Icosahedral Maps for a Multiresolution Representation of Earth Data

Mohammad Imrul Jubair
Visualization and Graphics Group

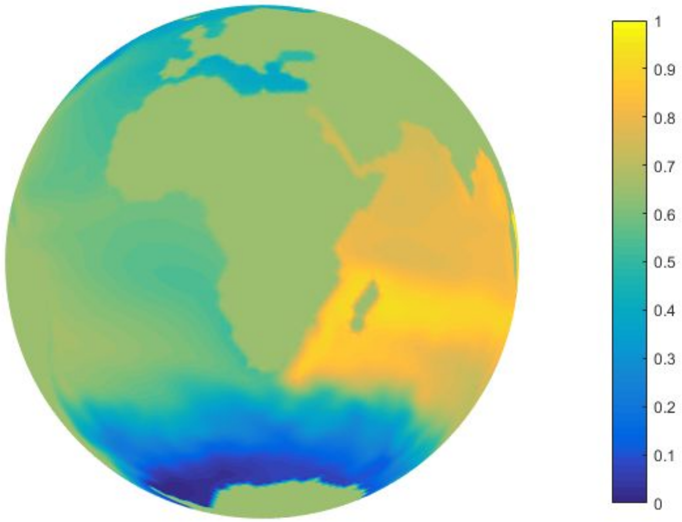
Outline

- Introduction
- Research Questions
- Contributions
- Literature Review
- Methodology
- Results
- Future Work

Introduction

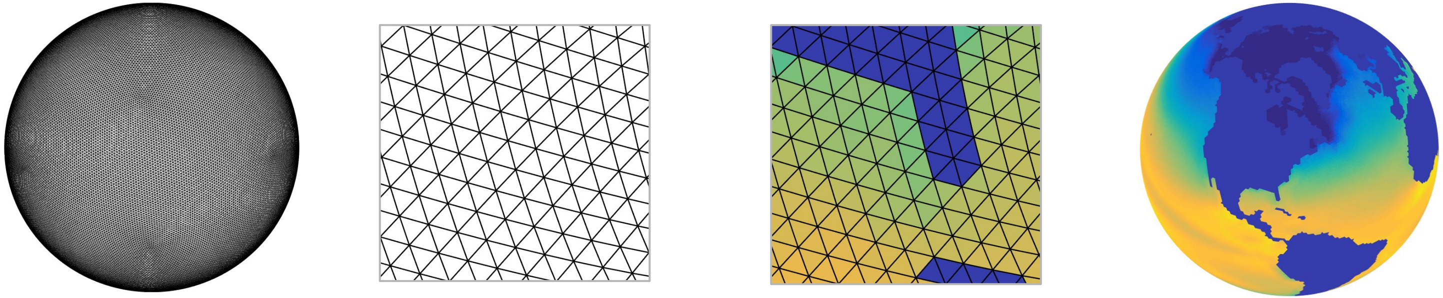
- The icosahedral non-hydrostatic (ICON) model is a digital Earth model that is used for numerical weather prediction.
- Designed via Discrete Global Grid Systems (DGGS).

Digital Earth



Discretization of DE

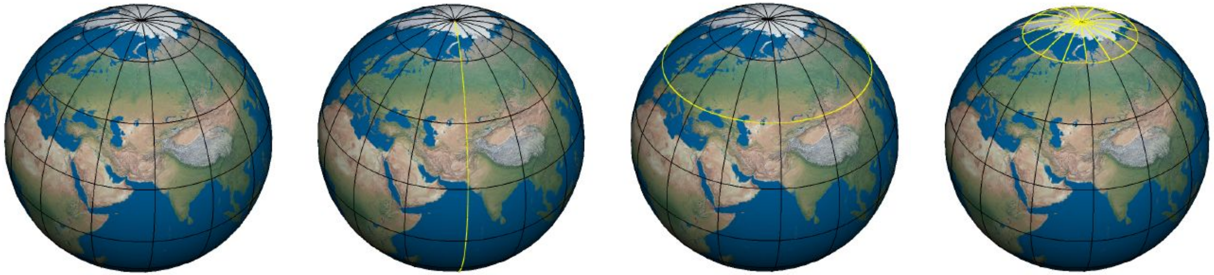
- Data are assigned to the cells of an underlying discretization of the Earth.
- Each cell represents a particular region and receives a unique index.
- Fast data access and/or hierarchical or adjacency queries.



Source: A Survey of Digital Earth

Parameterization

- Latitude / Longitude parameterization
- Problems: Cells becomes smaller approaching to the poles, poles are singularities, cells incident to the poles are triangular.



Source: Categorization and conversion for indexing methods of discrete global grid systems

DGGS

Five Design choices:

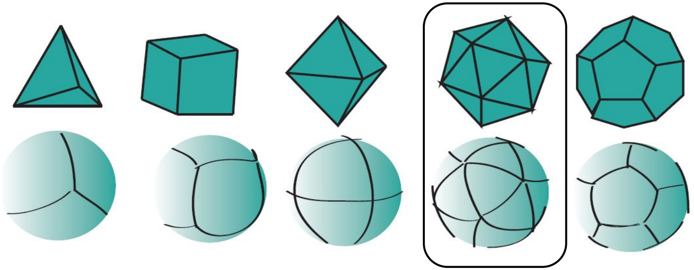
1. A base regular polyhedron.
2. A fixed orientation of the base regular polyhedron relative to the Earth.
3. A hierarchical spatial partitioning method defined symmetrically on a face (or set of faces) of the base regular polyhedron.
4. A method for transforming that planar partition to the corresponding spherical/ ellipsoidal surface.
5. A method for assigning points to grid cells

Source: Geodesic Discrete Global Grid Systems

DGGS: Base Polyhedron

- The tetrahedron, cube, octahedron, icosahedron and dodecahedron.

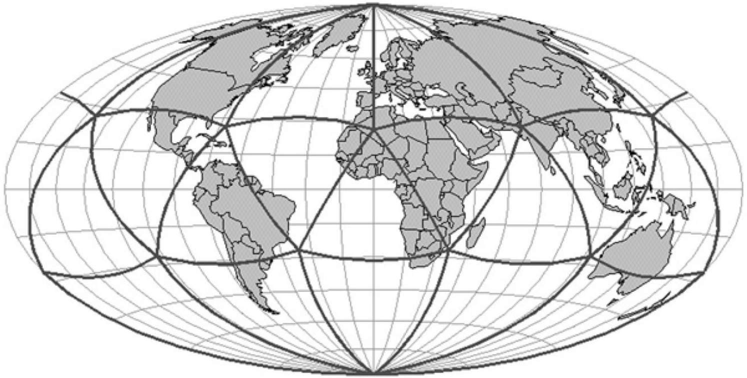
Icosahedron shows less triangular and area distortion under equal area projection.



Source: Geodesic Discrete Global Grid Systems

DGGS: Orientation

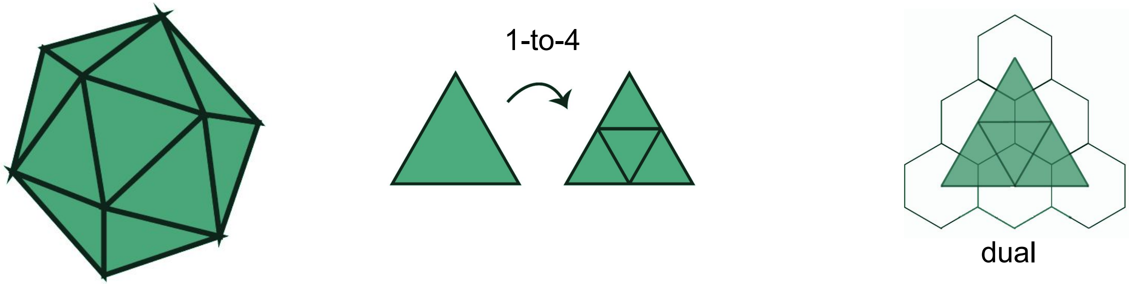
- In the case of the icosahedron, the most common orientation is to place a vertex at each of the poles and then align one of the edges emanating from the vertex at the north pole with the prime meridian.



Source: Geodesic Discrete Global Grid Systems

DGGS: Partitioning

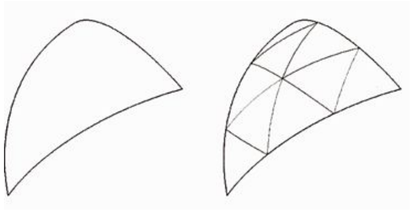
- Creating multiple resolution discrete grids
- Defining subdivision methodology on faces



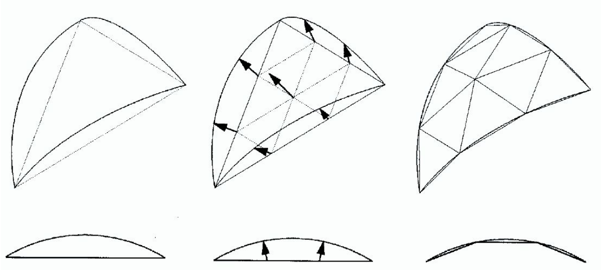
Source: Categorization and conversion for indexing methods of discrete global grid systems

DGGS: Transformation

- Creating a similar topology on the corresponding spherical or ellipsoidal surface.

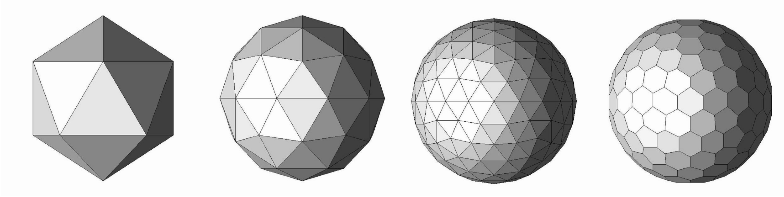


Sphere partitioning



Polyhedral partitioning

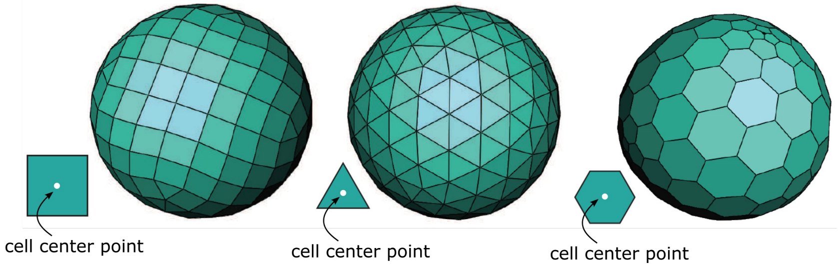
Types of projections:
- Snyder
- Song
- Fuller/ Gray
- ZOT



Source: Comparing area and shape distortion on polyhedral-based recursive partitions of the sphere, Geodesic Discrete Global Grid Systems

DGGS: Assigning points to grid cells

- Usually centroids of the cell region
- Can be specified as - vertices of the triangle, vertices of the dual of the triangles
- Next step: Arrange the prognostic variables on the grid cells (assign data)



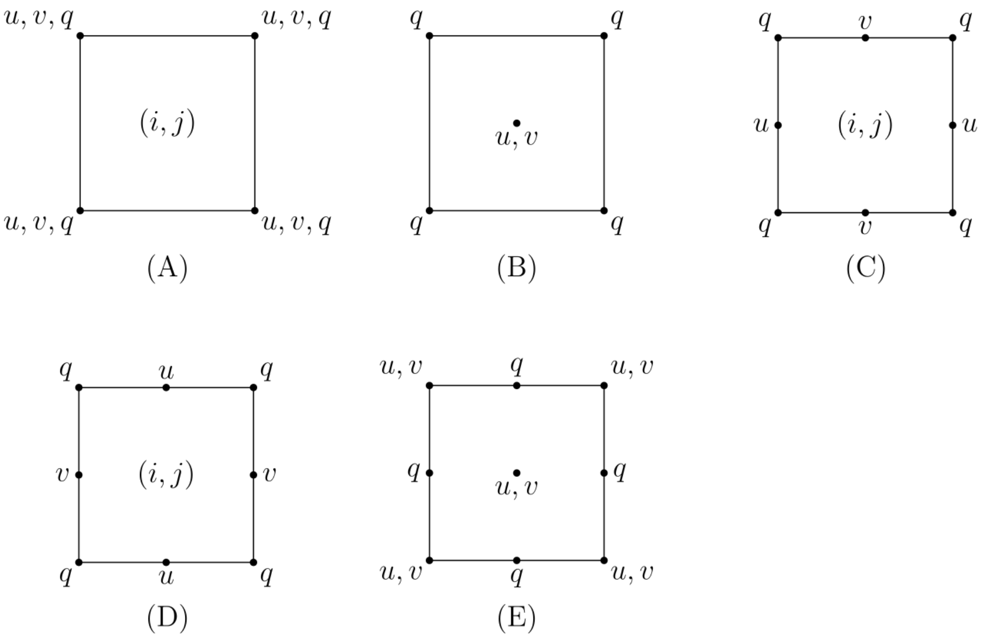
Source: Categorization and conversion for indexing methods of discrete global grid systems

Grid Staggering

- When all the prognostic variables are defined at the same point in a grid, it is called an unstaggered grid (A- Grid).
- When prognostic variables are defined at more than one point in a grid, it is called a staggered grid.
    - B-Grid
    - C-Grid
    - D-Grid
    - E-Grid

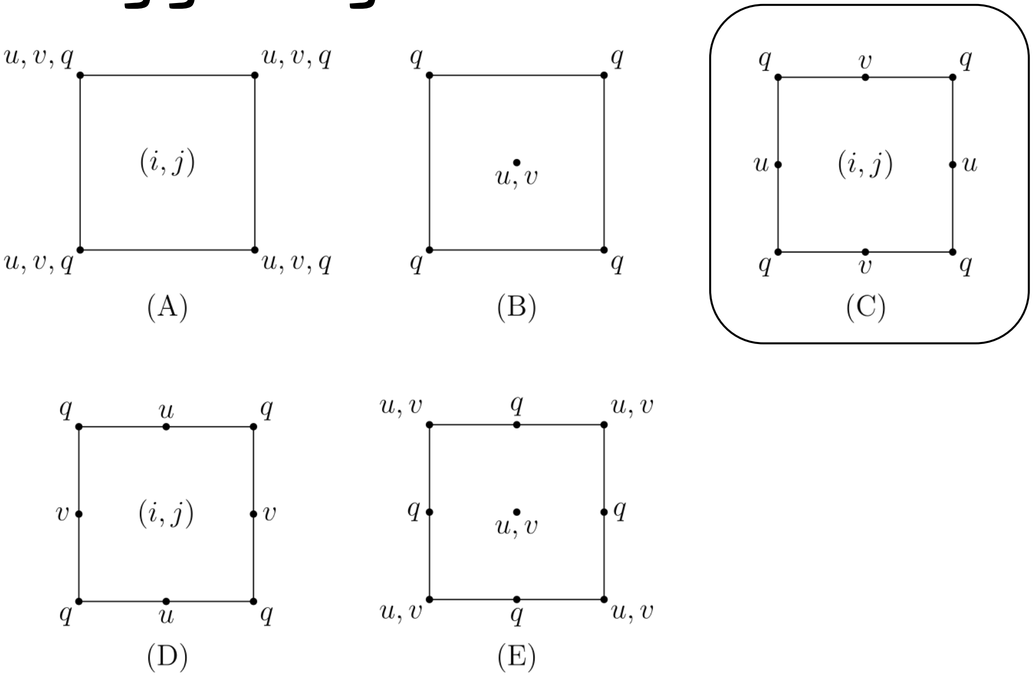
Source: Grids in Numerical Weather and Climate Models

Grid Staggering



Source: Grids in Numerical Weather and Climate Models, Wikipedia

Grid Staggering

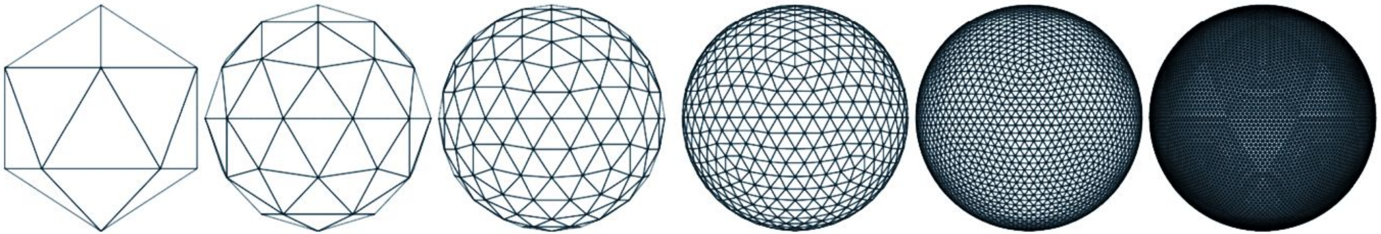


- C-Grid is popular in Weather Research and Forecasting Model
- e.g. ICON (Icosahedral non-hydrostatic)

Source: Grids in Numerical Weather and Climate Models, Wikipedia

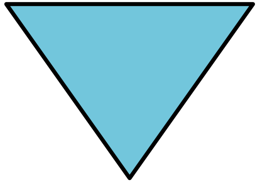
ICON

- Joint project of MPI-DWD
- Used for NWP and climate research
- Icosahedral grid with C-grid staggering

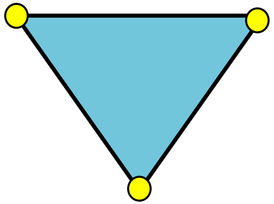


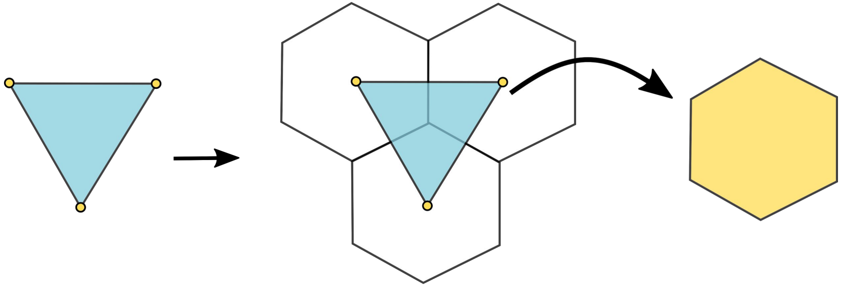
Source: The Non-hydrostatic Icosahedral Atmospheric Model: description and development

ICON

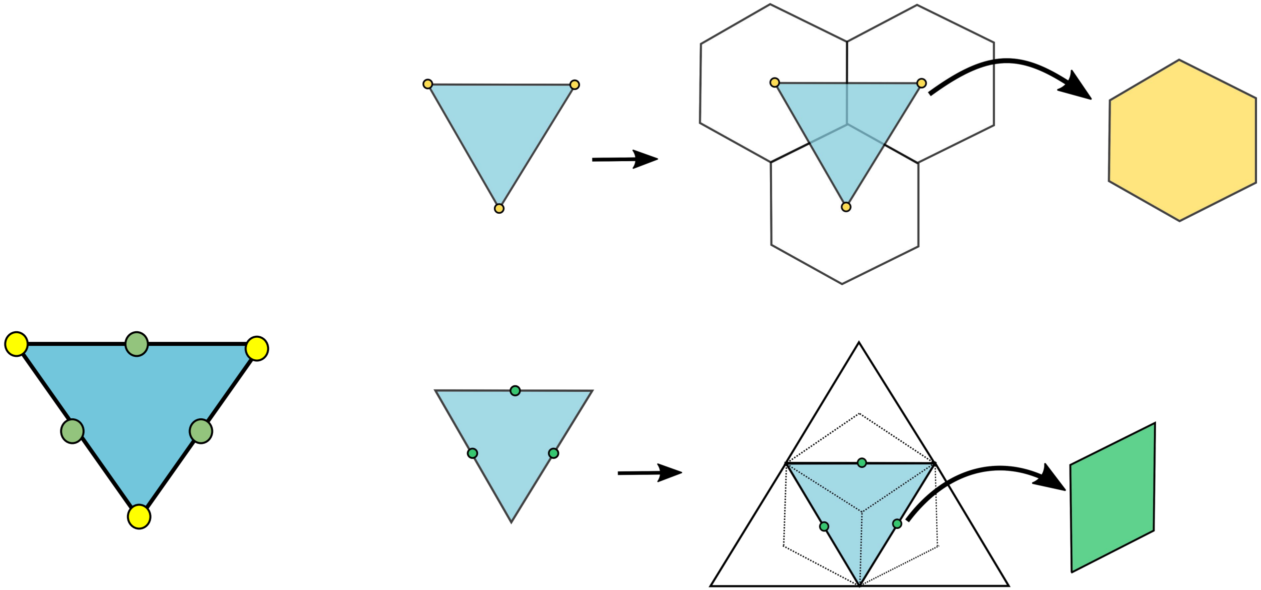


ICON

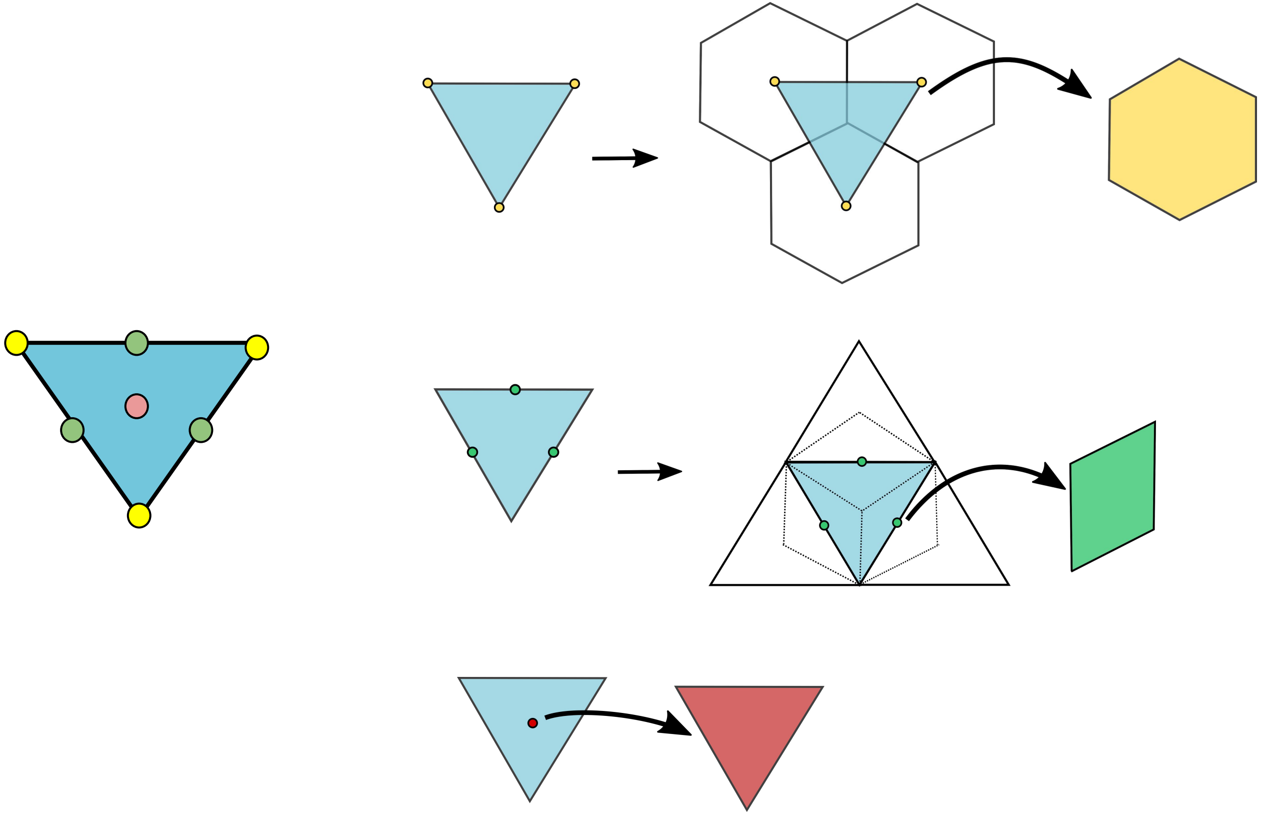




ICON

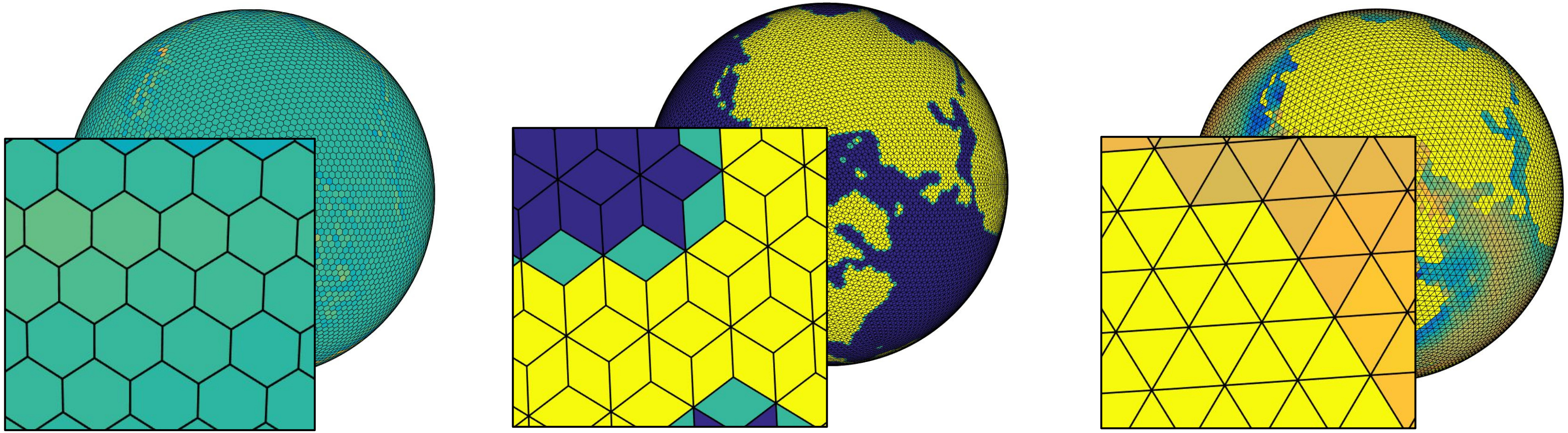


ICON



ICON

Hexagonal cells, Quadrilateral cells and Triangular cells



Research Goal

Research Goal

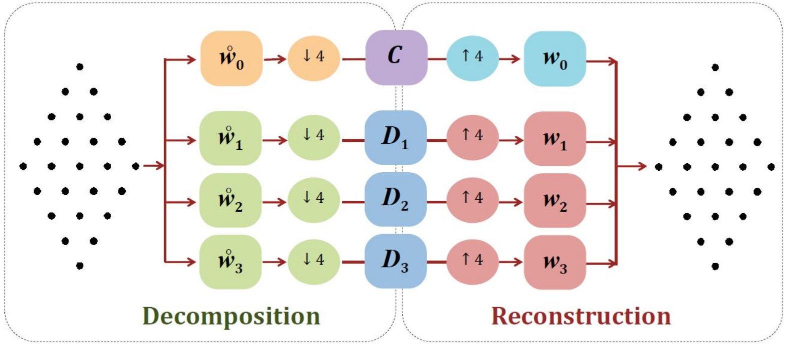
- Wavelets on Digital Earth data for multiresolution visualization
- Must work for all types of cellular data: Center of the hexagons, Center of the quads and Center of the triangles.
    - Need a common data structure
- Application of wavelets:
    - Apply Compression
    - Observing its performance

Source: Geodesic Discrete Global Grid Systems

Methodology

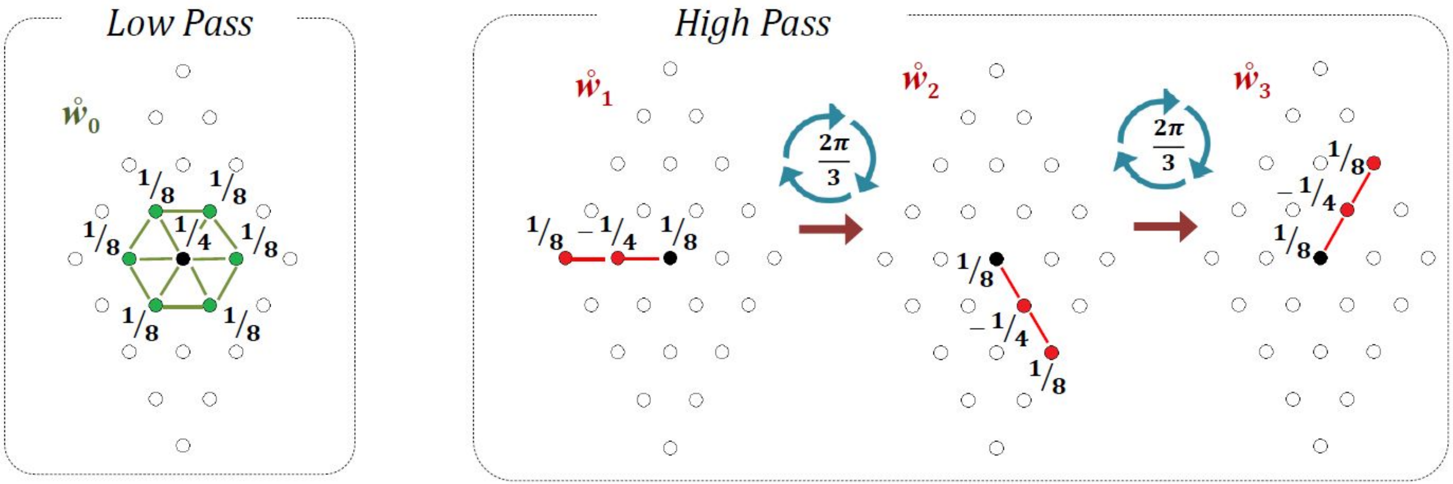
Multiresolution Scheme

- Based on the work of *Cohen and Schlenker*
- Works for triangular grid
- Perfect reconstruction



Source: Geodesic Discrete Global Grid Systems

Multiresolution Scheme

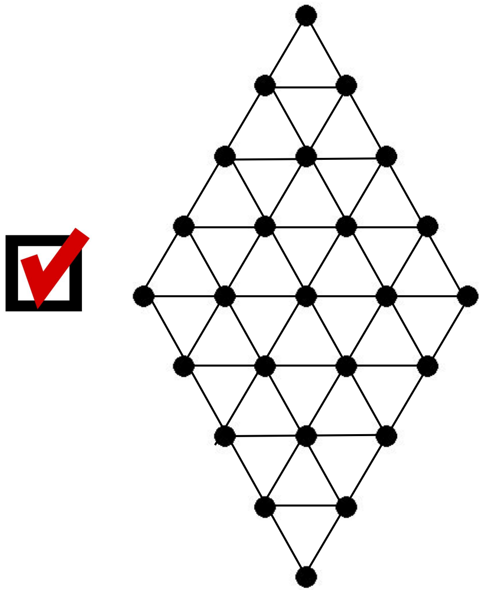


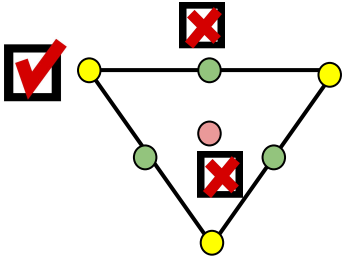
$$C = (F * \mathring{w}_0)_{\downarrow 4}\,,$$

$$D_i = (F * \mathring{w}_i)_{\downarrow 4}\,, \quad \text{where } i = 1,2,3.$$

$$F = (C)_{\uparrow 4} * w_0 + \sum_{i=1}^{3}((D_i)_{\uparrow 4} * \mathring{w}_i)$$

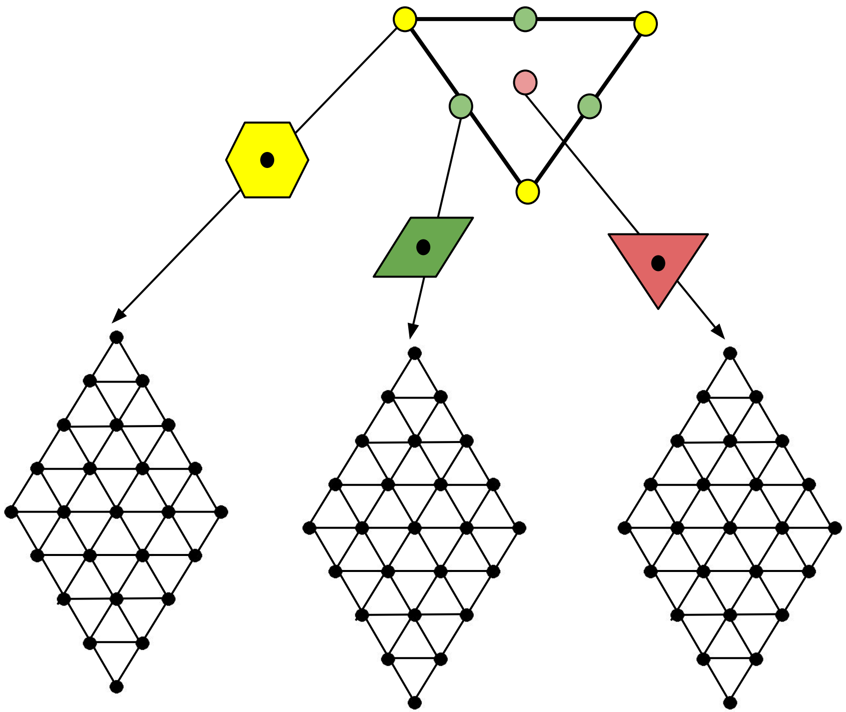
Multiresolution Scheme



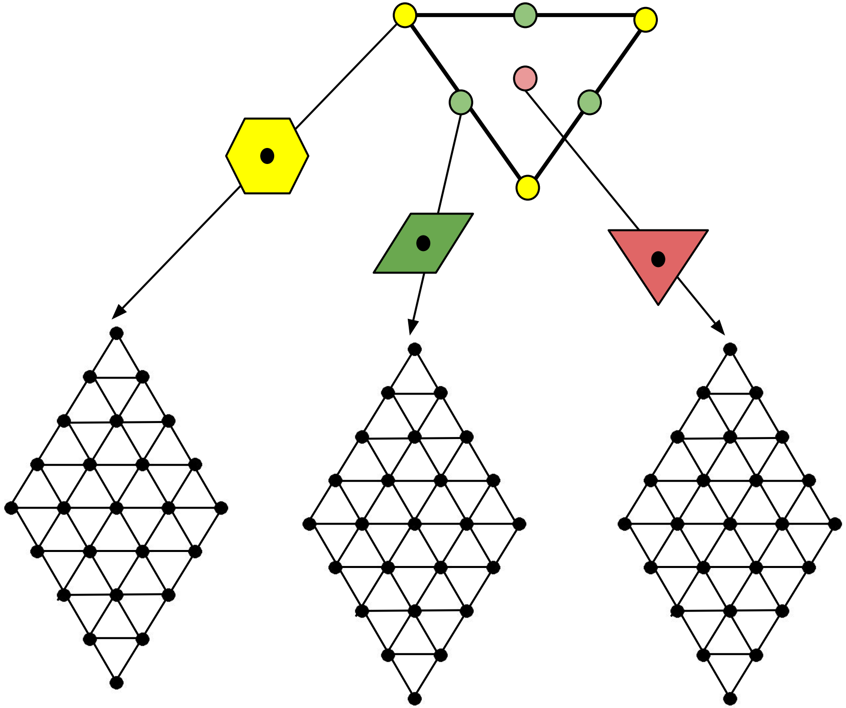


$$C = (F * \mathring{w}_0)_{\downarrow 4} \,,$$
$$D_i = (F * \mathring{w}_i)_{\downarrow 4} \,, \quad \text{where } i = 1,2,3.$$
$$F = (C)_{\uparrow 4} * w_0 + \textstyle\sum_{i=1}^{3}((D_i)_{\uparrow 4} * \mathring{w}_i)$$

Conversion



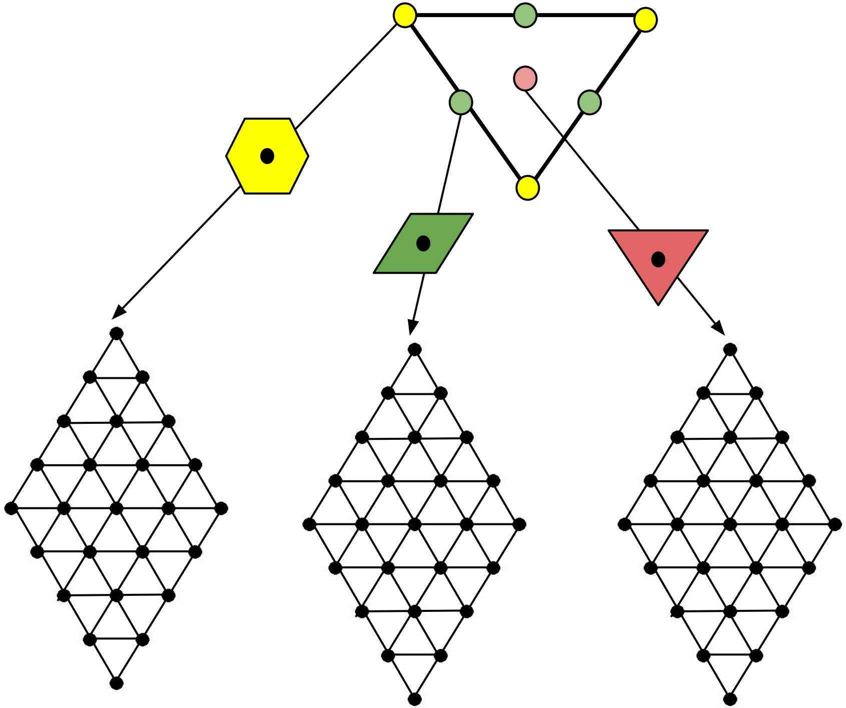
Conversion

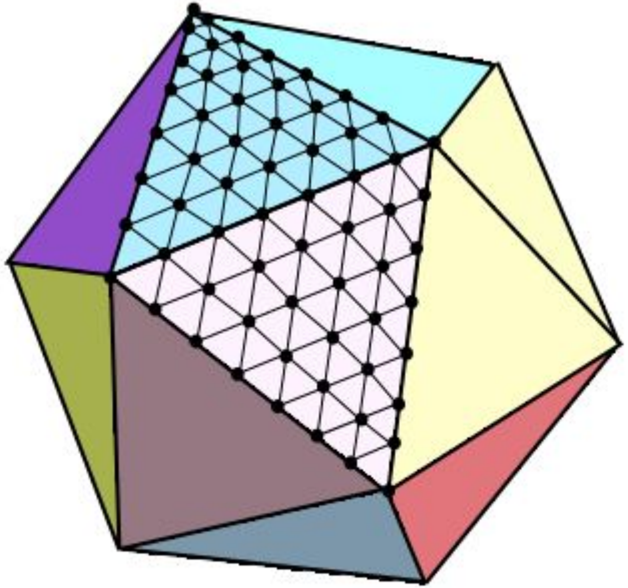


Benefits:

- Highpass and lowpass filters remains consistent.
- Common data structure to handle neighbouring information.
- Triangles are GPU friendly (faces can be rendered using barycentric interpolation)

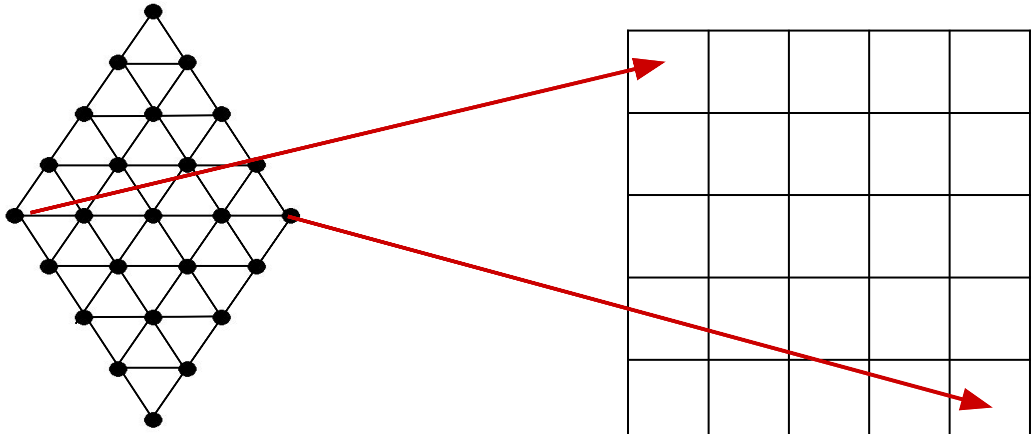
Icosahedral Map





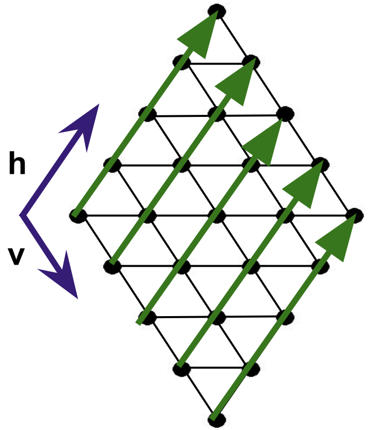
Icosahedral Map

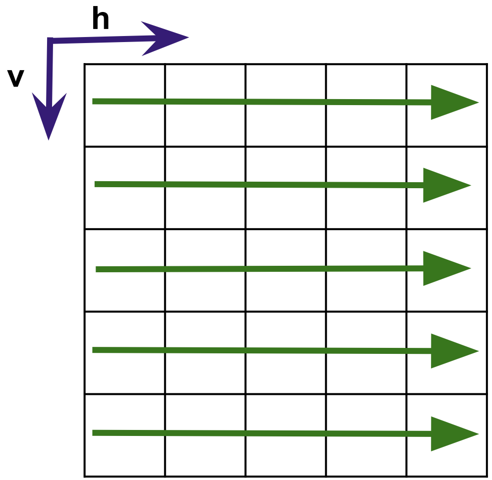
- Digital Earth grid is unstructured
- We need to store is in structured manner (inspired from ACM)
- We want to store the vertex information in 2D array (Connectivity Map)



Icosahedral Map

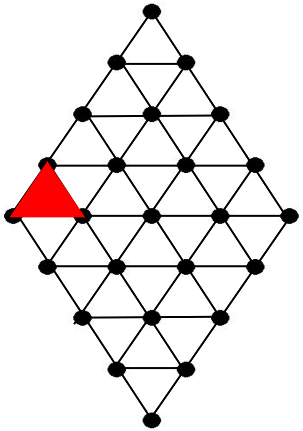
- Digital Earth grid is unstructured
- We need to store is in structured manner (inspired from ACM)
- We want to store the vertex information in 2D array (Connectivity Map)

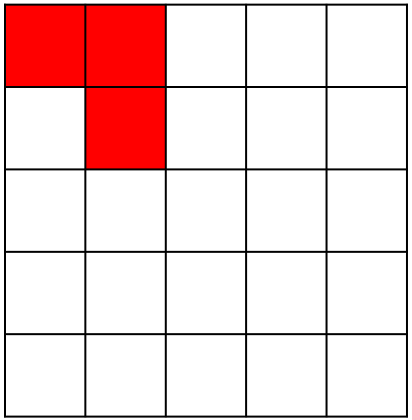




Icosahedral Map

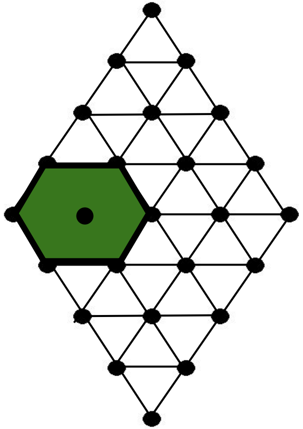
- Digital Earth grid is unstructured
- We need to store is in structured manner (inspired from ACM)
- We want to store the vertex information in 2D array (Connectivity Map)

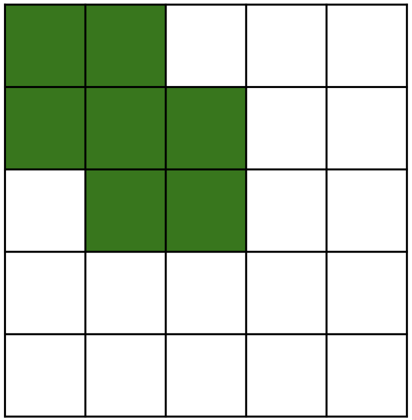




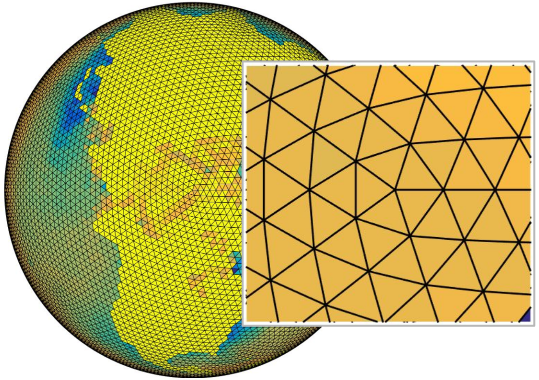
Icosahedral Map

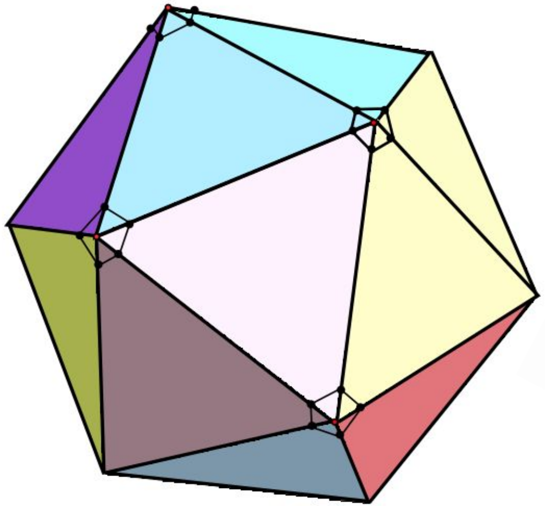
- Digital Earth grid is unstructured
- We need to store is in structured manner (inspired from ACM)
- We want to store the vertex information in 2D array (Connectivity Map)

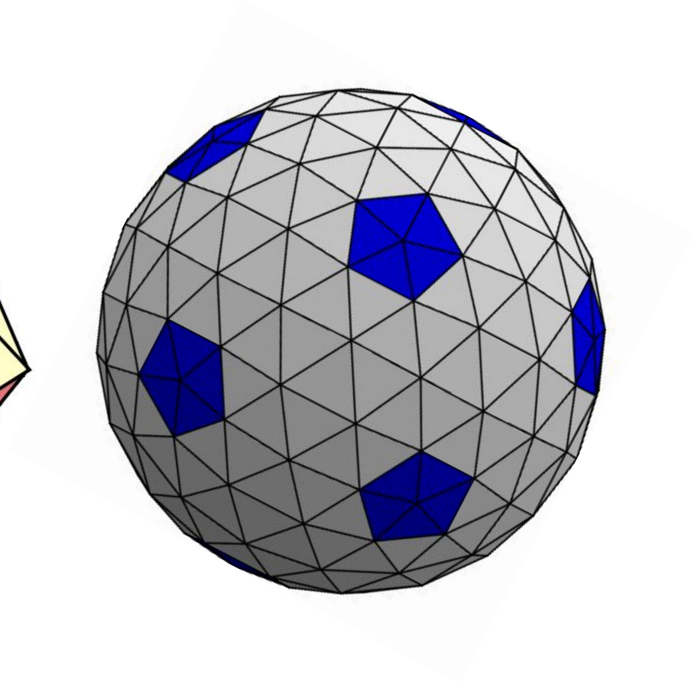




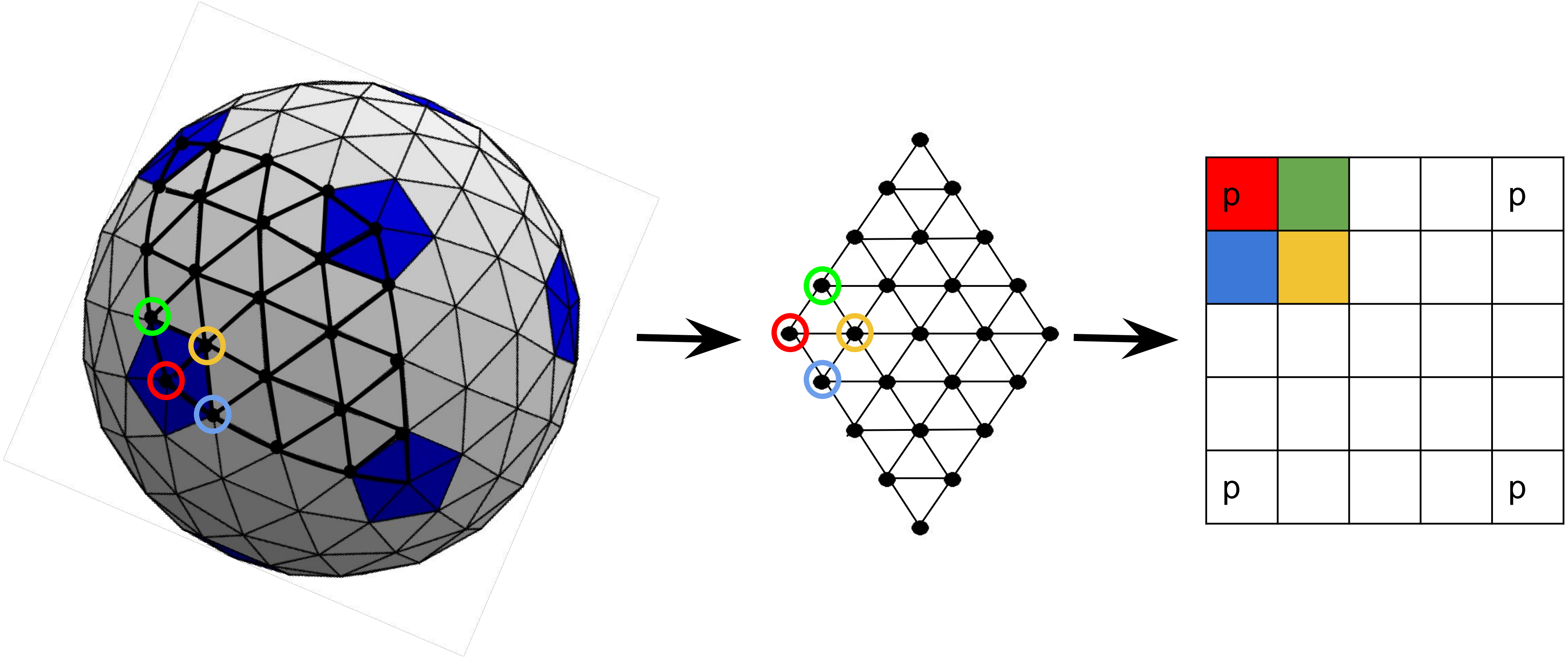
Icosahedral Map





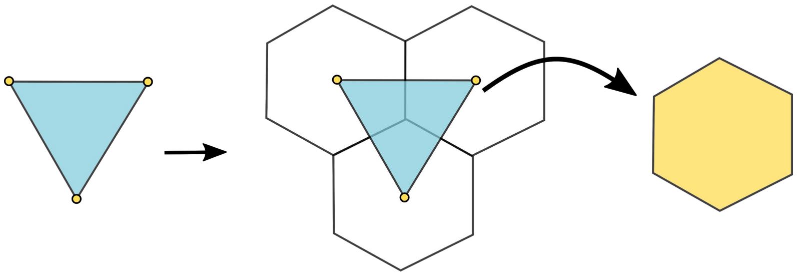


Icosahedral Map

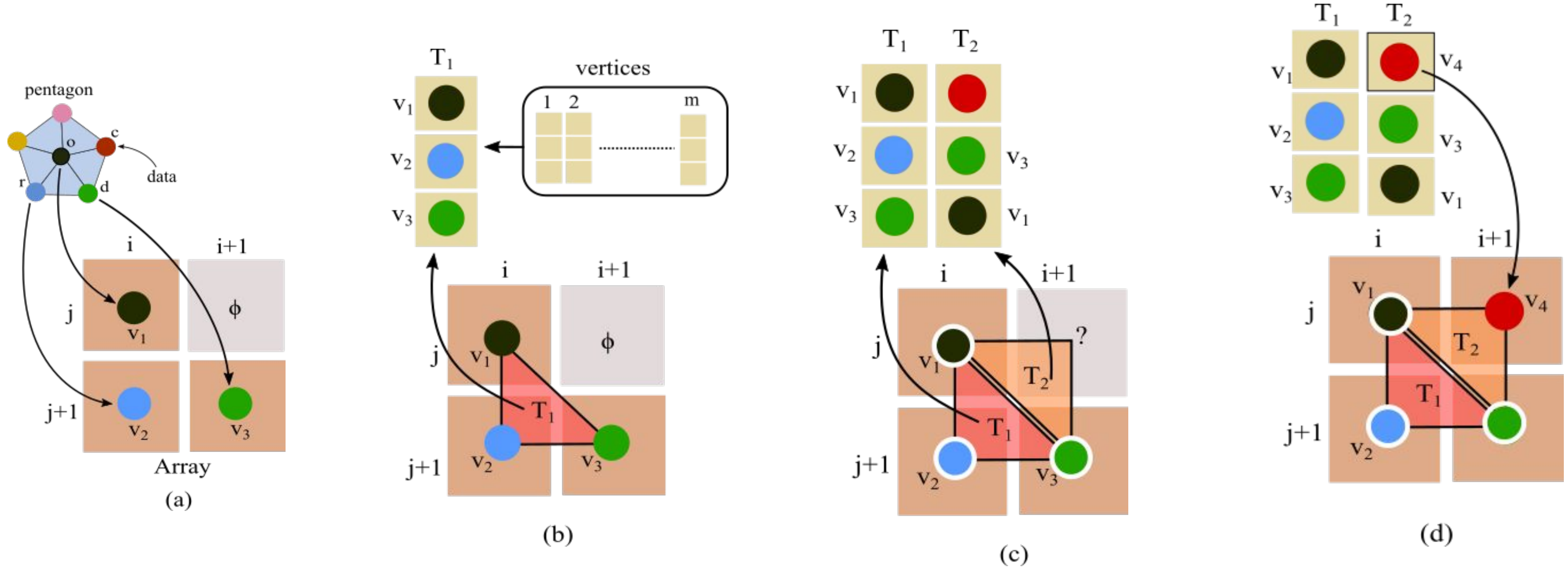


Icosahedral Map: CoH

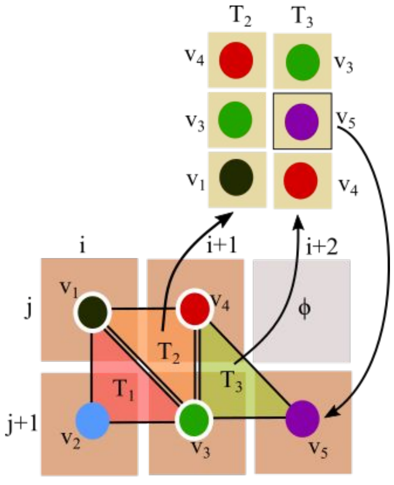
- Center of Hexagons → Vertices of Triangles (Already!)
- Need to find connectivity information and store it in 2D array



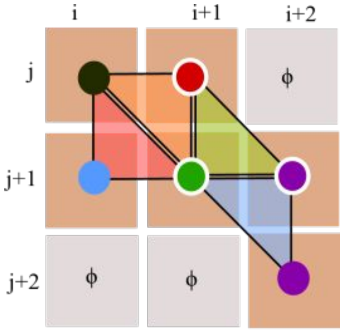
Icosahedral Map: CoH



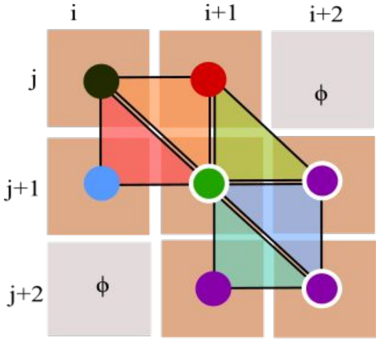
Icosahedral Map: CoH



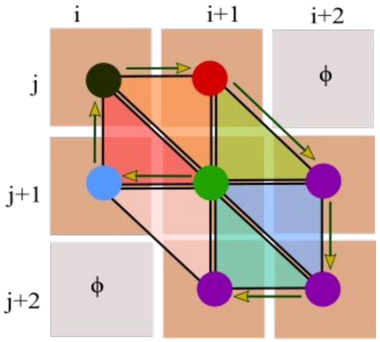
(e)



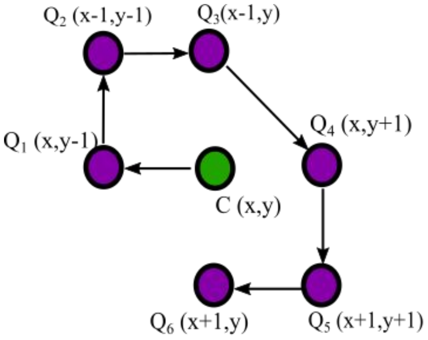
(f)



(g)



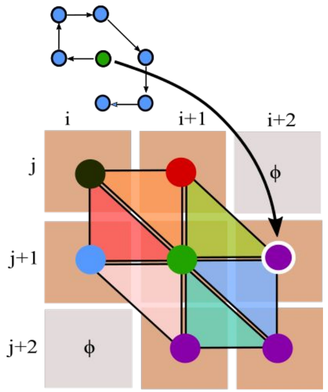
(h)



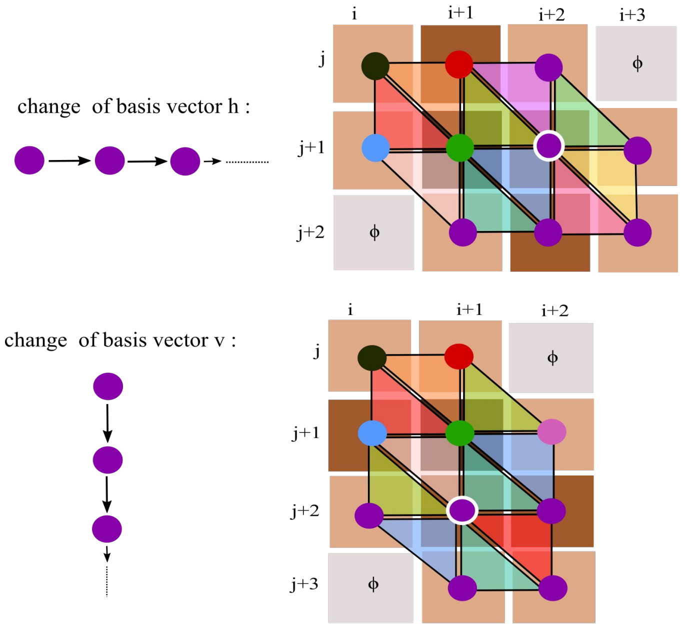
$\text{connectivity}(C, Q_i, Q_{i+1}) \longrightarrow Q_{i+2},\ i = 1, 2, 3, 4$

(i)

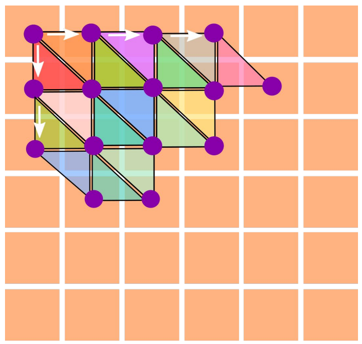
Icosahedral Map: CoH



(a)

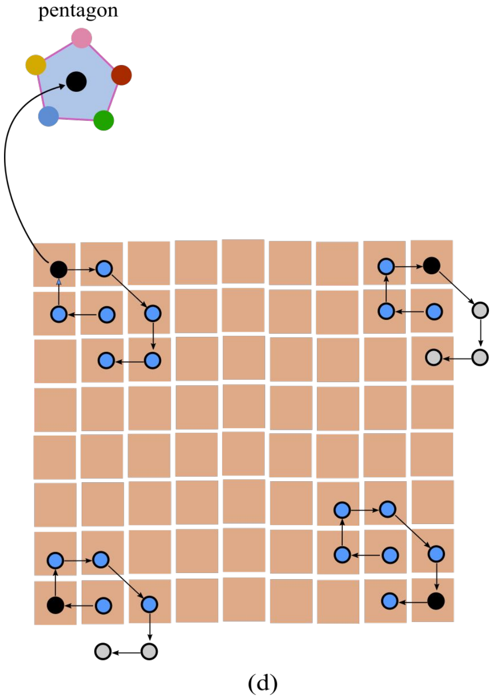


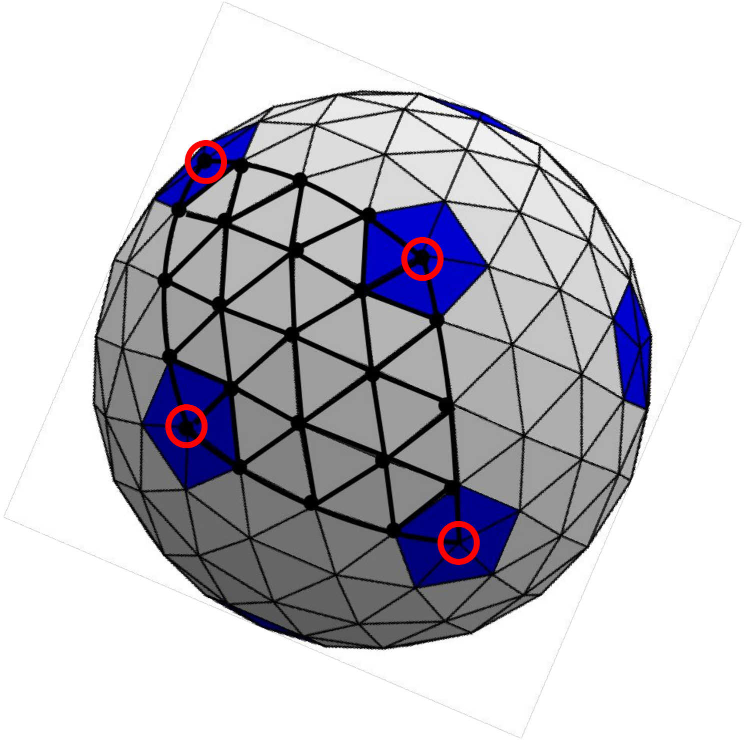
(b)



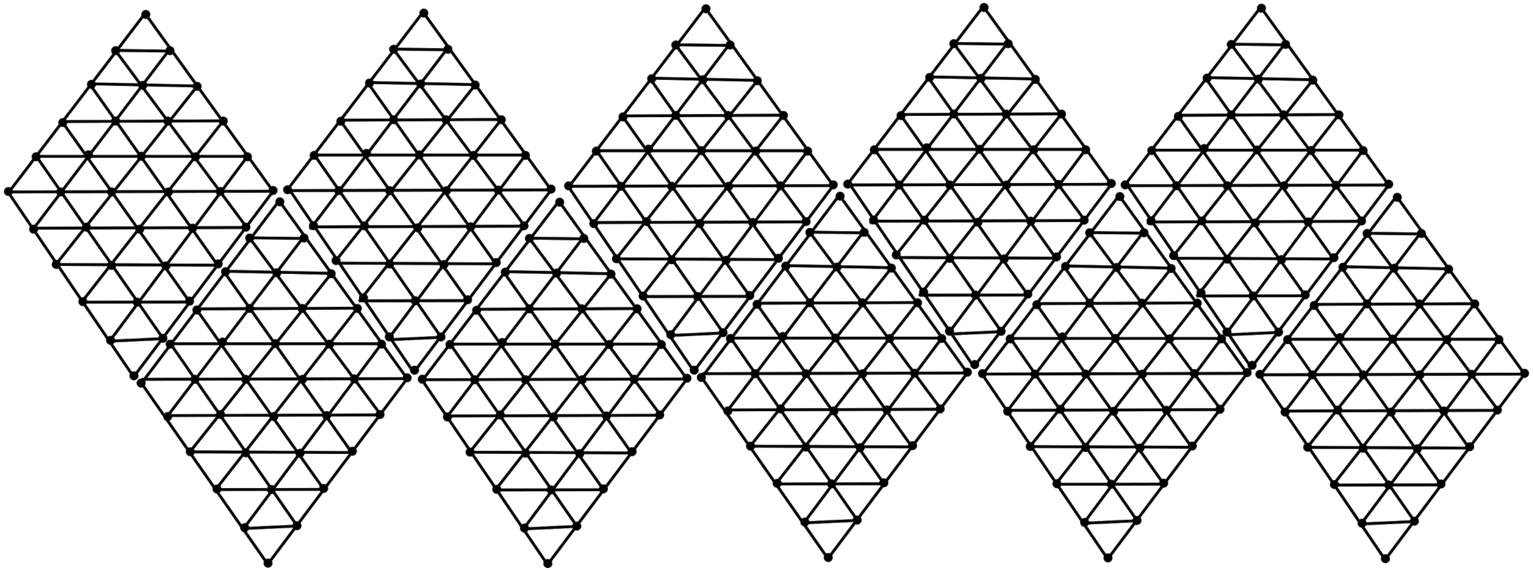
(c)

Icosahedral Map: CoH

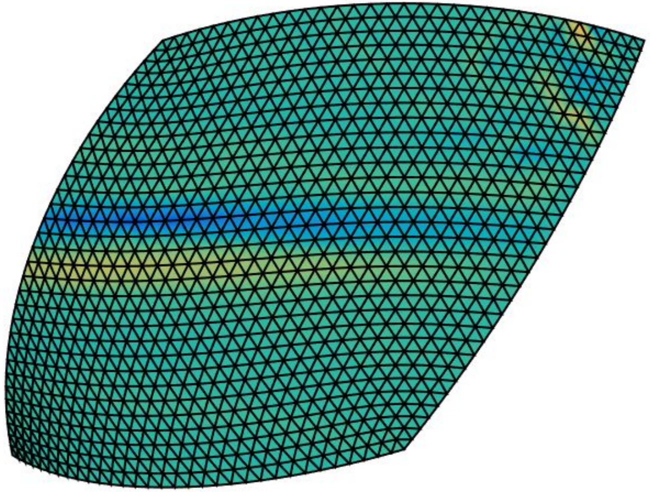


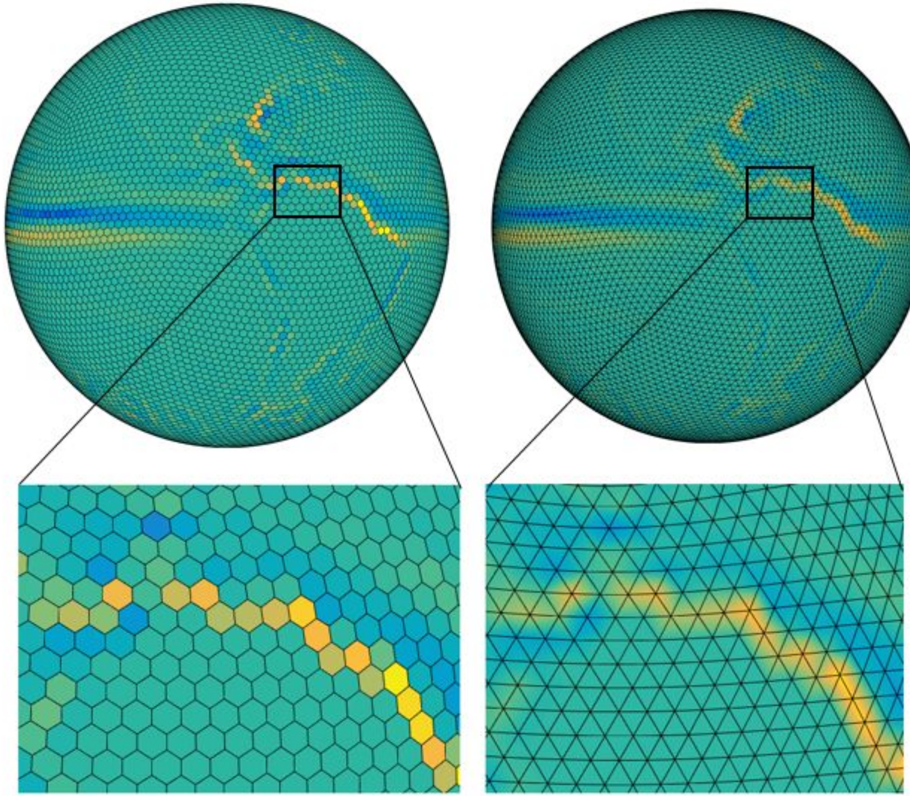


Icosahedral Map

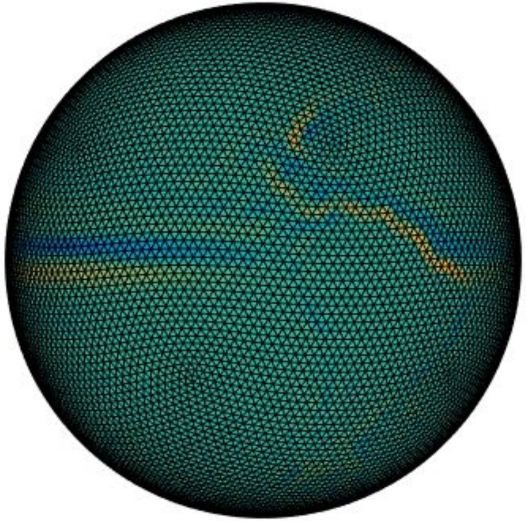
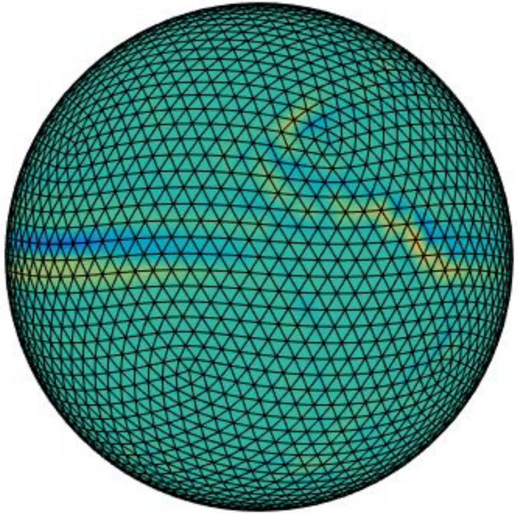
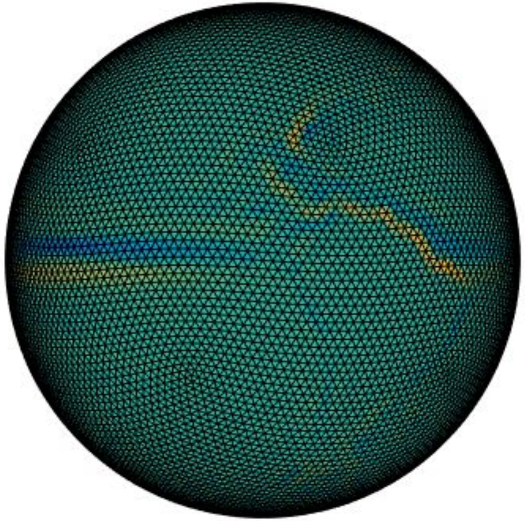


Icosahedral Map: CoH



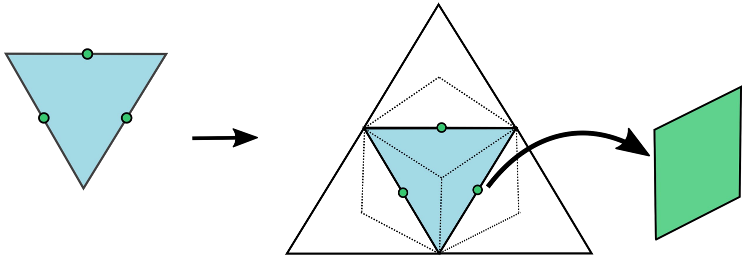


Icosahedral Map: CoH

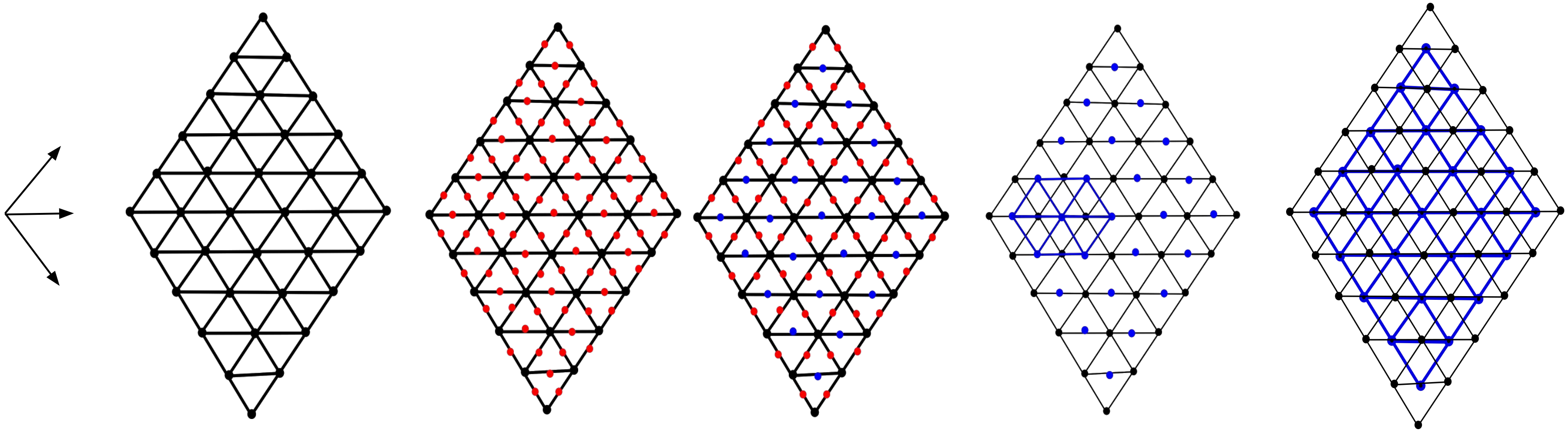
  

Icosahedral Map: CoQ

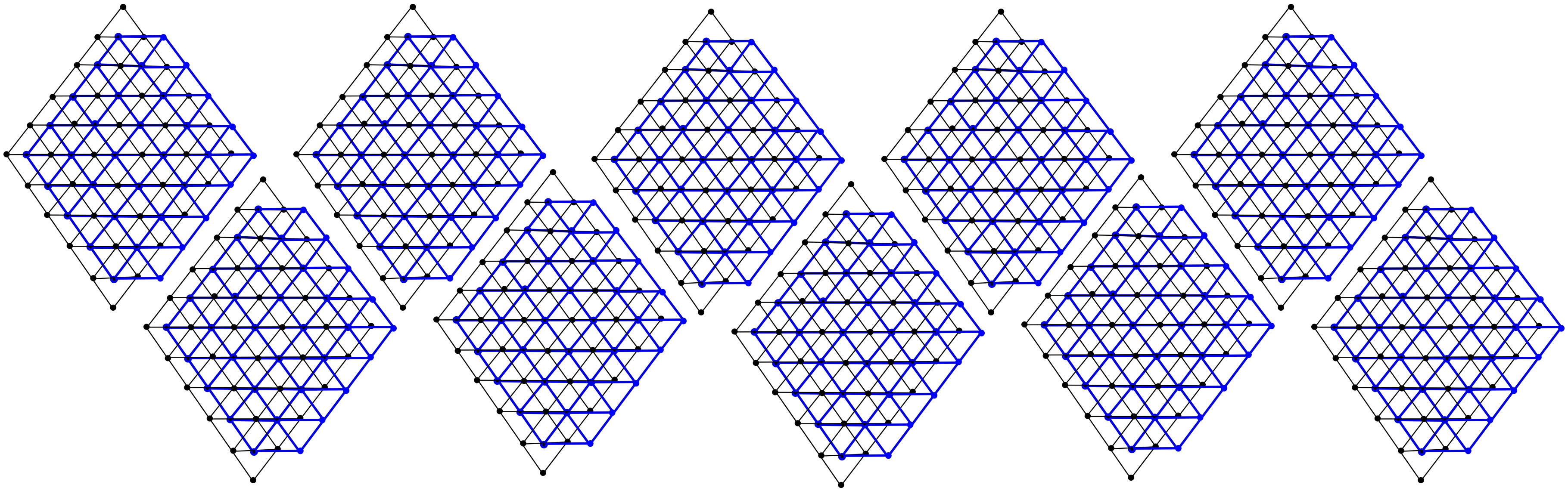
- Center of Quads→ Edge midpoints of Triangles



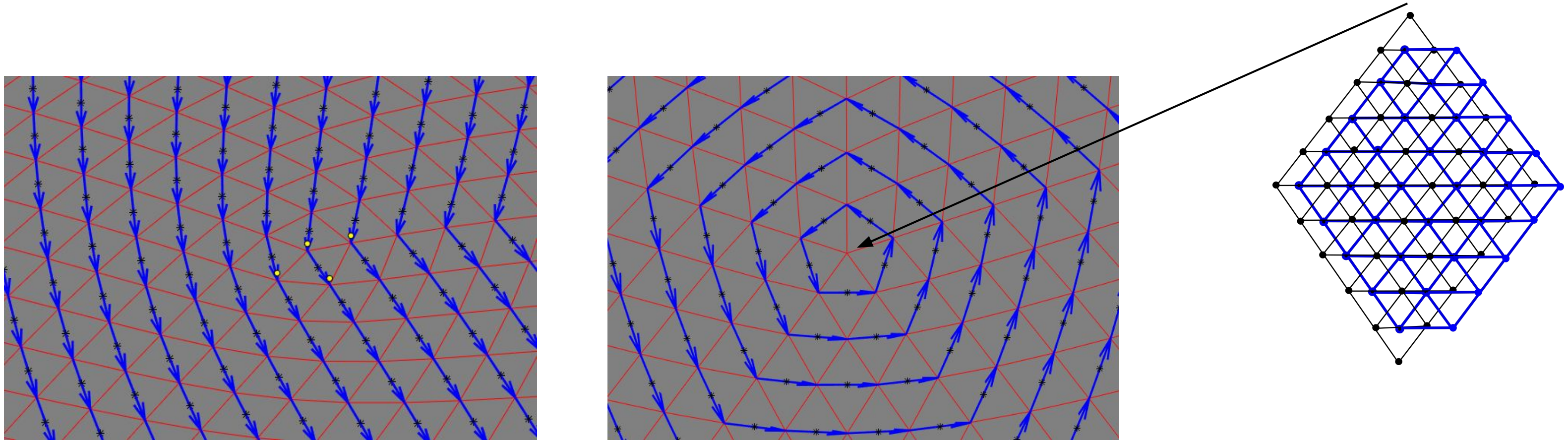
Icosahedral Map: CoQ



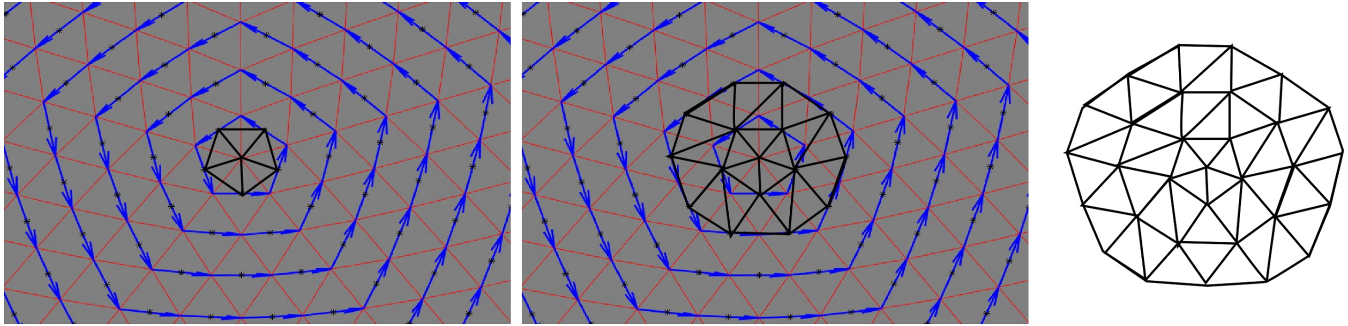
Icosahedral Map: CoQ



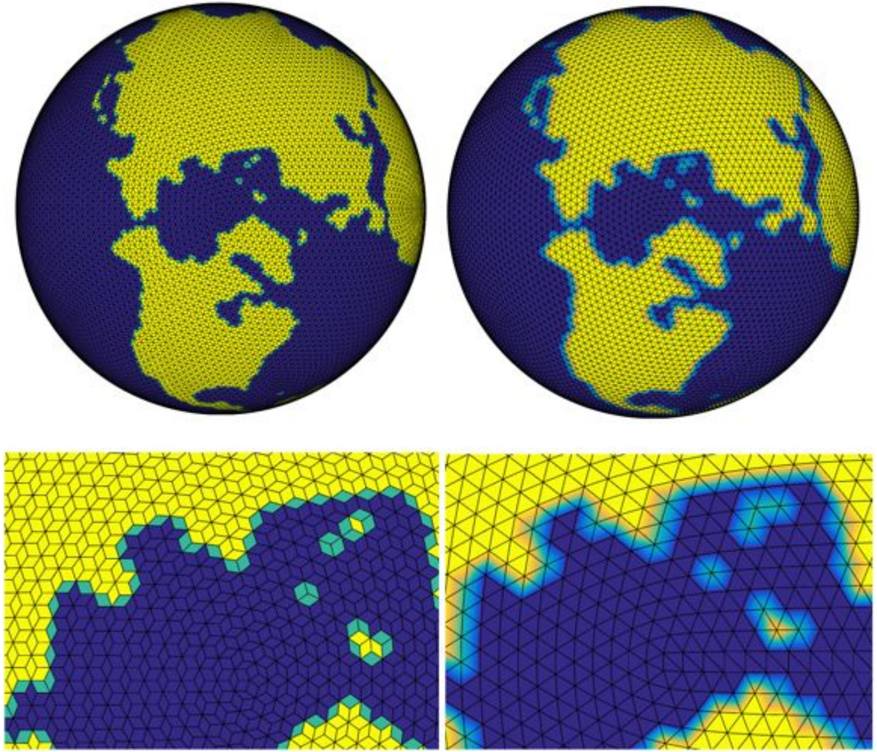
Icosahedral Map: CoQ



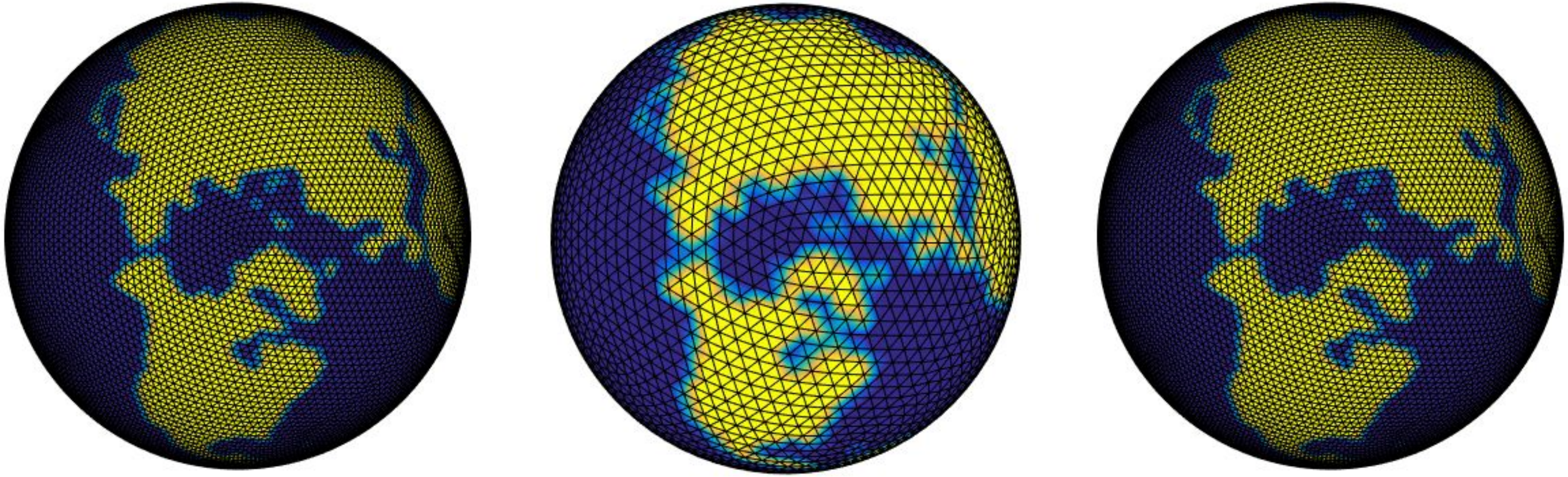
Icosahedral Map: CoQ



Icosahedral Map: CoQ

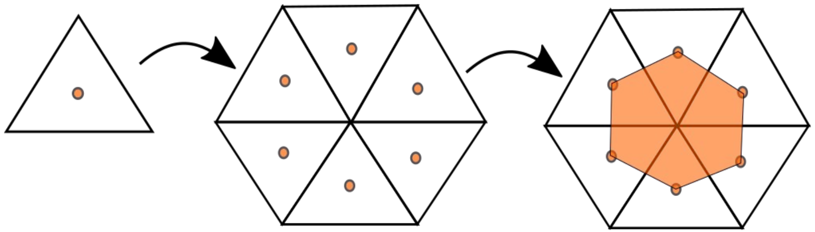


Icosahedral Map: CoQ

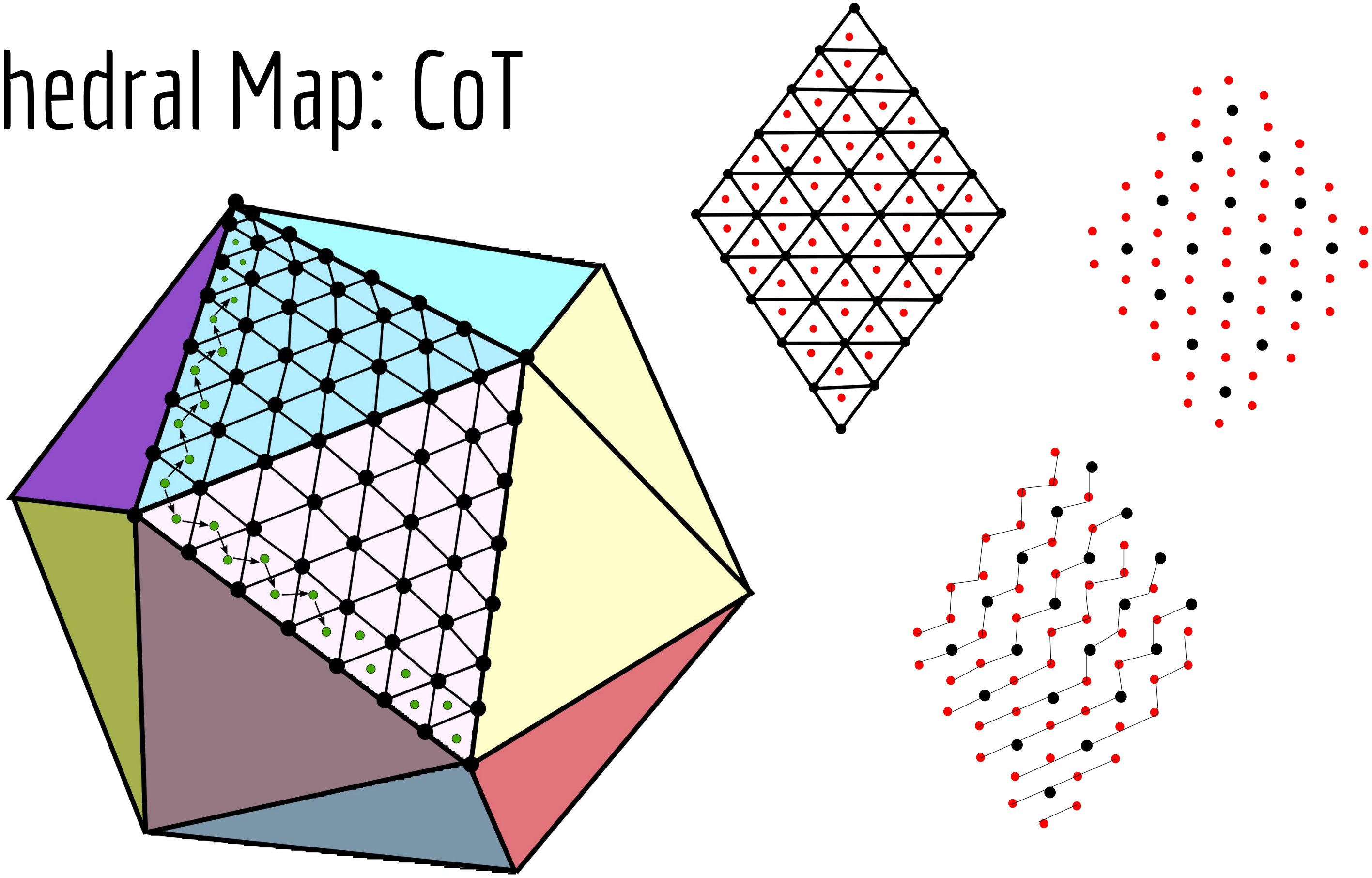


Icosahedral Map: CoT

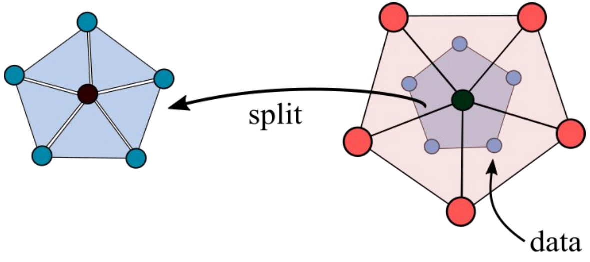
- Center of Triangles→ Vertices of The voronoi cell (hexagon)



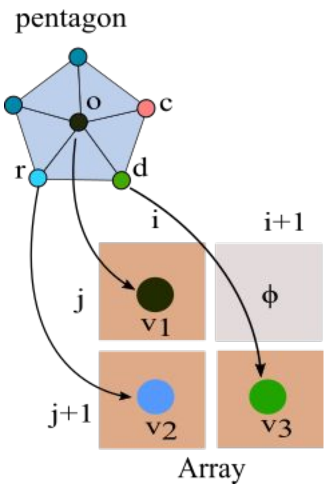
Icosahedral Map: CoT



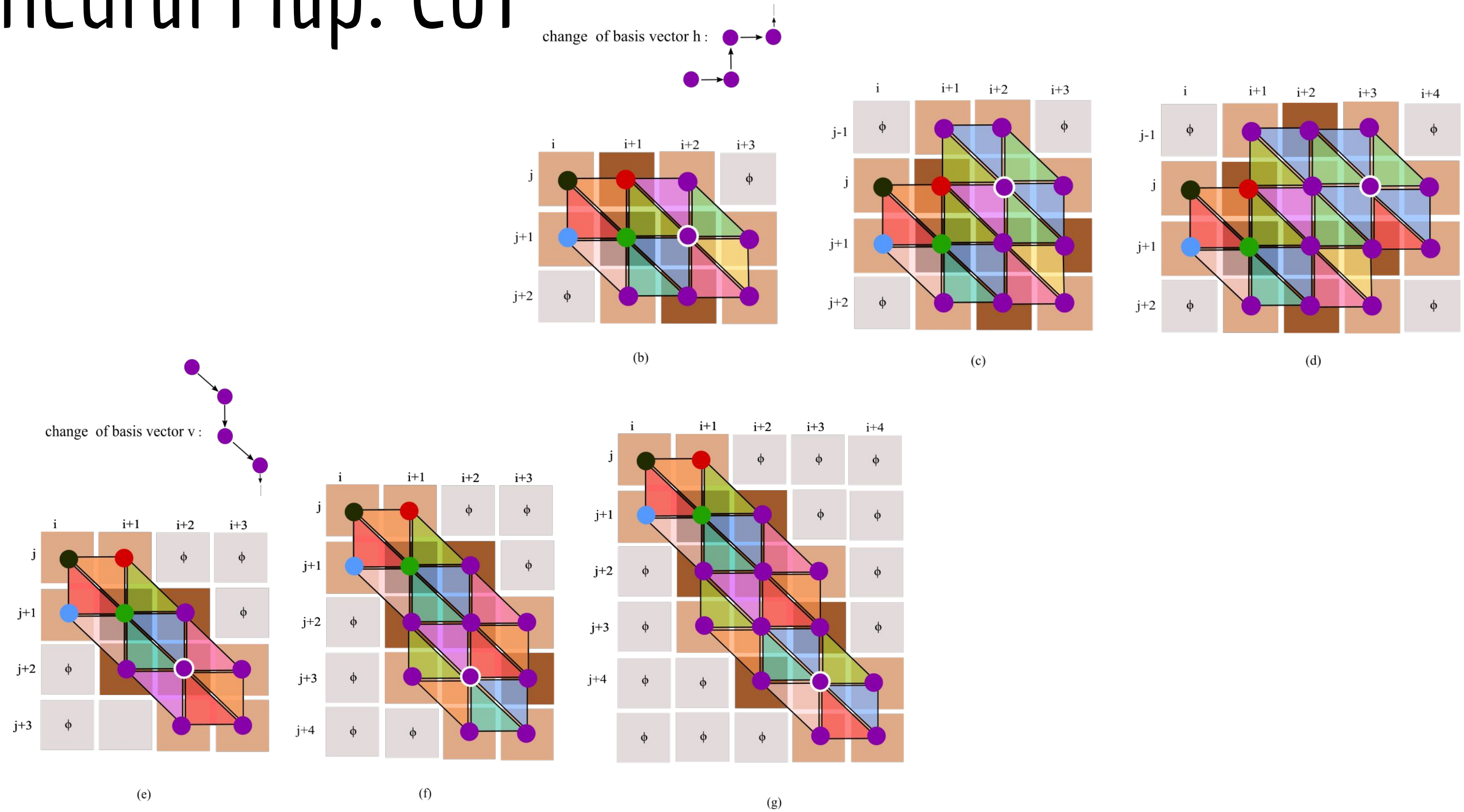
Icosahedral Map: CoT



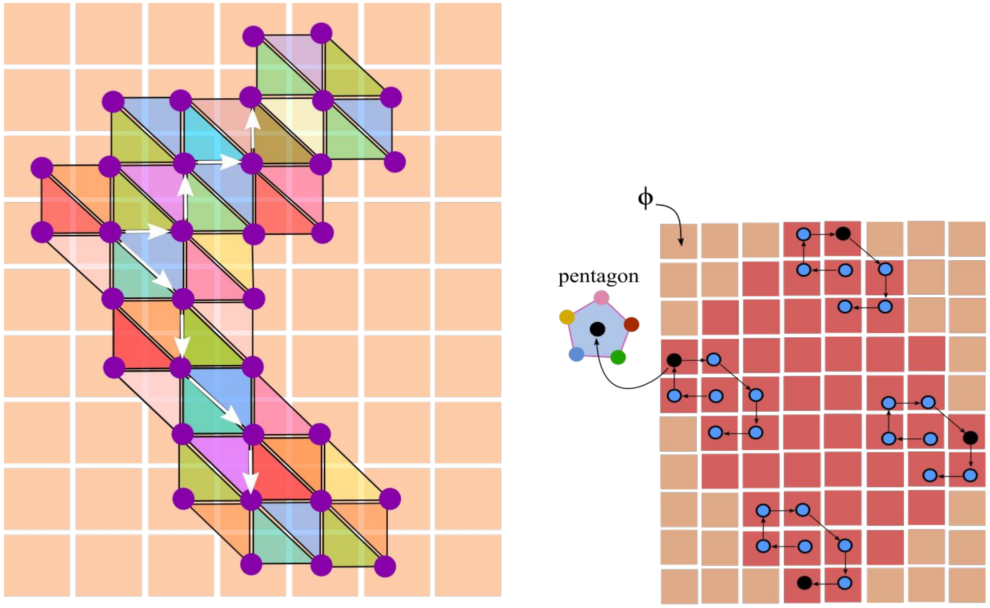
(a)



Icosahedral Map: CoT

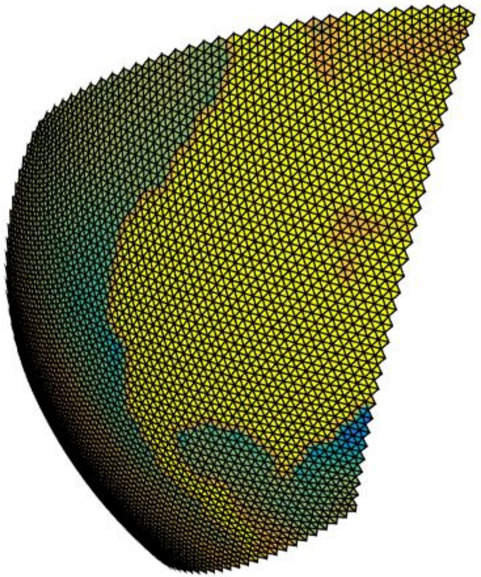


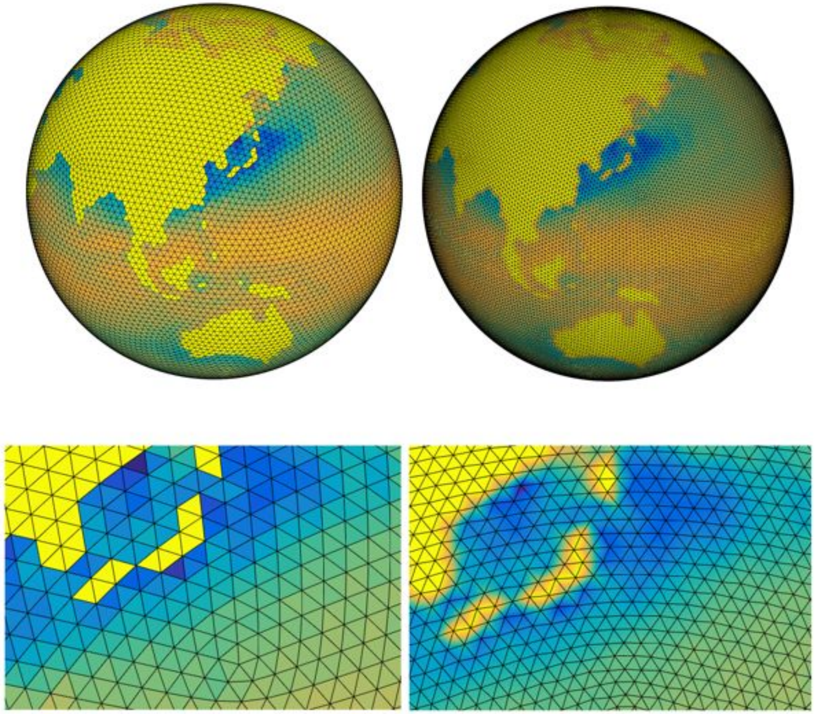
Icosahedral Map: CoT



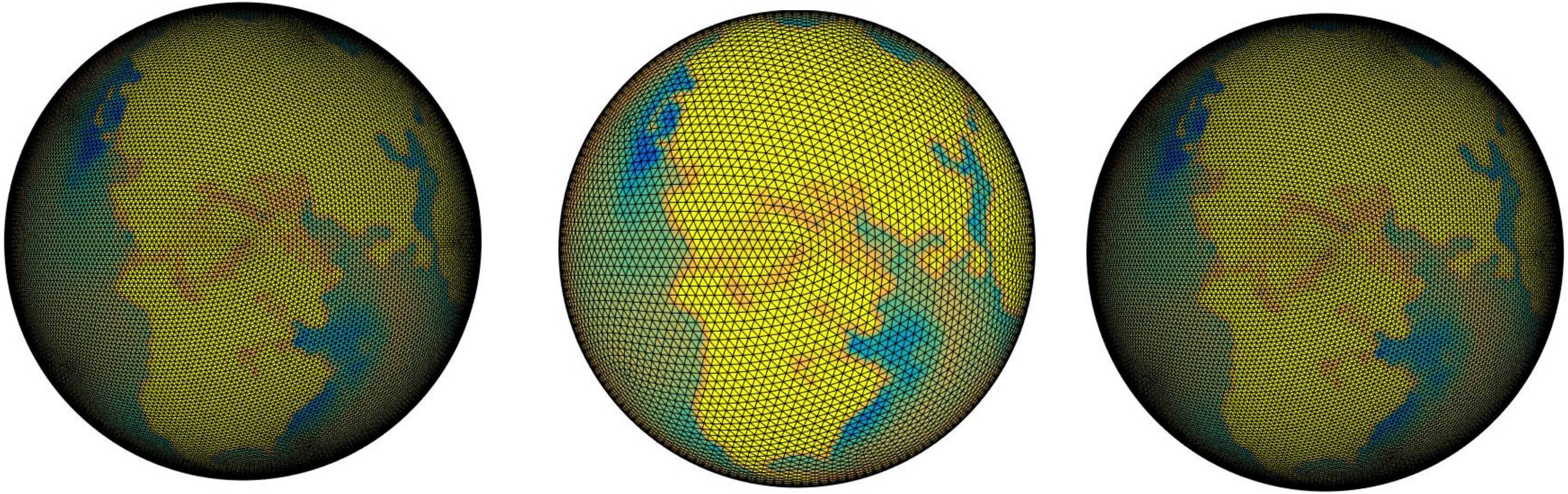
(a) (b)

Icosahedral Map: CoT



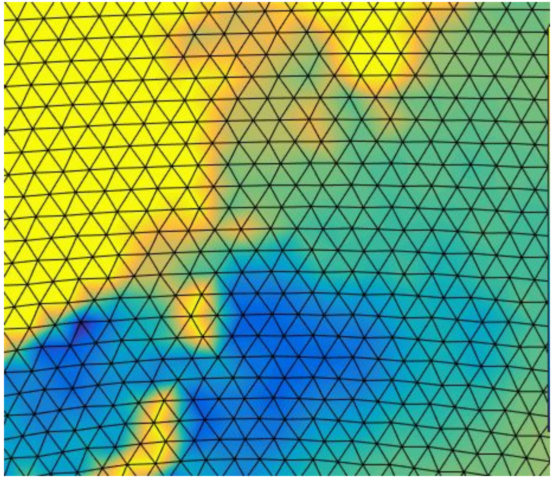


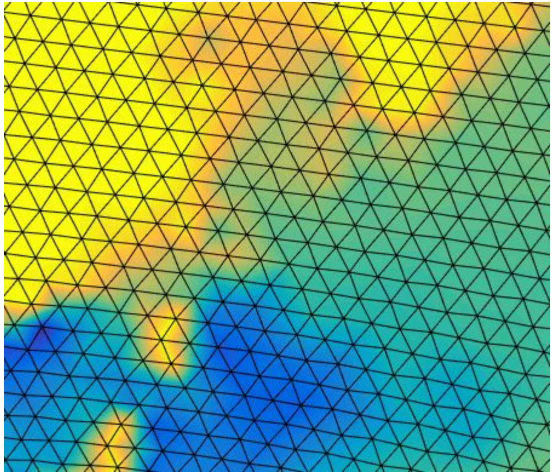
Icosahedral Map: CoT



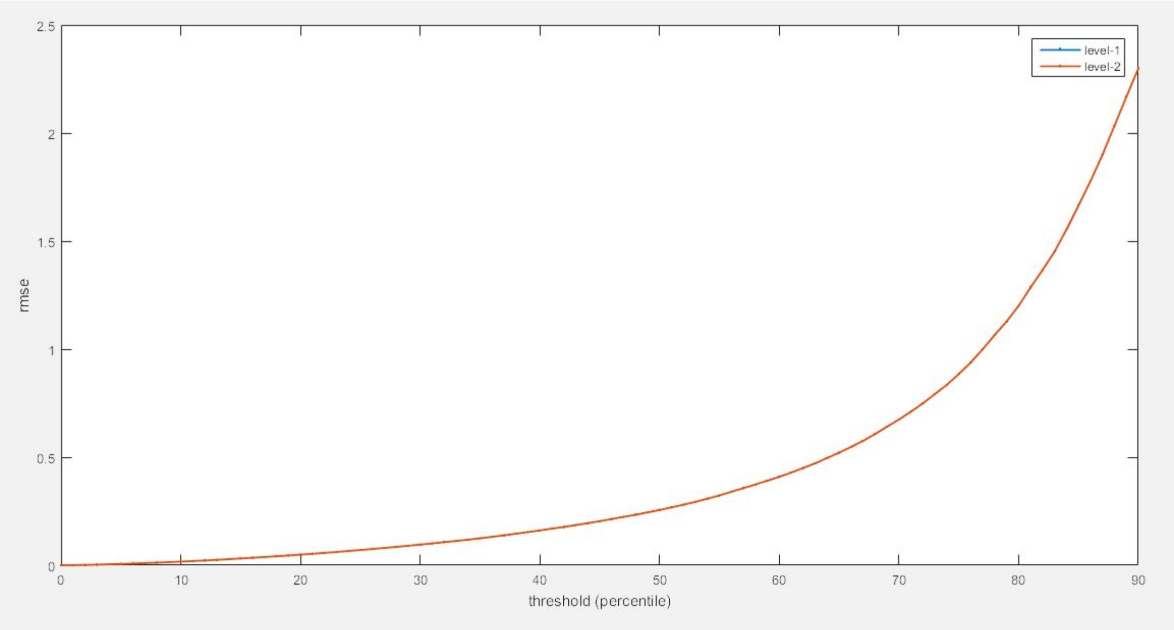
Compression

Zero out magnitudes in details based on threshold





Compression



Future Work

- Reduce Empty spaces in array
- Create a single array that holds the entire polyhedron net
- GPU implementation
- Bricking

Thanks