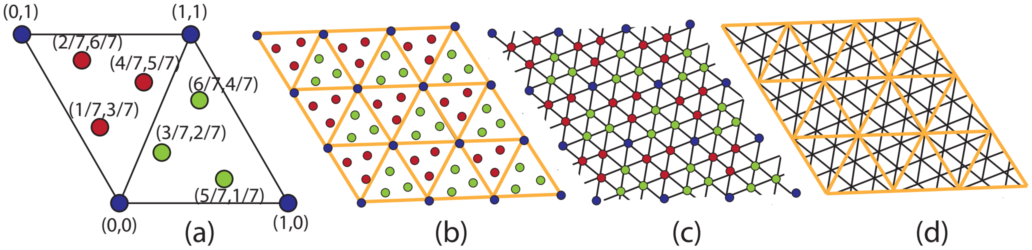


Figure D.31: (a) Masks of 1-to-7 refinement. (b) New vertices are inserted. (c) Old edges are removed and new and old vertices are connected. (d) Orange coarse triangular grid is refined by 1-to-7 refinement.

[1] M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf, Computational Geometry: Algorithms and Applications, 2nd Edition, Springer-Verlag, 2000.

[2] C. Loop, Smooth Subdivision Surfaces Based on Triangles, The University of Utah,Department of Mathematics.

[3] L. Piegl, W. Tiller, The NURBS book (2nd ed.), Springer-Verlag New York, Inc., New York, NY, USA, 1997.

[4] E. Catmull, J. Clark, Recursively generated B-spline surfaces on arbitrary topological meshes, CAD 10 (6) (1978) 350–355.

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

[6] X. He, et al., Hexagonal structure for intelligent vision, in: First International Conference of Information and Communication Technologies, ICIT 2005, 2005, pp. 52–64.

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

[8] J. Claes, K. Beets, F. Van Reeth, A corner-cutting scheme for hexagonal subdivision surfaces, in: Proceedings of the Shape Modeling International 2002 (SMI'02), SMI '02, IEEE Computer Society, Washington, DC, USA, 2002, pp. 13–20.

[9] K. Weiss, L. De Floriani, Simplex and diamond hierarchies: Models and applications, Computer Graphics Forum 30 (8) (2011) 2127–2155.

[10] A. Mahdavi-Amiri, F. Samavati, ACM: Atlas of connectivity maps for semiregular models, in: Proceedings of Graphics Interface 2013, 2013, pp. 99–107.

[11] A. Mahdavi-Amiri, F. Samavati, Atlas of connectivity maps, Computers & Graphics 39 (2014) 1–11.

[12] P. Lindstrom, D. Koller, W. Ribarsky, L. F. Hodges, N. Faust, G. A. Turner, Real-time, continuous level of detail rendering of height fields, in: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, SIGGRAPH '96, 1996, pp. 109–118.

[13] M. Guenette, A. J. Stewart, Triangulation of hierarchical hexagon meshes, in: Proceedings of the 2008 ACM symposium on Solid and physical modeling, SPM '08, 2008, pp. 307–313.

[14] D. Bommes, B. Levy, N. Pietroni, E. Puppo, C. S. a, M. Tarini, D. Zorin, State of the art in quad meshing, in: Eurographics STARS, 2012.

[15] E. Akleman, V. Srinivasan, E. Mandal, Remeshing schemes for semiregular tilings, in: Proceedings of the International Conference on Shape Modeling and Applications 2005, SMI '05, 2005, pp. 44–50.

[16] P. Bo, H. Pottmann, M. Kilian, W. Wang, J. Wallner, Circular arc structures, ACM Trans. Graph. 30 (4) (2011) 101:1–101:12.

[17] A. Schiftner, M. Höbinger, J. Wallner, H. Pottmann, Packing circles and spheres on surfaces, in: ACM SIGGRAPH Asia 2009 papers, SIGGRAPH Asia '09, 2009, pp. 139:1–139:8.

[18] M. Nieser, J. Palacios, K. Polthier, E. Zhang, Hexagonal global parameterization of arbitrary surfaces, in: ACM SIGGRAPH ASIA 2010 Sketches, SA '10, 2010, pp. 5:1–5:2.

[19] P. Oswald, P. Schröder, Composite primal/dual $\sqrt{3}$ subdivision schemes, Comput. Aided Geom. Des. 20 (3) (2003) 135–164.

[20] I. Guskov, A. Khodakovsky, P. Schröder, W. Sweldens, Hybrid meshes: Multiresolution using regular and irregular refinement, in: Proceedings of the Eighteenth Annual Symposium on Computational Geometry, SCG '02, ACM, New York, NY, USA, 2002, pp. 264–272.

[21] K. Weiss, L. De Floriani, Simplex and diamond hierarchies: Models and applications, Accepted to Computer Graphics Forum.

[22] L. Velho, Semi-regular 4-8 refinement and box spline surfaces, in: Computer Graphics and Image Processing, 2000. Proceedings XIII Brazilian Symposium on, 2000, pp. 131–138.

[23] J. Peters, U. Reif, The simplest subdivision scheme for smoothing polyhedra, ACM Trans. Graph. 16 (4) (1997) 420–431.

[24] M. Alexa, Refinement operators for triangle meshes, Comput. Aided Geom. Des. 19 (3) (2002) 169–172.

[25] I. P. Ivrissimtzis, N. A. Dodgson, M. A. Sabin, A generative classification of mesh refinement rules with lattice transformations, Comput. Aided Geom. Des. 21 (1) (2004) 99–109.

[26] N. Dodgson, An heuristic analysis of the classification of bivariate subdivision schemes, in: R. Martin, H. Bez, M. Sabin (Eds.), Mathematics of Surfaces XI, Vol. 3604 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2005, pp. 161–183.

[27] F. Destelle, C. Gérot, A. Montanvert, A topological lattice refinement descriptor for subdivision schemes, in: Proceedings of the 7th International Conference on Mathematical Methods for Curves and Surfaces, MMCS'08, Springer-Verlag, 2010, pp. 153–177.

[28] A. Vince, X. Zheng, Arithmetic and fourier transform for the PYXIS multi-resolution digital earth model, Int. J. Digital Earth 2 (1) (2009) 59–79.

[29] F. S. Ali Mahdavi-Amiri, Faraz Bhojani, One-to-two digital earth, in: 9th International Symposium on Visual Computing (ISVC 2013), Vol. 8034 of Lecture Notes in Computer Science, Springer, 2013.

[30] D. Zorin, P. Schröder, W. Sweldens, Interactive multiresolution mesh editing, in: Proceedings of the 24th annual conference on Computer graphics and interactive techniques, SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., 1997, pp. 259–268.

[31] L. Olsen, F. Samavati, R. Bartels, Multiresolution for curves and surfaces based on constraining wavelets, Computers and Graphics 31 (3) (2007) 449 – 462.

[32] K. Weiler, Edge-based data structures for solid modeling in curved-surface environments, Computer Graphics and Applications, IEEE 5 (1) (1985) 21 –40.

[33] P. Kraemer, D. Cazier, D. Bechmann, Extension of half-edges for the representation of multiresolution subdivision surfaces, The Visual Computer 25 (2009) 149–163.

[34] I. Gargantini, An effective way to represent quadtrees, Commun. ACM 25 (12) (1982) 905–910.

[35] M. Bertram, Biorthogonal loop-subdivision wavelets, Computing 72 (1-2) (2004) 29–39.

[36] A. Mahdavi-Amiri, F. Samavati, Connectivity maps for subdivision surfaces, in: GRAPP/IVAPP, 2012, pp. 26–37.

[37] W. E. Snyder, H. Qi, W. Sander, A coordinate system for hexagonal pixels, in: The International Society for Optical Engineering, Vol. 3661, 1999, pp. 716–727.

[38] L. Middleton, J. Sivaswamy, Hexagonal Image Processing: A Practical Approach (Advances in Pattern Recognition), Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[39] J. Petersen, Die theorie der regulren graphs, Acta Mathematica 15 (1891) 193–220.

[40] T. C. Biedl, P. Bose, E. D. Demaine, A. Lubiw, Efficient algorithms for petersen's matching theorem, Journal of Algorithms 38 (1) (2001) 110 – 134.

[41] I. Ivrissimtzis, N. Dodgson, M. Sabin, $\sqrt{5}$ subdivision, in: N. Dodgson, M. Floater, M. Sabin (Eds.), Advances in Multiresolution for Geometric Modelling, Mathematics and Visualization, Springer Berlin Heidelberg, 2005, pp. 285–299.

[42] C. K. Chui, Q. Jiang, R. N. Ndao, Triangular $\sqrt{7}$ and quadrilateral $\sqrt{5}$ subdivision schemes: Regular case, Journal of Mathematical Analysis and Applications 338 (2) (2008) 1204 – 1223.

[43] PYXIS Innovation, `http://www.pyxisinnovation.com` (March 2013).

[44] L. Kobbelt, $\sqrt{3}$ subdivision, in: SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques, 2000, pp. 103–112.