Department of
Computer Science

CPSC 601 | Project Final Presentation

Implementing Atlas of Connectivity Maps for ICON Grid

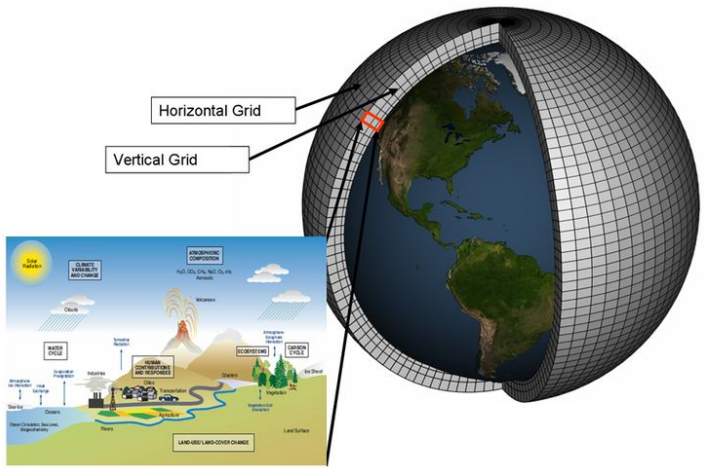
Mohammad Imrul Jubair
mohammadimrul.jubair@ucalgary.ca

Outline

- Computer-based globe model
- Study on ICON Grid
- ICONverter: Implementing Atlas of Connectivity Maps for ICON Grid
- visICON

Computer-based globe model

- Representation of geospatial data on digitized globe system
  - ✓ e.g. ICON globe model
- Data is obtained from various kind of data acquisition process
- Important in Meteorology
  - ✓ e.g. prediction of climate performance for future.



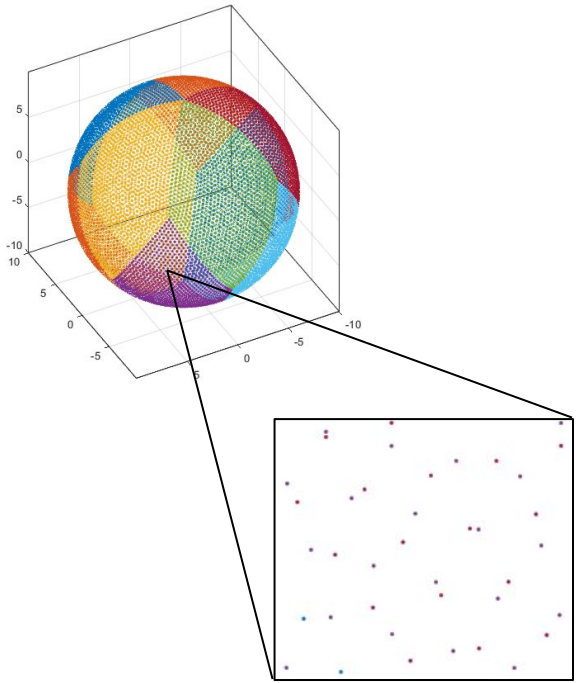
Computer-based globe model *(cont.….)*

- Discretizing Earth's surface into different *geometric entities*:

Computer-based globe model *(cont.….)*

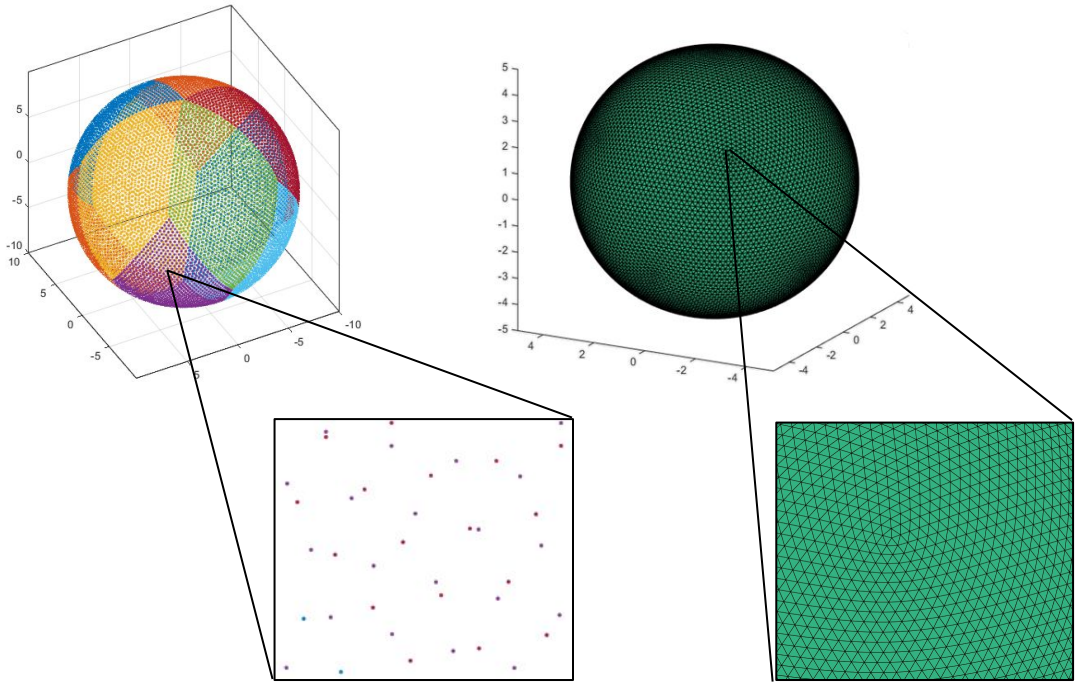
- Discretizing Earth's surface into different *geometric entities*:

  ✓ vertices



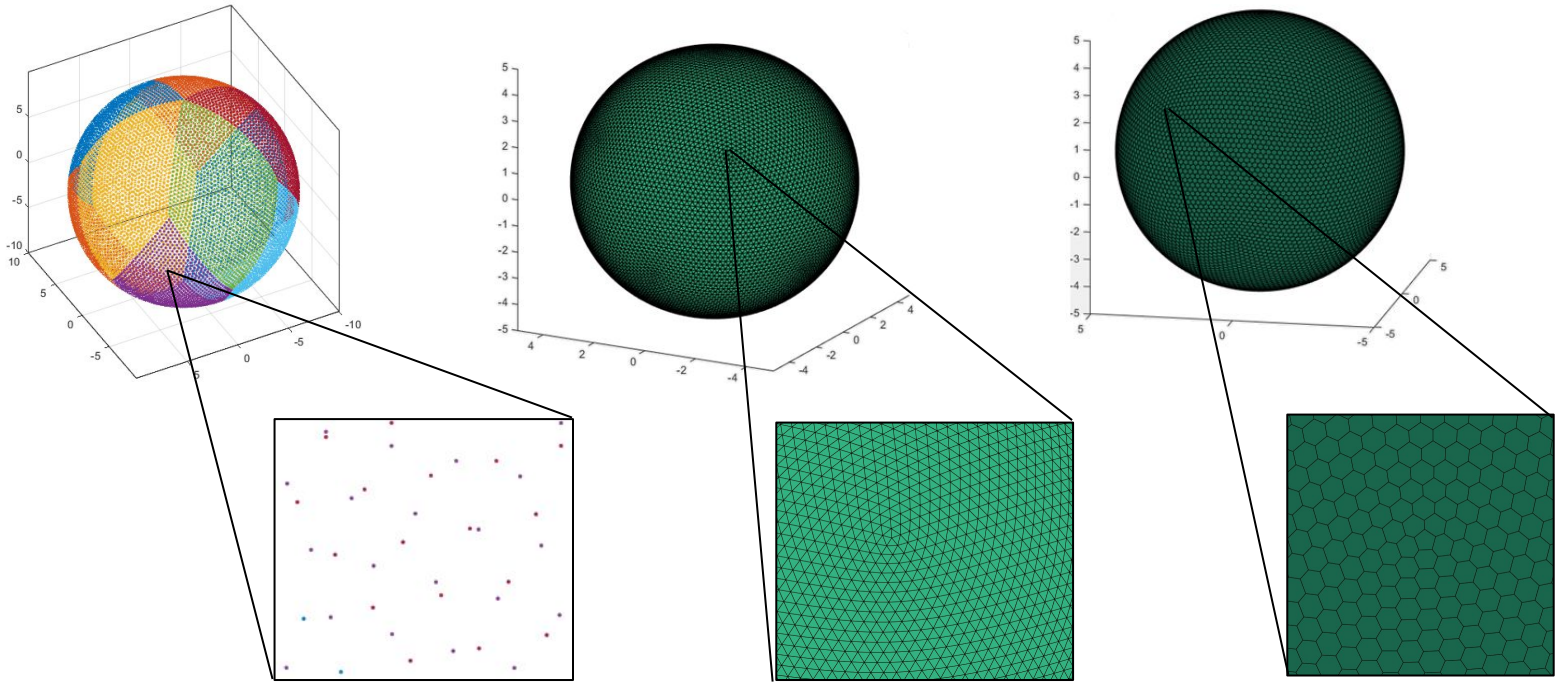
Computer-based globe model *(cont.….)*

- Discretizing Earth's surface into different *geometric entities*:
  - ✓ vertices, triangles



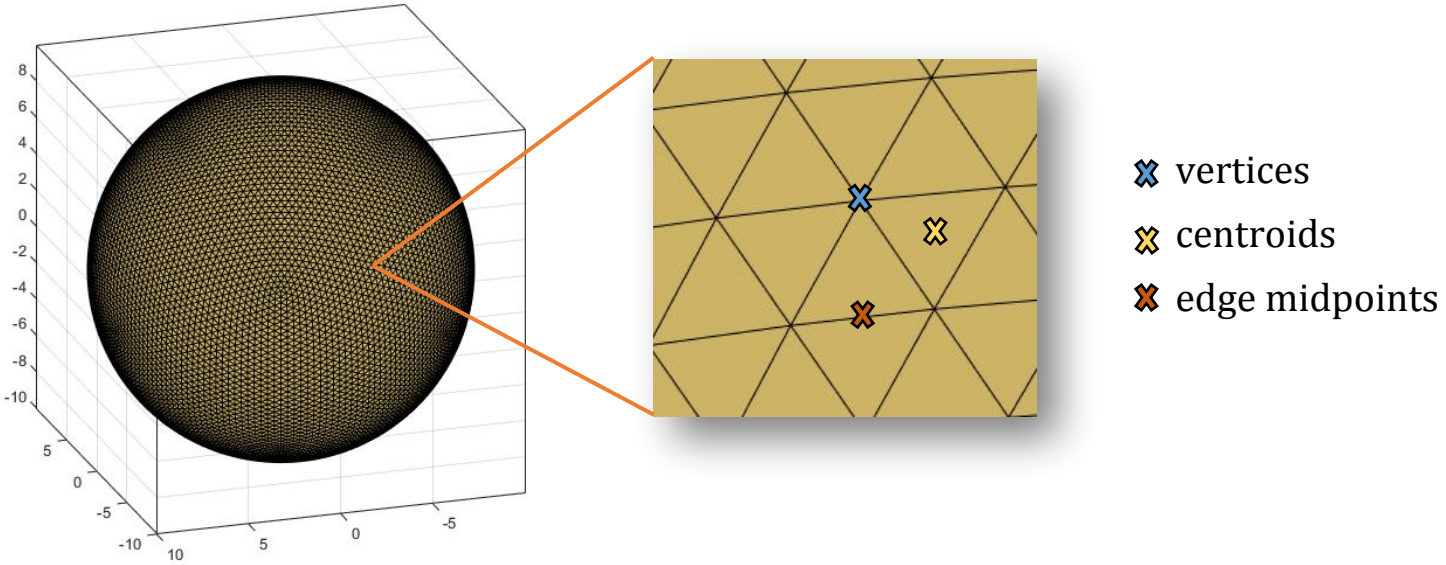
Computer-based globe model *(cont.….)*

- Discretizing Earth's surface into different *geometric entities*:
  - ✓ vertices, triangles, hexagons etc.



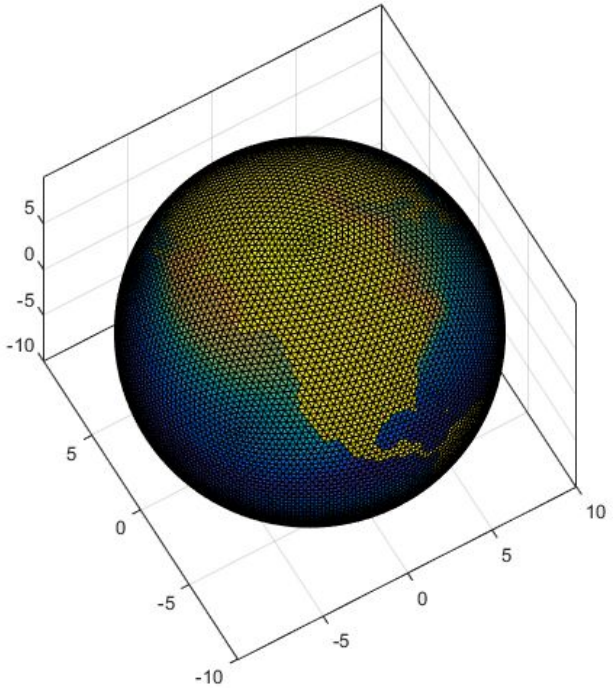
Computer-based globe model *(cont.….)*

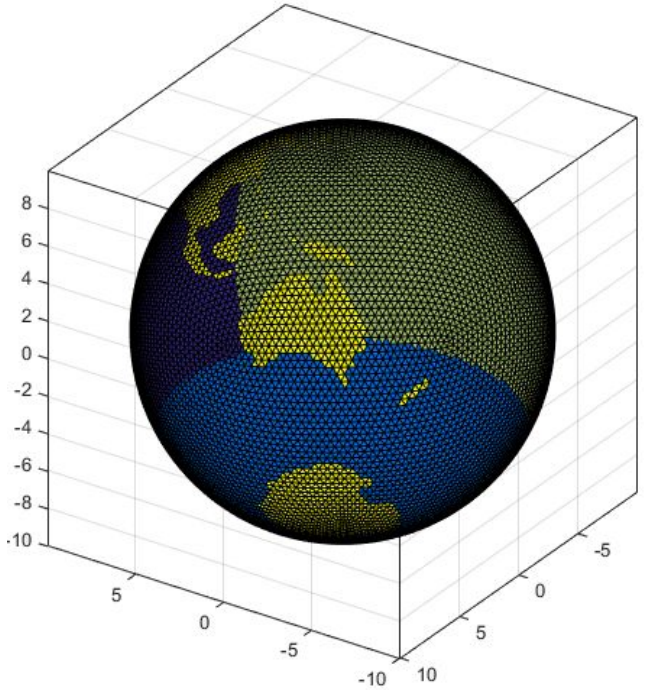
- Different Digital Earth systems use different *geometric entity or entitles* to store geospatial data
  - ✓ E.g. – at vertices, at centroids of the triangle, at midpoint of edges etc.



Computer-based globe model *(cont.….)*

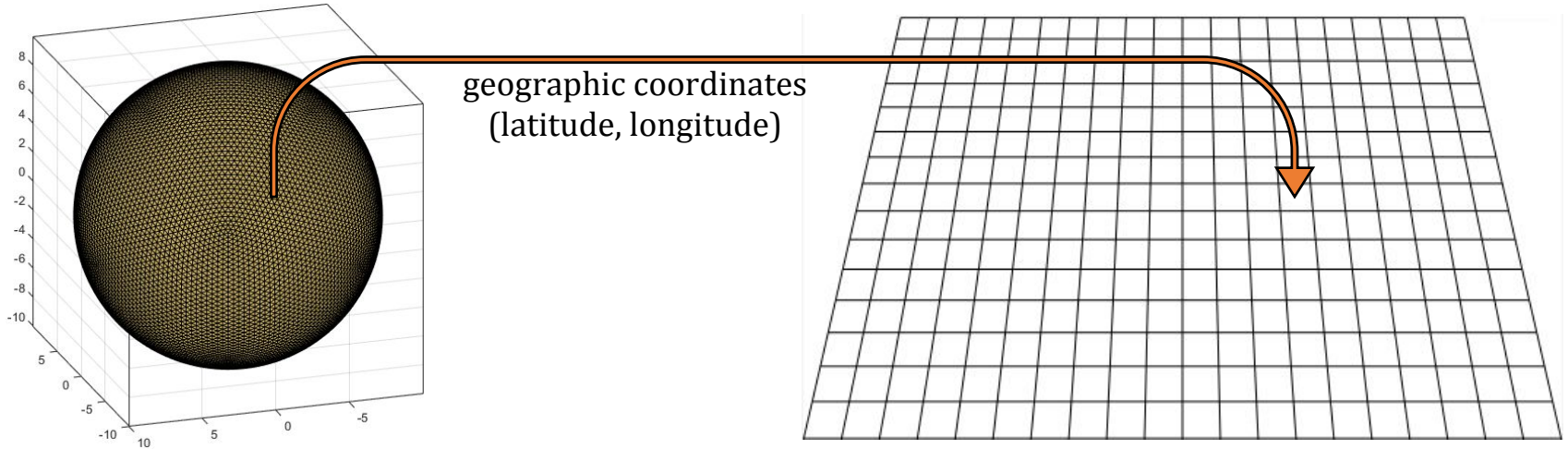
- Data can be visualized with proper colormap applied on these *geometric entities*





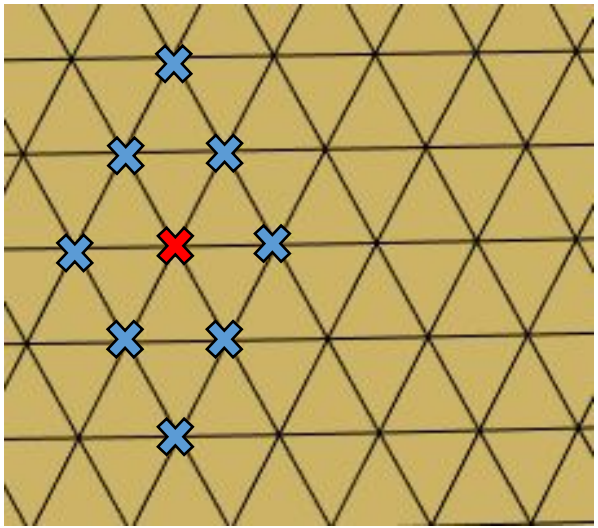
Data Structure for Geometric Entity

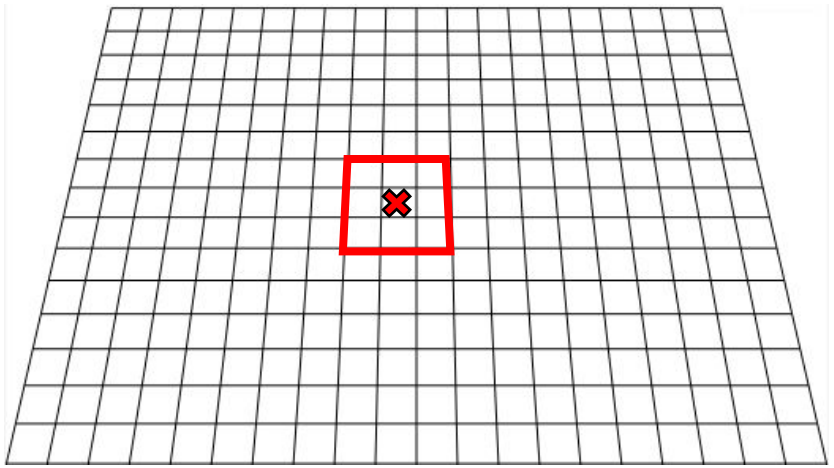
- Storing Geometric information into a data structure –
  - ✓ Array or List



Data Structure for Geometric Entity *(cont.….)*

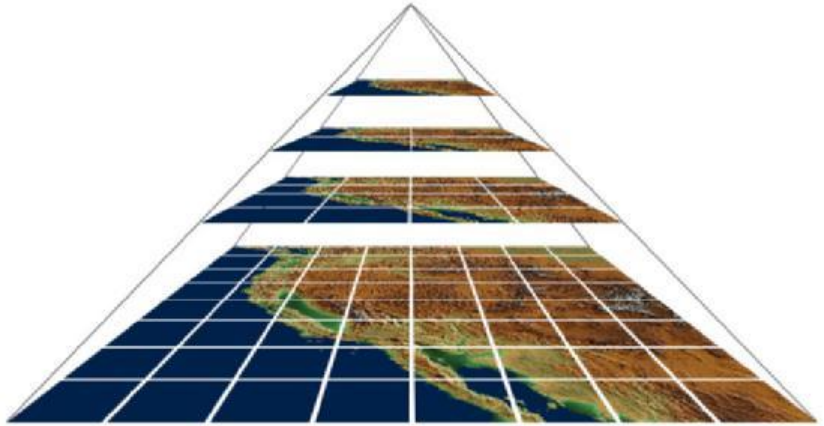
- Why Data Structure is important –
  - ✓ Accessing neighborhood

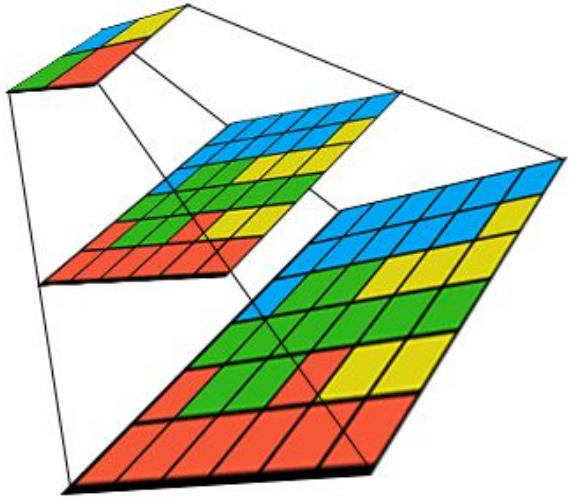




Data Structure for Geometric Entity *(cont.….)*

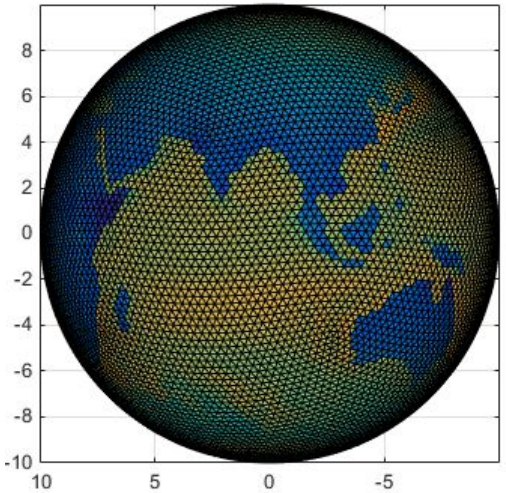
- Why Data Structure is important –
  - ✓ Accessing neighborhood, multi-resolutions etc.





The ICON Grid

- **ICO**sahedral **N**on-hydrostatic model
  - ✓ Joint project of German Weather Service (DWD) and Max-Planck-Institute for Meteorology (MPI-M)
  - ✓ Used for numerical weather prediction as well as for future climate predictions.



Max-Planck-Institut
für Meteorologie

DWD

Study on ICON Grid

- Can be described with three descriptors:
  - ✓ Dimension :

    - Specifies the size of data and variables
  - ✓ Attributes :

    - Metadata, relation between variables
  - ✓ Variables :

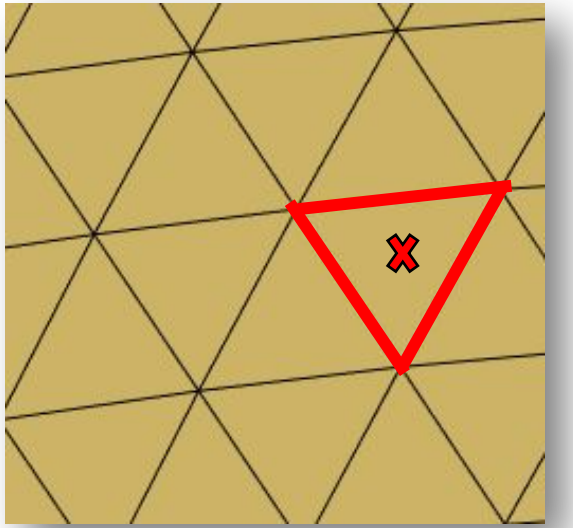
    - Holds data and Geographic coordinates (latitude and longitude) of geometric entity

The ICON Grid *(cont.….)*

Study on Variables : -

clon, clat:

- geographic coordinates of the center of a triangular cell



The ICON Grid *(cont.….)*

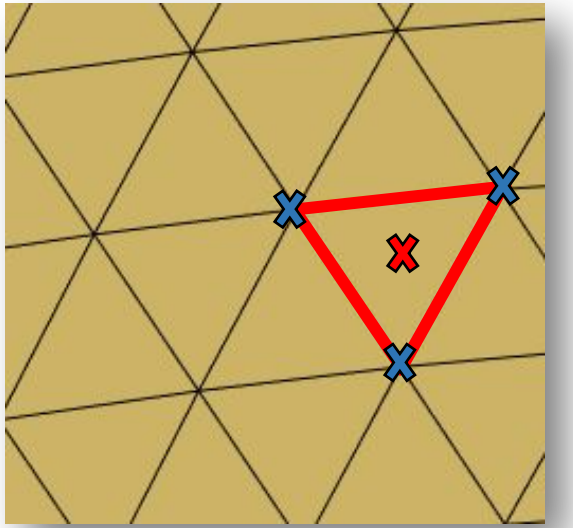
Study on Variables : -

clon, clat:

- geographic coordinates of the center of a triangular cell

clon_vertices, clat_vertices:

- geographic coordinates of three edge vertices of a triangular cell

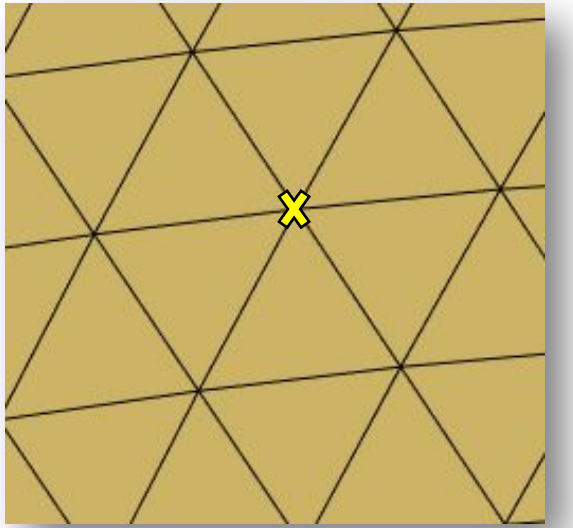


The ICON Grid *(cont.….)*

Study on Variables : -

vlon, vlat:

- geographic coordinates of vertices



The ICON Grid *(cont.….)*

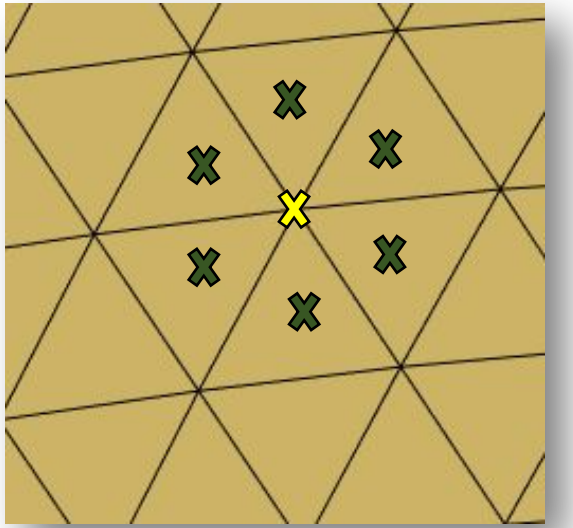
Study on Variables : -

vlon, vlat:

- geographic coordinates of vertices

vlon_vertices, vlat_vertices:

- geographic coordinates of six vertices of hexagons (six neighboring triangle centers)



The ICON Grid *(cont.….)*

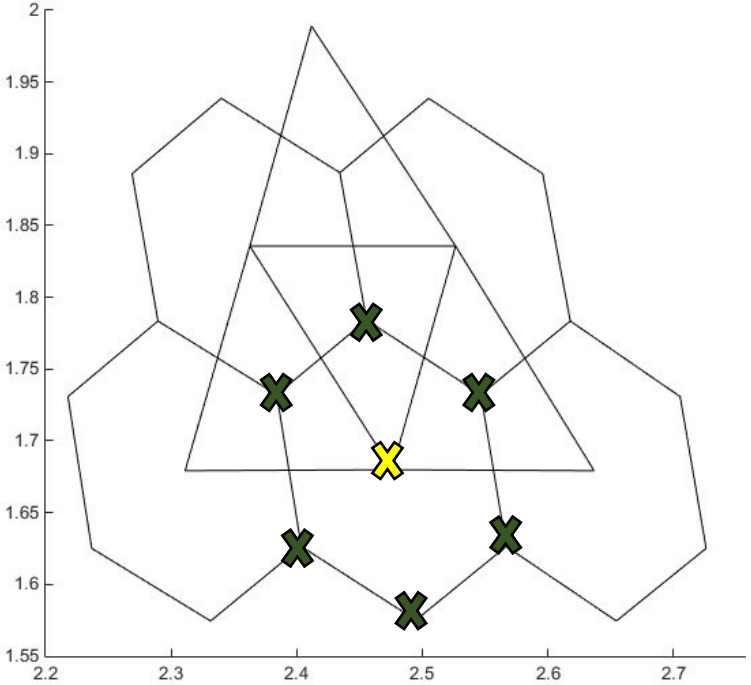
Study on Variables : -

vlon, vlat:

- geographic coordinates of vertices

vlon_vertices, vlat_vertices:

- geographic coordinates of six vertices of hexagons (six neighboring triangle centers)

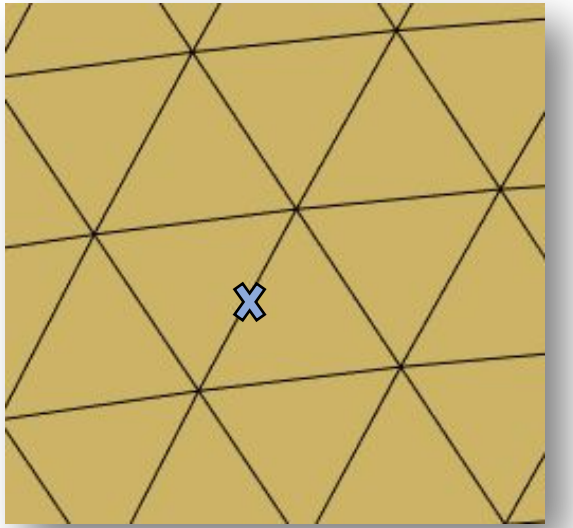


The ICON Grid *(cont.….)*

Study on Variables : -

elon, elat:

- geographic coordinates of edge midpoint vertices



The ICON Grid *(cont.….)*

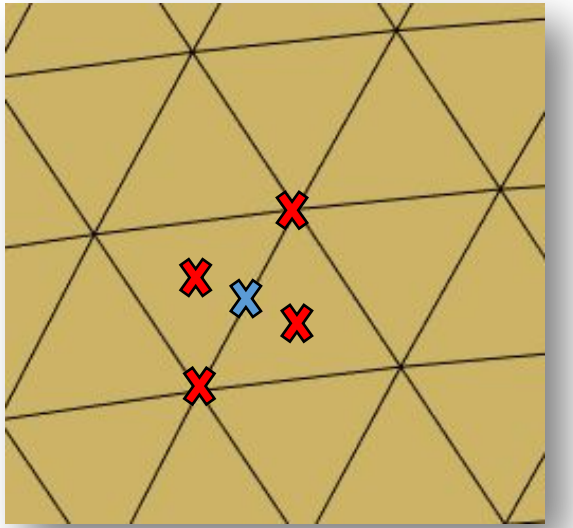
Study on Variables : -

elon, elat:

- geographic coordinates of edge midpoint vertices

elon_vertices, elat_vertices:

- geographic coordinates of four neighboring vertices of edge midpoint



The ICON Grid *(cont.….)*

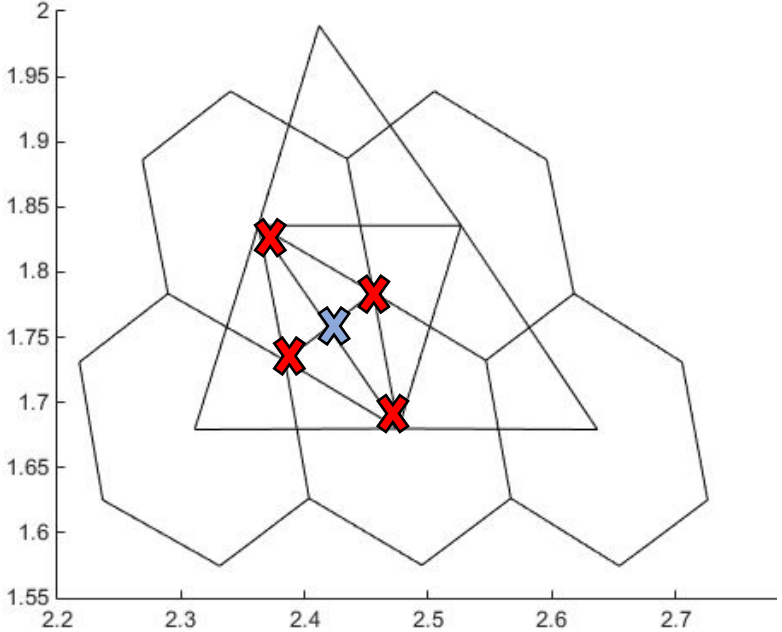
Study on Variables : -

elon, elat:

- geographic coordinates of edge midpoint vertices

elon_vertices, elat_vertices:

- geographic coordinates of four neighboring vertices of edge midpoint

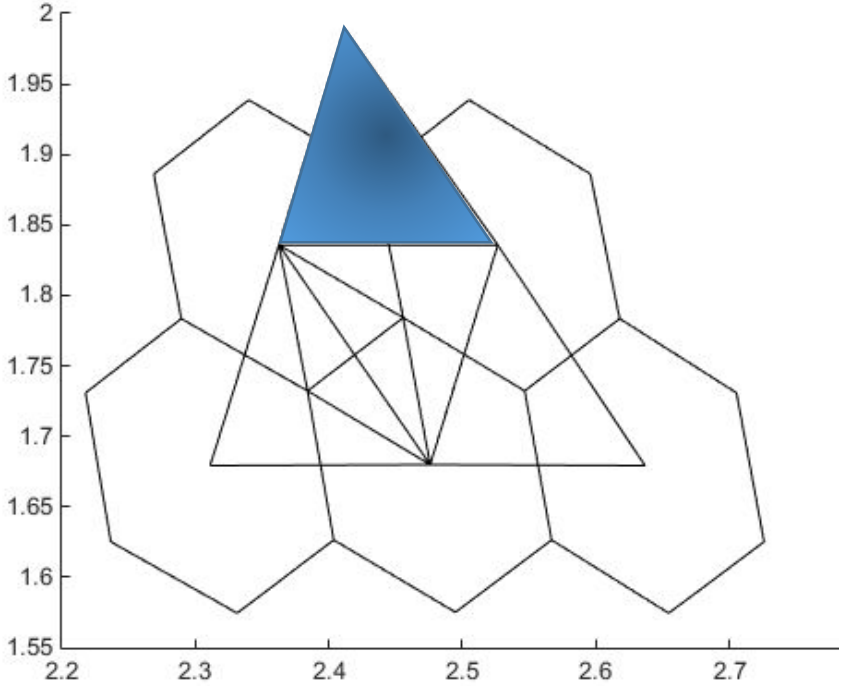


The ICON Grid *(cont.….)*

Study on Data : -

Data stored in –

- triangles

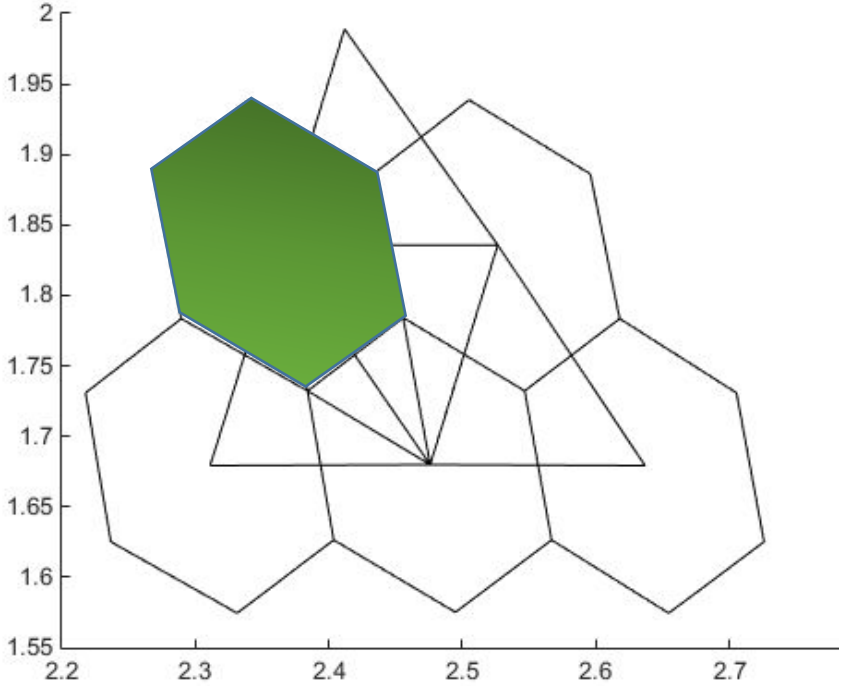


The ICON Grid *(cont.….)*

Study on Data : -

Data stored in –

- triangles
- hexagons

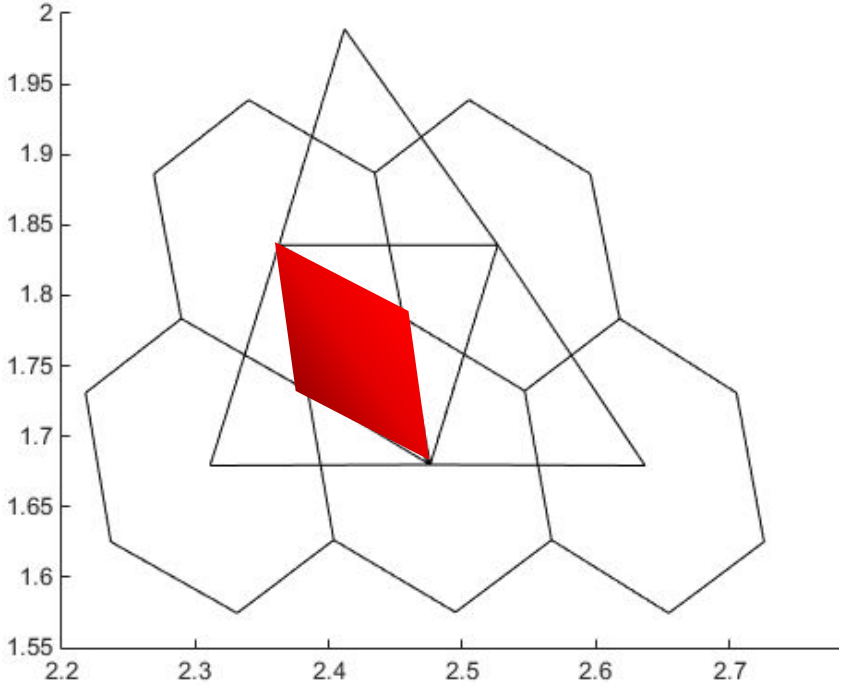


The ICON Grid *(cont.….)*

Study on Data : -

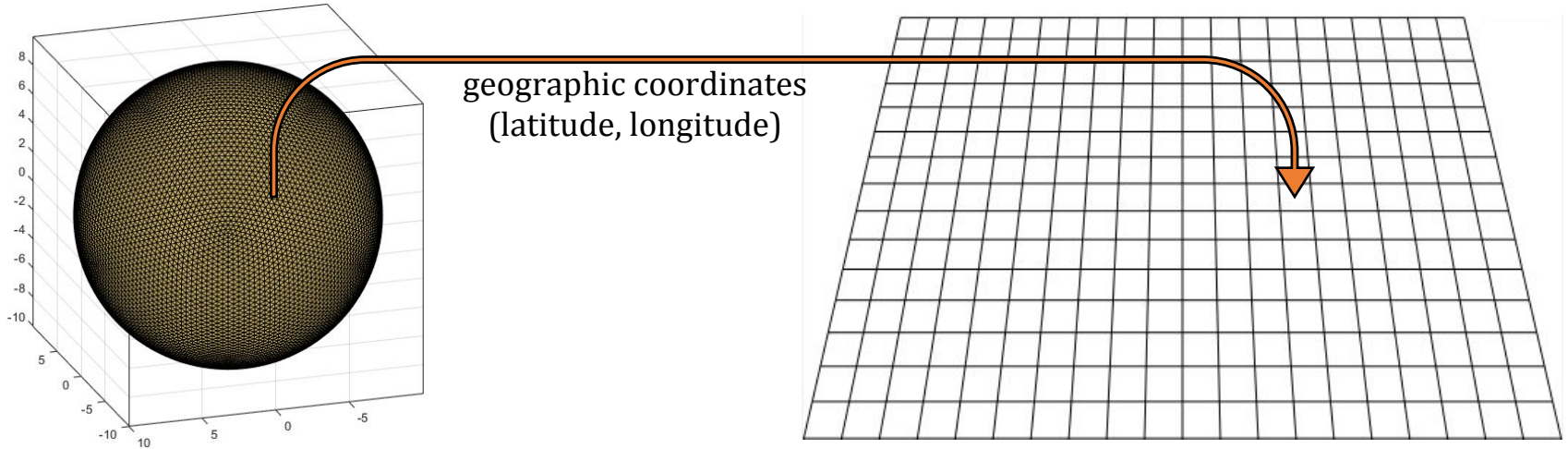
Data stored in –

- triangles
- hexagons
- rectangle



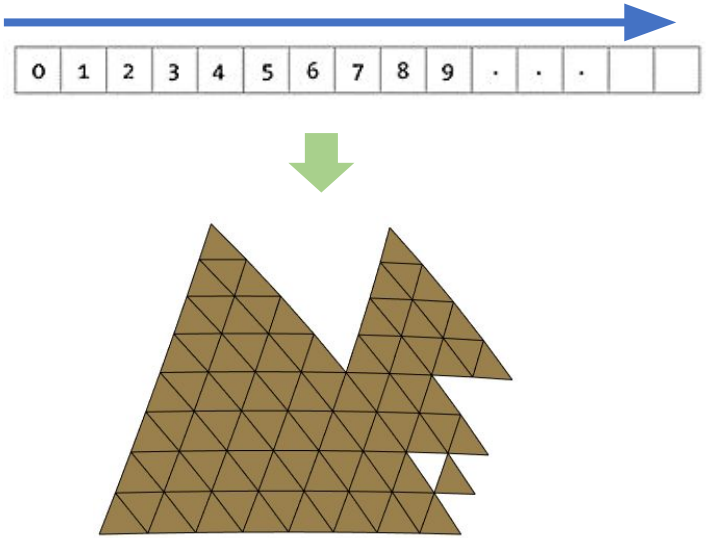
The ICONverter

- ***ICON*** + ***Con***verter
- Storing geometric layout of ICON grid (vertices)into array structure

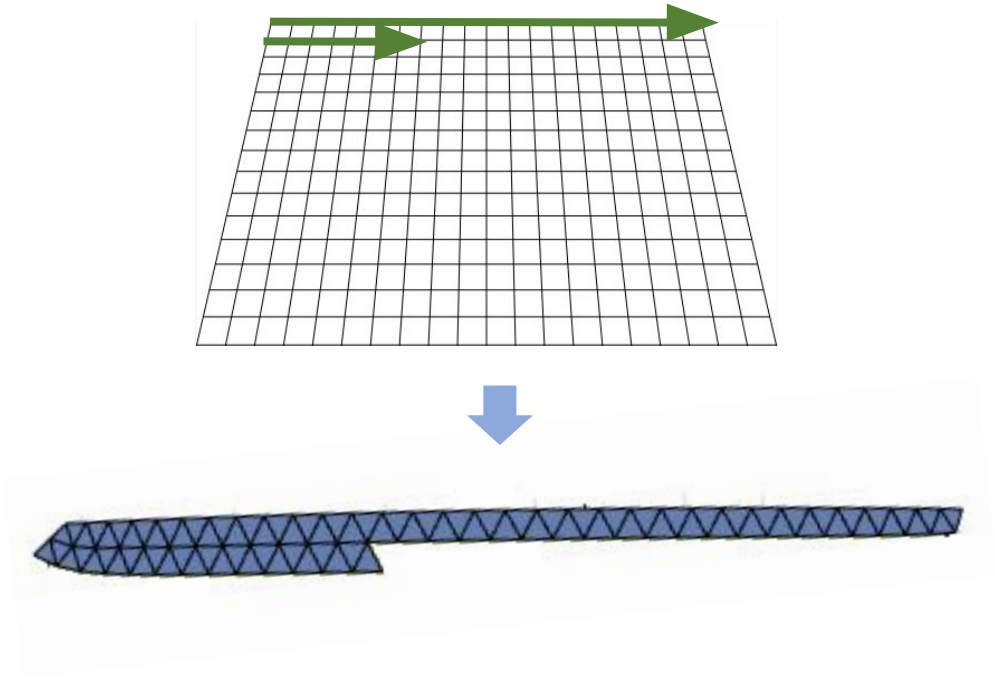


The ICONverter : *Overview*

Before conversion

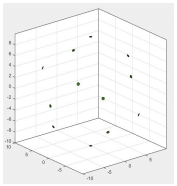


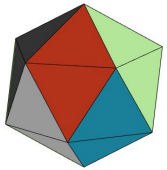
After conversion



The ICONverter : *Conversion Pipeline*

Build initial Earth skeleton (Icosahedron)

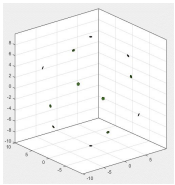


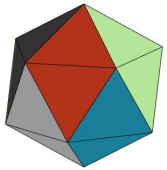


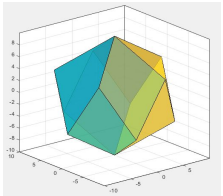
The ICONverter : *Conversion Pipeline*

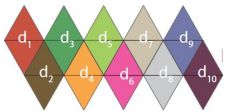
Build initial Earth skeleton (Icosahedron)

Obtain diamond shape architecture from Icosahedron







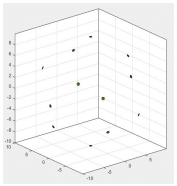


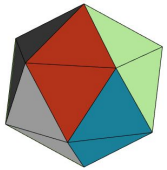
The ICONverter : *Conversion Pipeline*

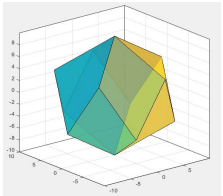
Build initial Earth skeleton (Icosahedron)

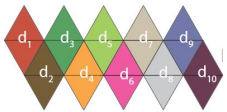
Obtain diamond shape architecture from Icosahedron

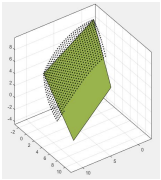
Generate array for preliminary Diamond





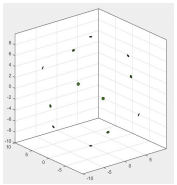


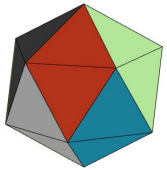




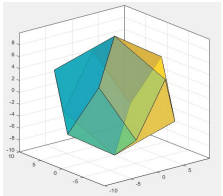
The ICONverter : *Conversion Pipeline*

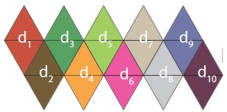
Build initial Earth skeleton (Icosahedron)



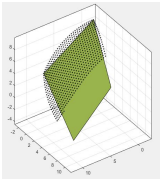


Obtain diamond shape architecture from Icosahedron

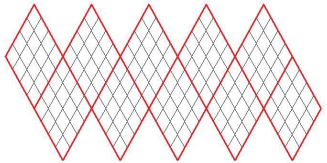


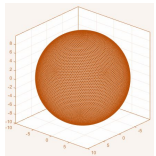


Generate array for preliminary Diamond

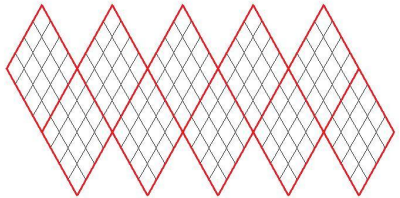


Generate array for rest of the Diamonds



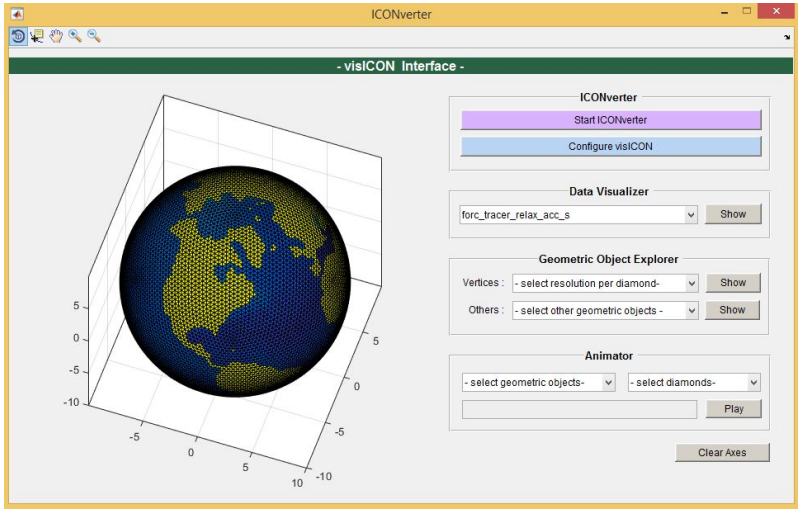


The ICONverter : *Visualization Pipeline*



Array generated by ICONverter

Visualization pre-processing

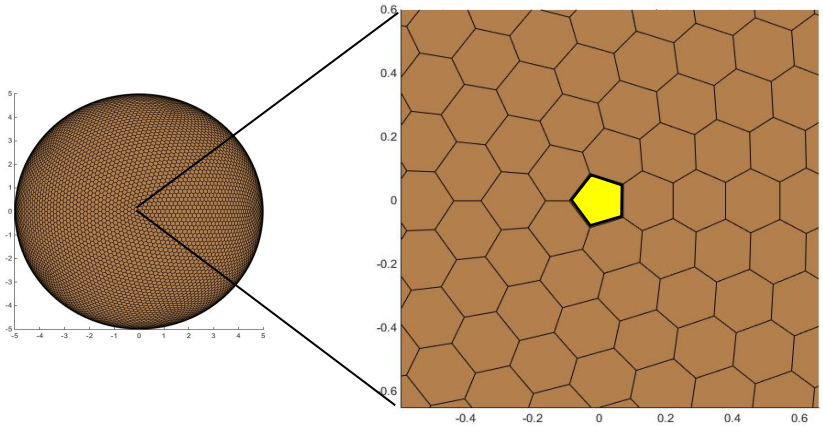


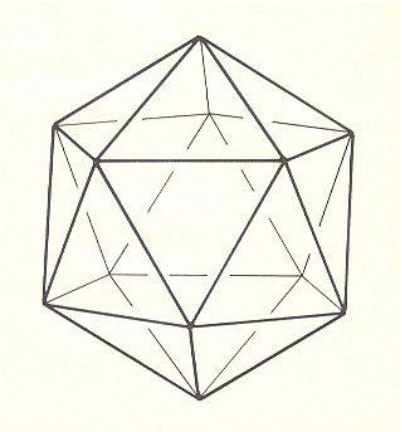
User Interface : visICON
(***vis***ualization + ***ICON***)

The ICONverter : *Initial Icosahedron*

Building initial Earth skeleton (Icosahedron) :

- Irregular hexagons (*pentagons*) is formed while covering Earth sphere with hexagonal cells
- There are total 12 such pentagons on the entire sphere

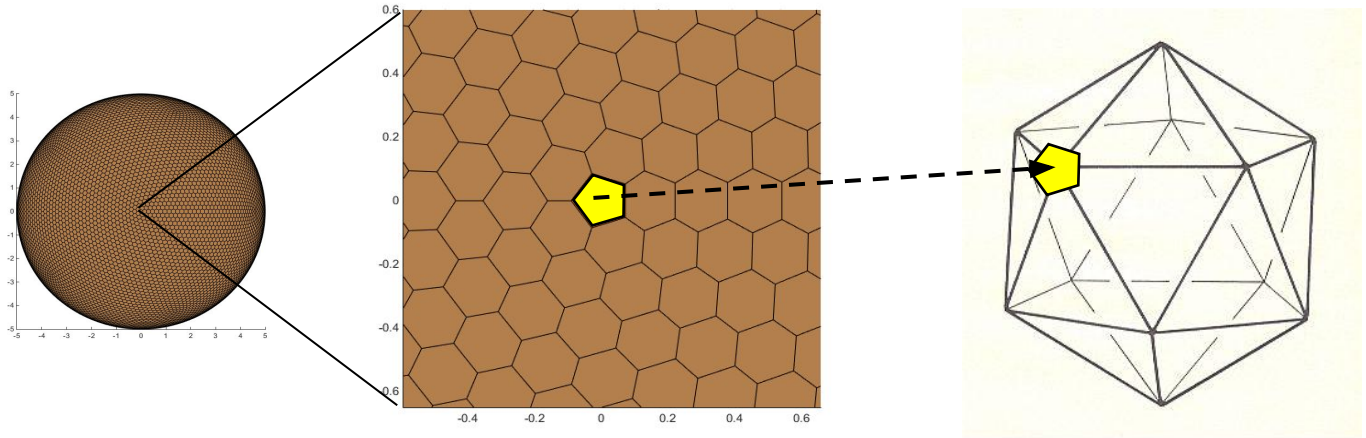




The ICONverter : *Initial Icosahedron*

Building initial Earth skeleton (Icosahedron) :

- Irregular hexagons (*pentagons*) is formed while covering Earth sphere with hexagonal cells
- There are total 12 such pentagons on the entire sphere
- 12 pentagons are pointing to the 12 vertices of the Icosahedron



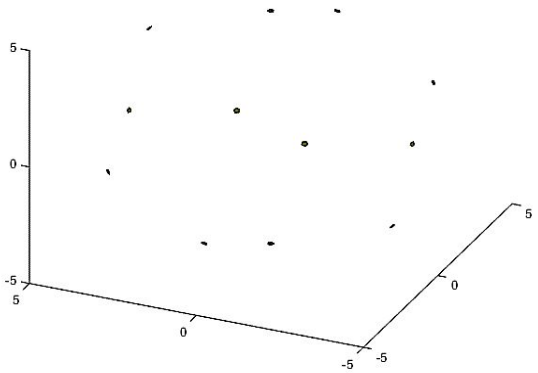
The ICONverter : *Initial Icosahedron*

Building initial Earth skeleton (Icosahedron) :

- ***Finding pentagon:***

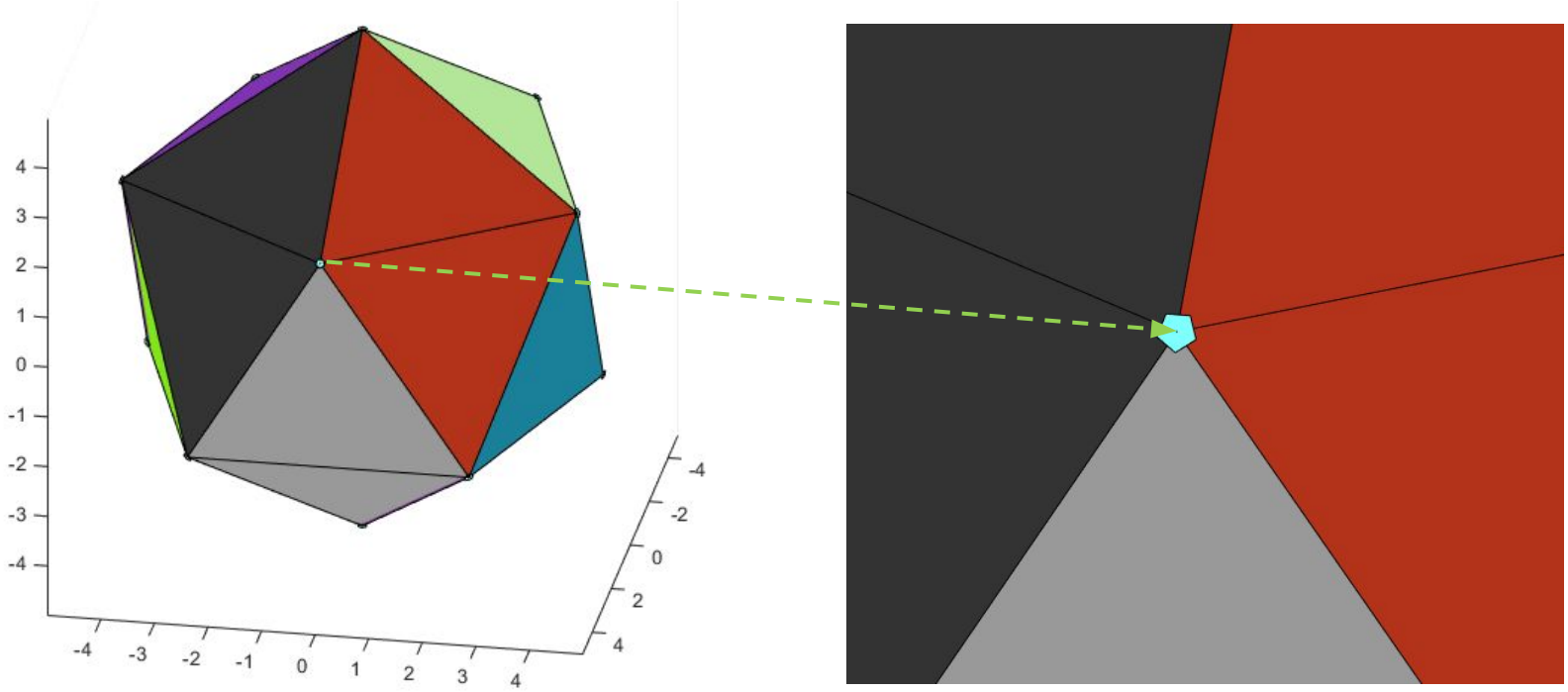
  Repeated values for last two vertices in (*vlon_vertices, vlat_vertices*) entries

	1	2	3	4	5	6
1	1.4217	1.4665	1.3941	1.3045	1.3216	1.3216
2	0.0725	-0.0277	-0.0895	-0.0277	0.0725	0.0725
3	4.4390	4.4390	4.4838	4.5115	4.4838	4.4838
4	-3.5603	-3.5774	-3.6499	-3.6775	-3.6222	-3.6222
5	-3.6222	-3.6775	-3.6499	-3.5774	-3.5603	-3.5603
6	1.3217	1.3046	1.3941	1.4666	1.4218	1.4218
7	3.6499	3.6775	3.6222	3.5603	3.5774	3.5774
8	-1.3941	-1.3046	-1.3217	-1.4218	-1.4666	-1.4666
9	-4.5115	-4.4838	-4.4390	-4.4390	-4.4838	-4.4838
10	-1.3941	-1.4666	-1.4218	-1.3217	-1.3046	-1.3046
11	3.6499	3.5774	3.5603	3.6222	3.6775	3.6775
12	0.0277	0.0896	0.0277	-0.0725	-0.0725	-0.0725



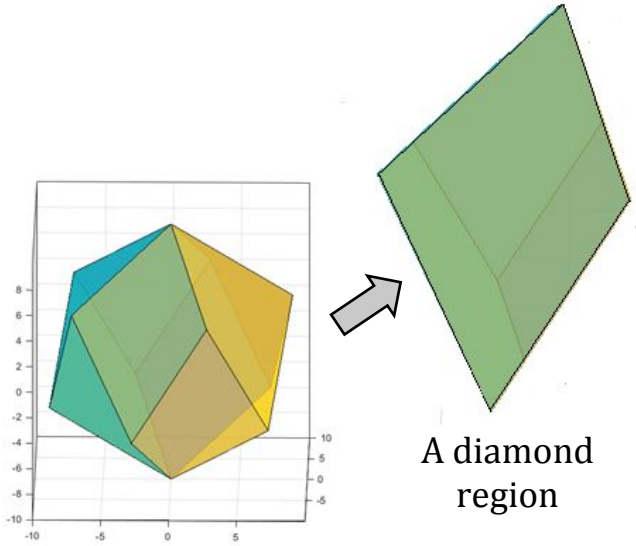
The ICONverter : *Initial Icosahedron*

Building initial Earth skeleton (Icosahedron) :

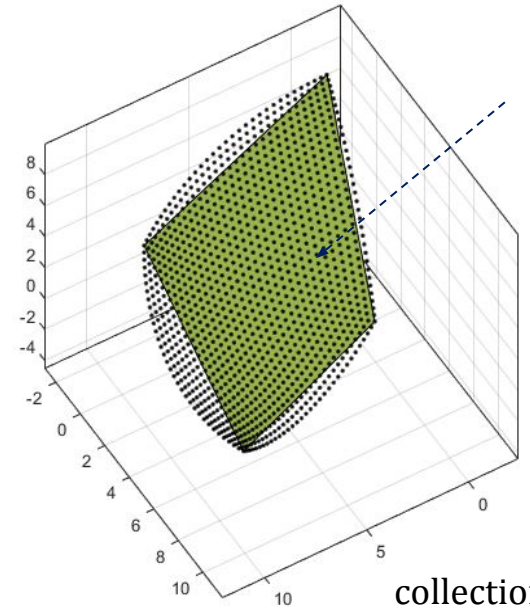


The ICONverter : *Diamonds from Icosahedron*

- Each diamond can be viewed as region on the Earth that covers a collection of geometric entities (vertices)

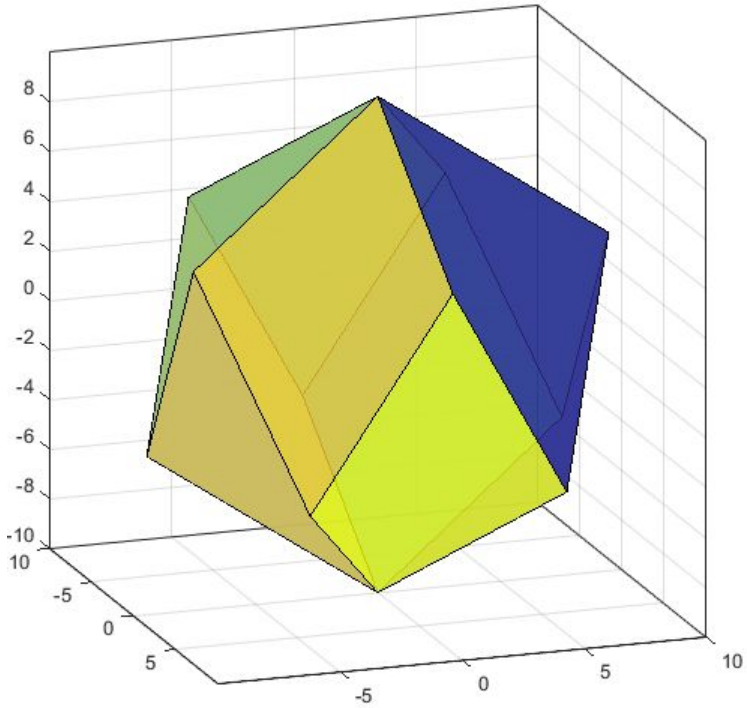


A diamond region

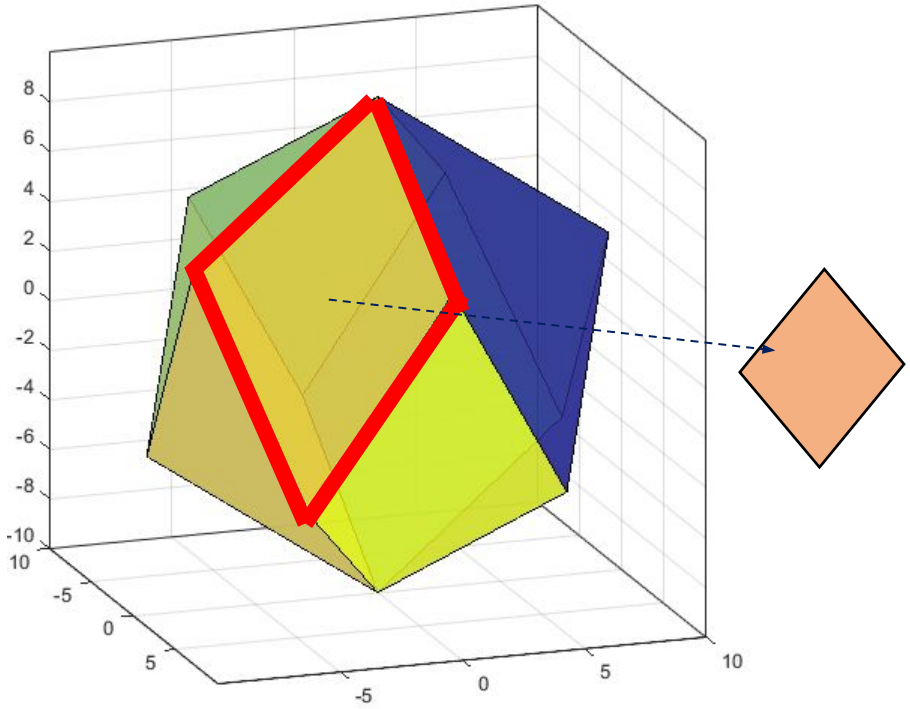


collection of vertices on diamond

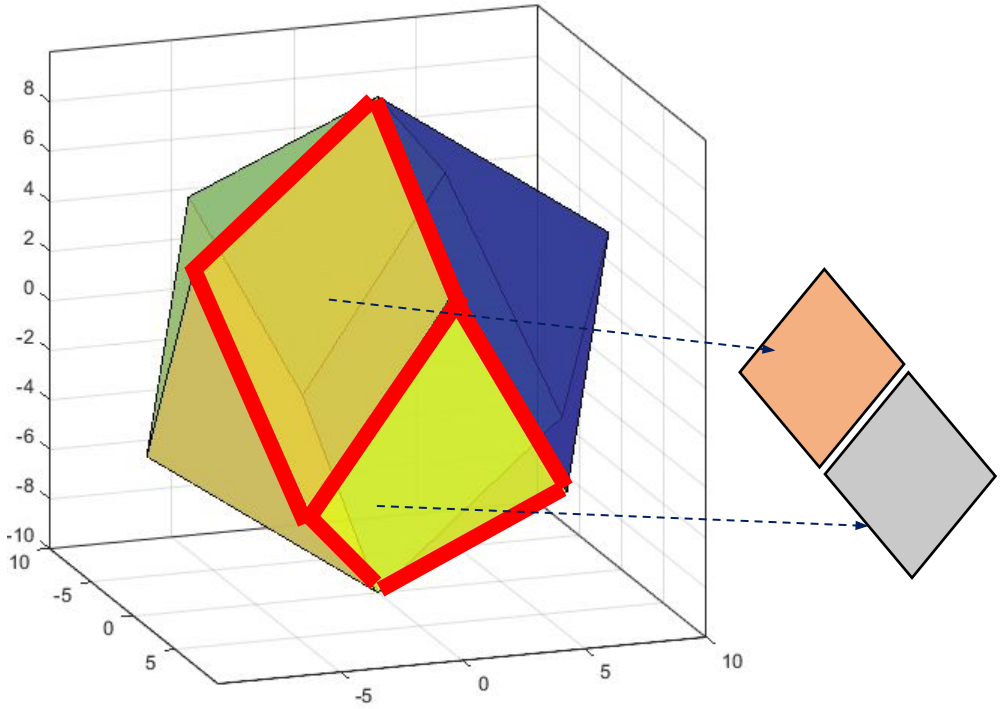
The ICONverter : *Diamonds from Icosahedron*



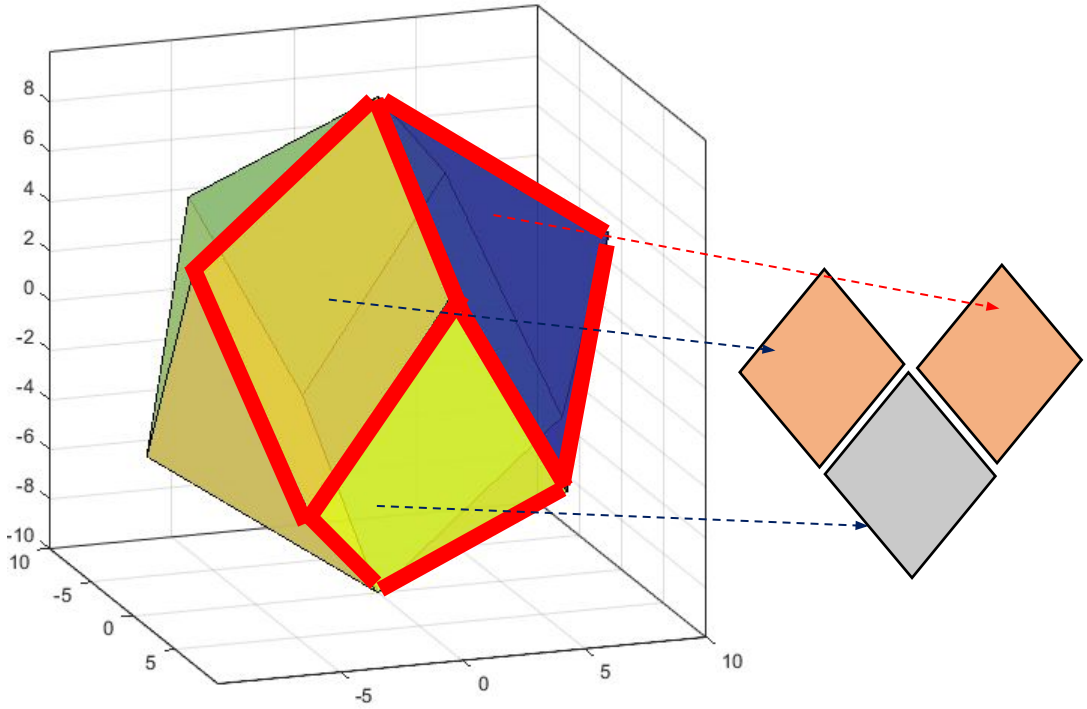
The ICONverter : *Diamonds from Icosahedron*



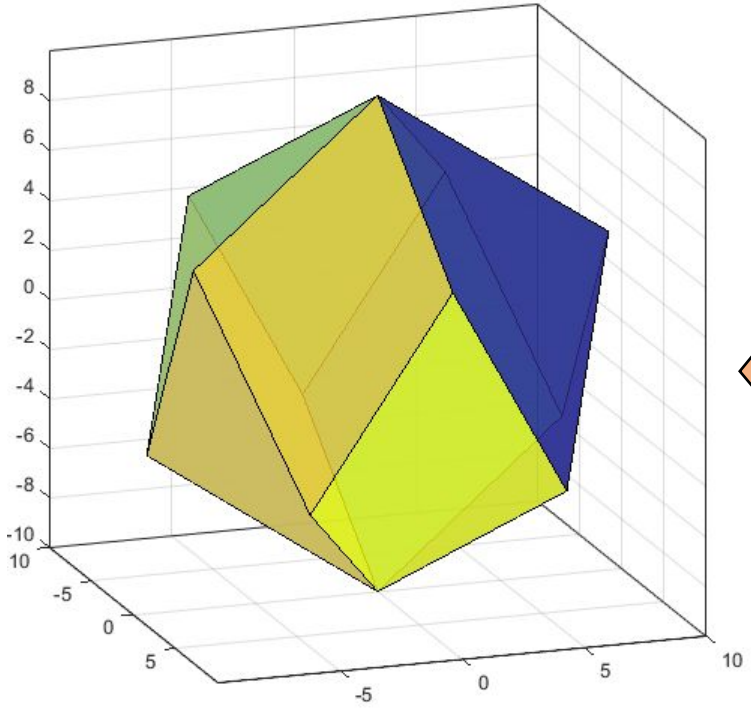
The ICONverter : *Diamonds from Icosahedron*

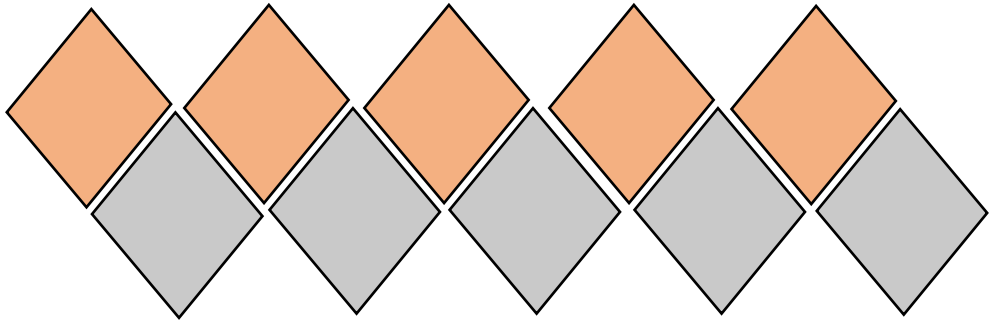


The ICONverter : *Diamonds from Icosahedron*

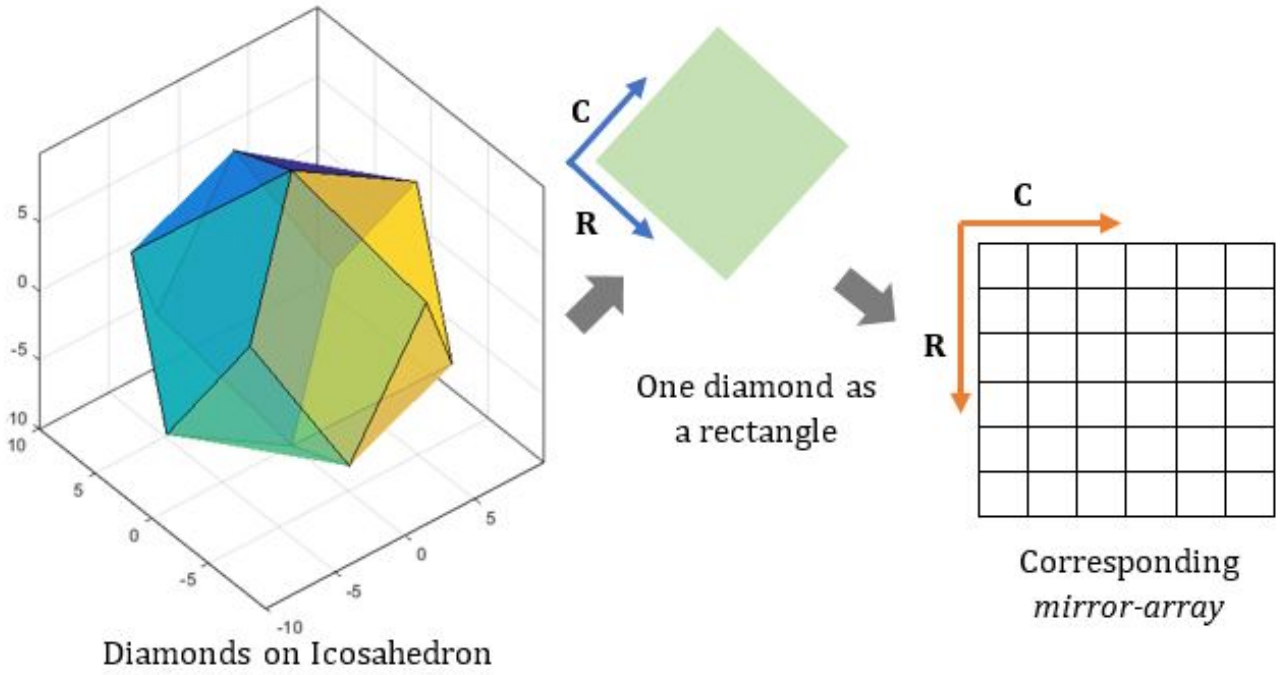


The ICONverter : *Diamonds from Icosahedron*

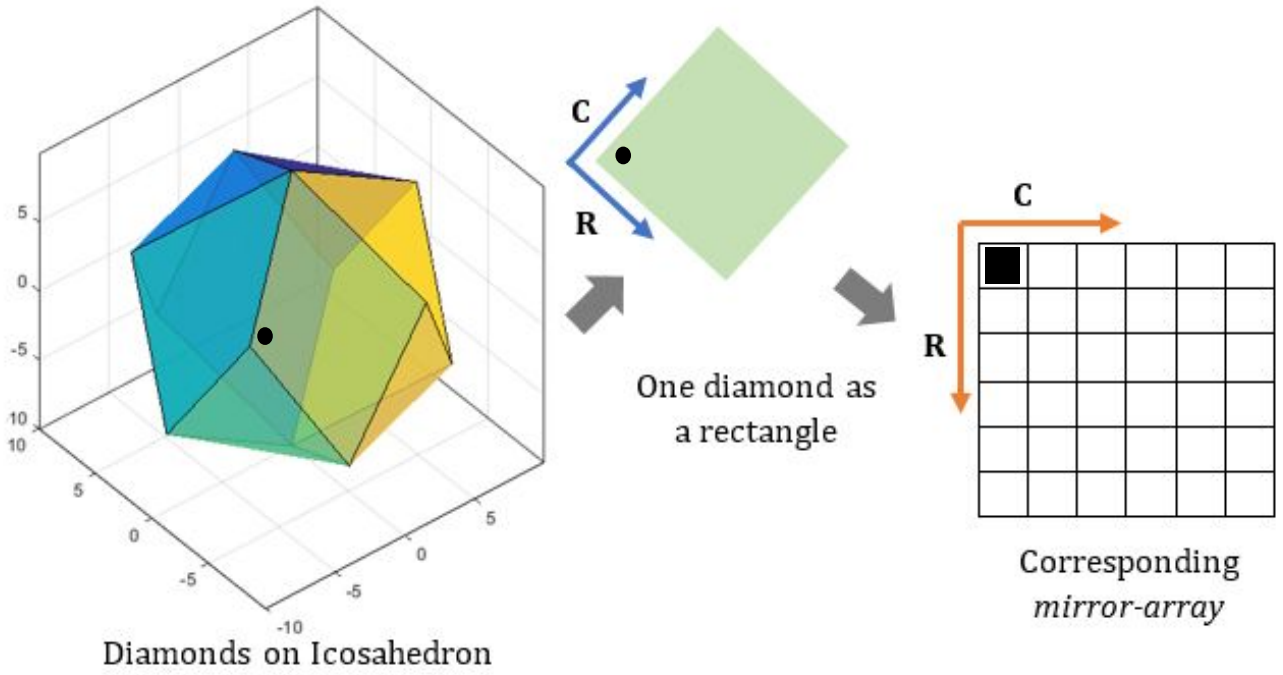




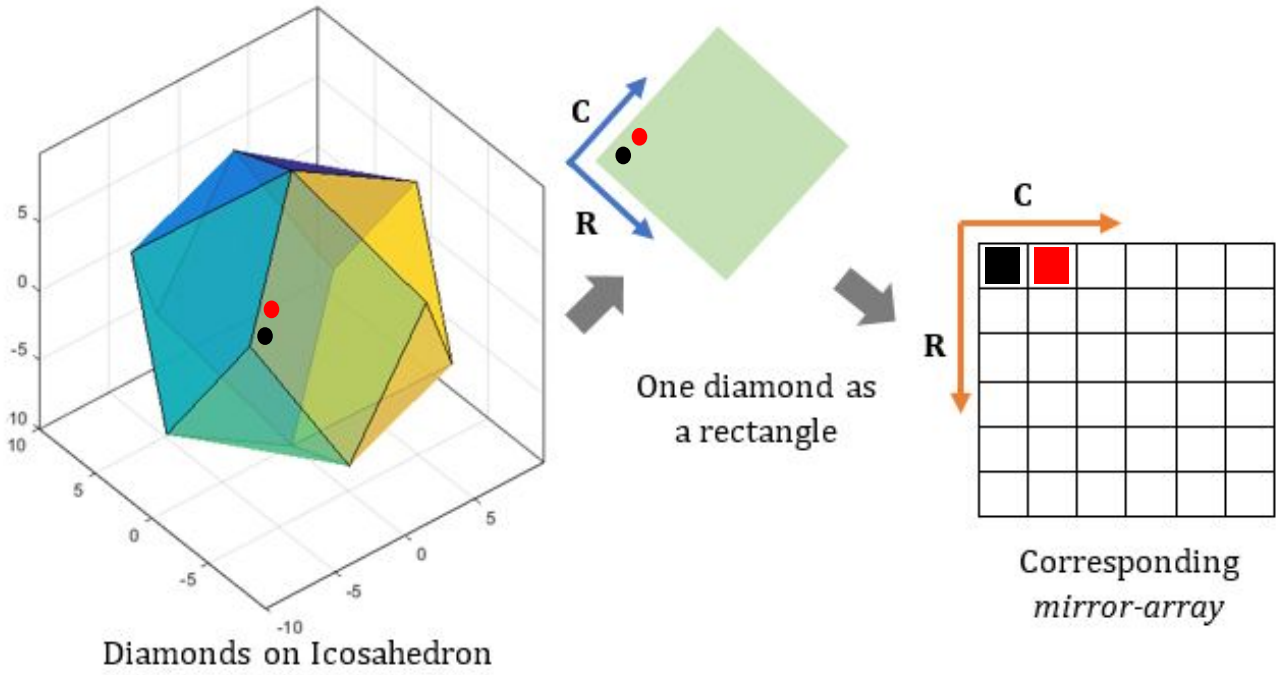
The ICONverter : *Mirror-Array of Diamond*



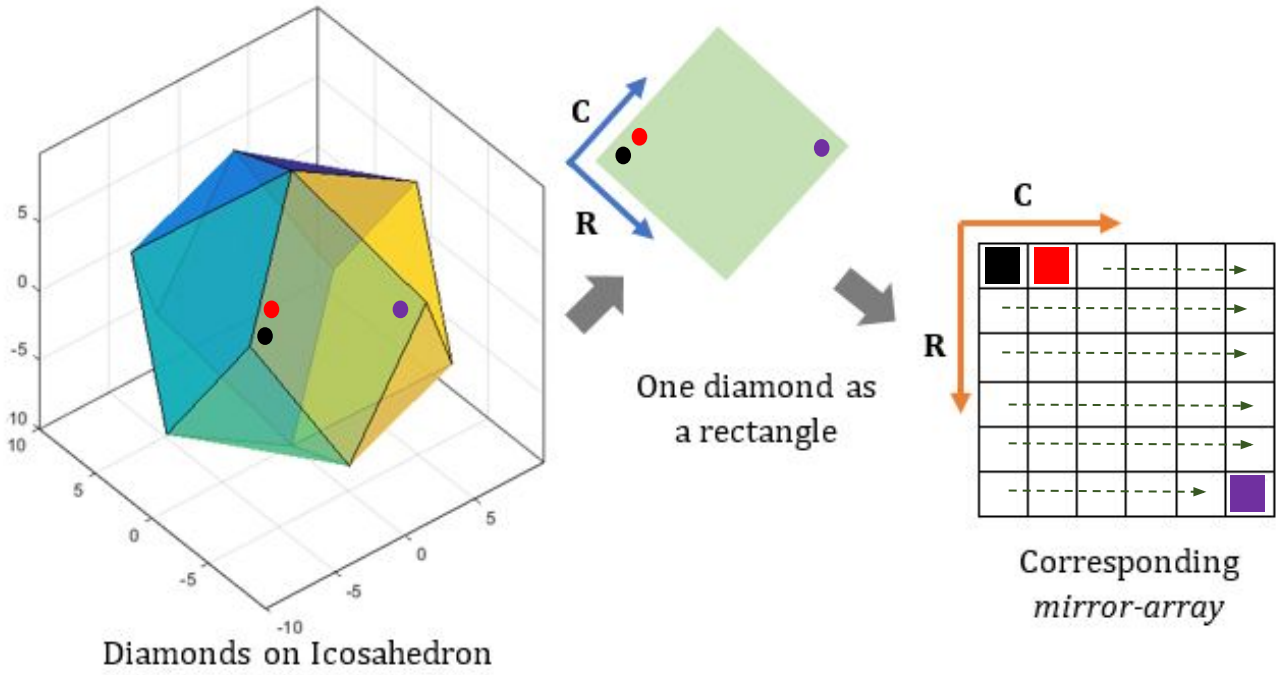
The ICONverter : *Mirror-Array of Diamond*



The ICONverter : *Mirror-Array of Diamond*



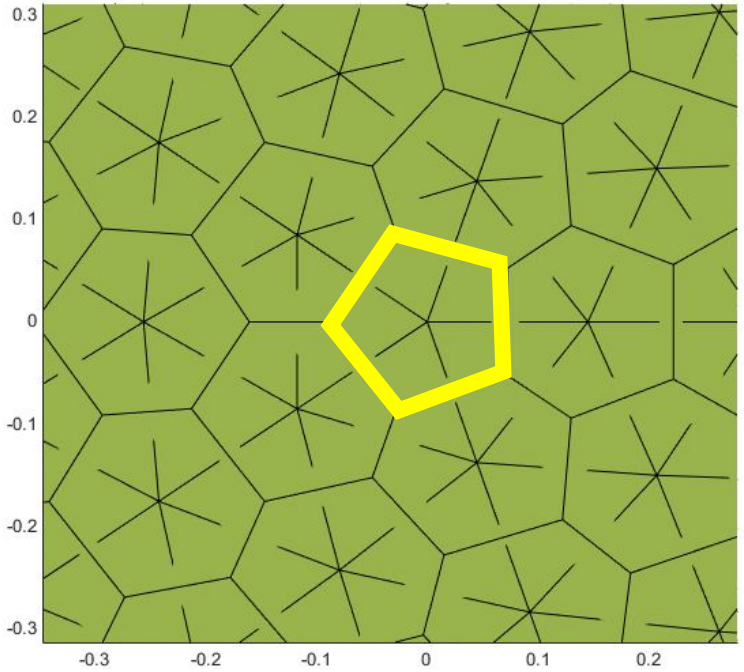
The ICONverter : *Mirror-Array of Diamond*



The ICONverter : *Generating Array for the Preliminary Diamond*

- The pentagons obtained from previous step gives us the vertices of the Icosahedron

- But we are not going to use it !

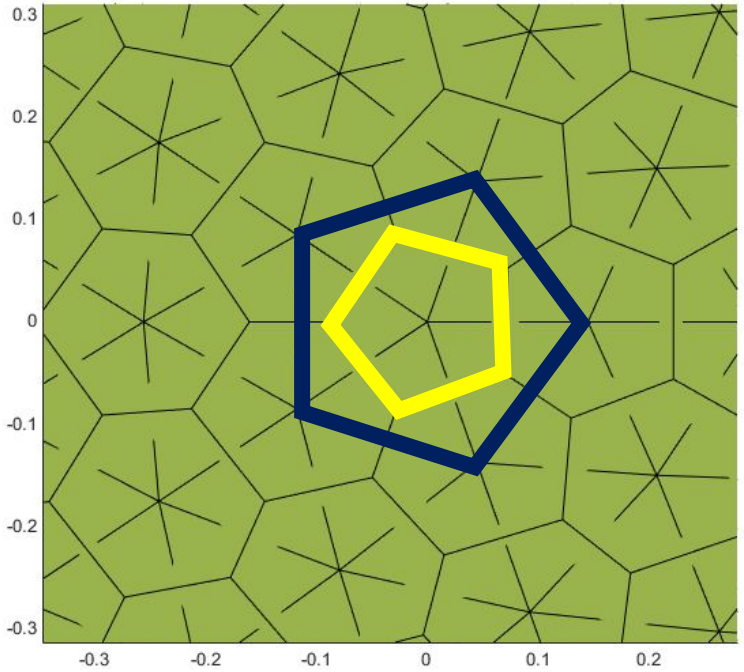


The ICONverter : *Generating Array for the Preliminary Diamond*

- The pentagons obtained from previous step gives us the vertices of the Icosahedron

- But we are not going to use it !

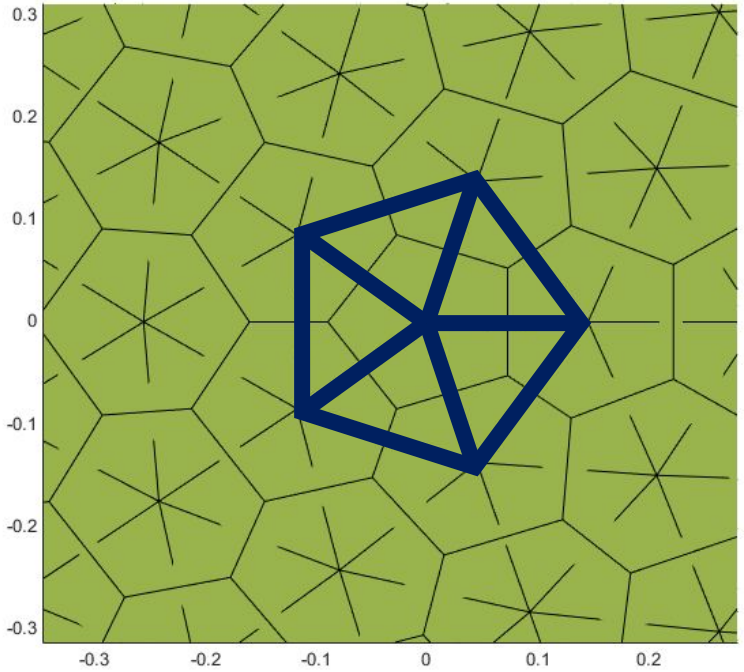
- Pentagons that are going to be used further are the five neighboring hexagons' centroids



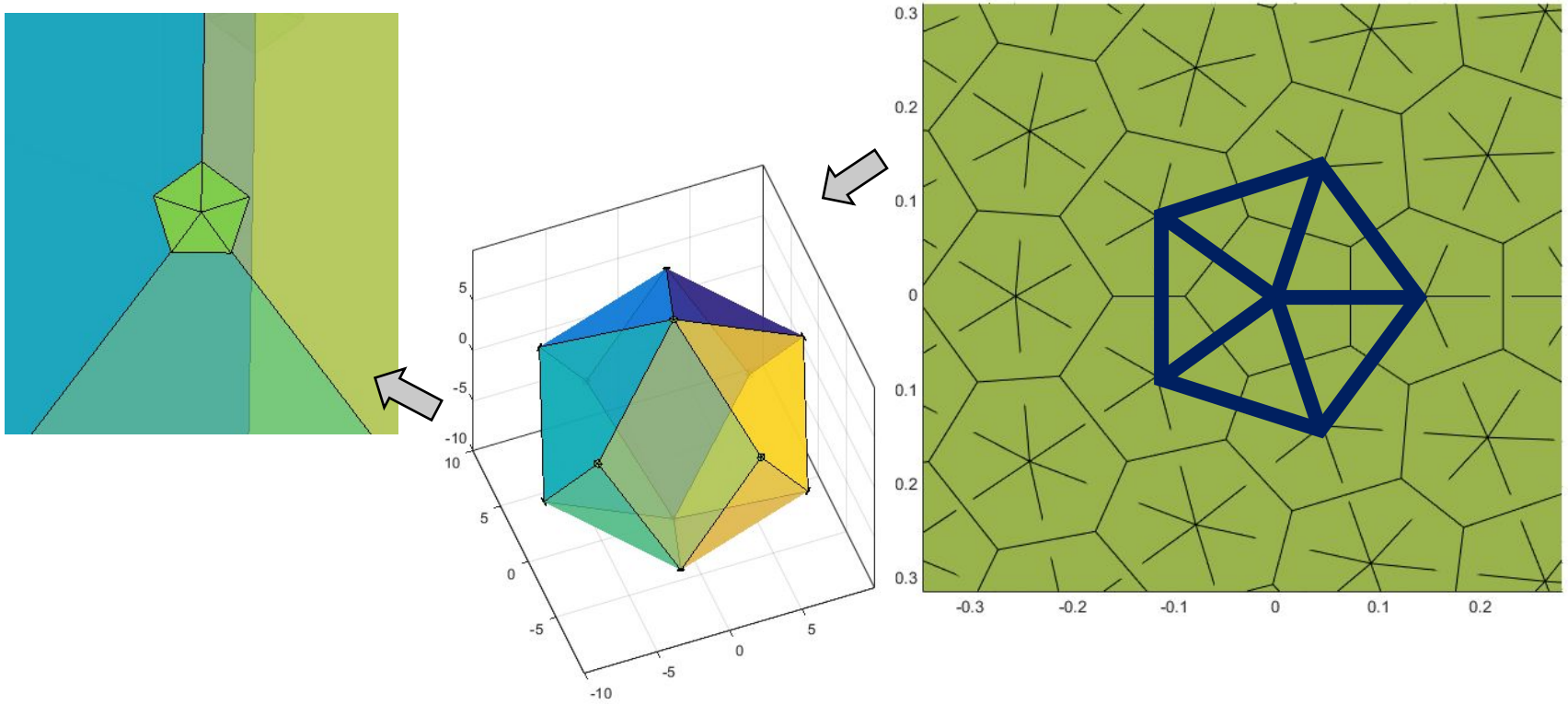
The ICONverter : *Generating Array for the Preliminary Diamond*

- To find them, we can simply search for five triangles that have shared the Icosahedron vertices.

- Those five triangle will give us the *usable* pentagon

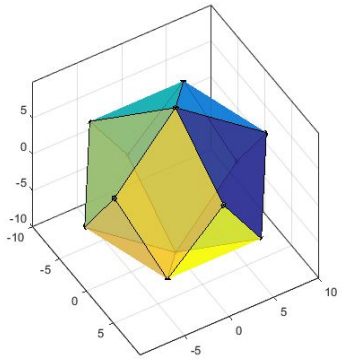


The ICONverter : *Generating Array for the Preliminary Diamond*



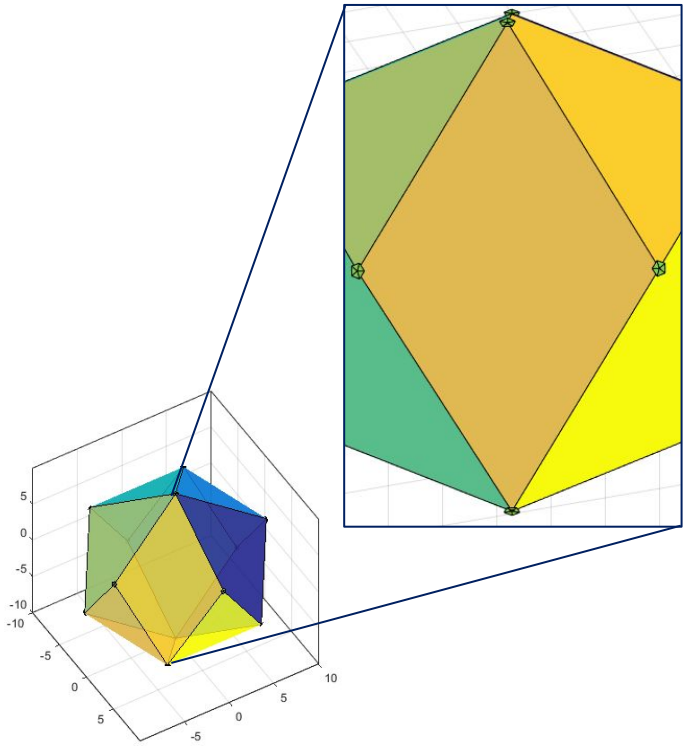
The ICONverter : *Generating Array for the Preliminary Diamond*

Initialize Columns and Rows



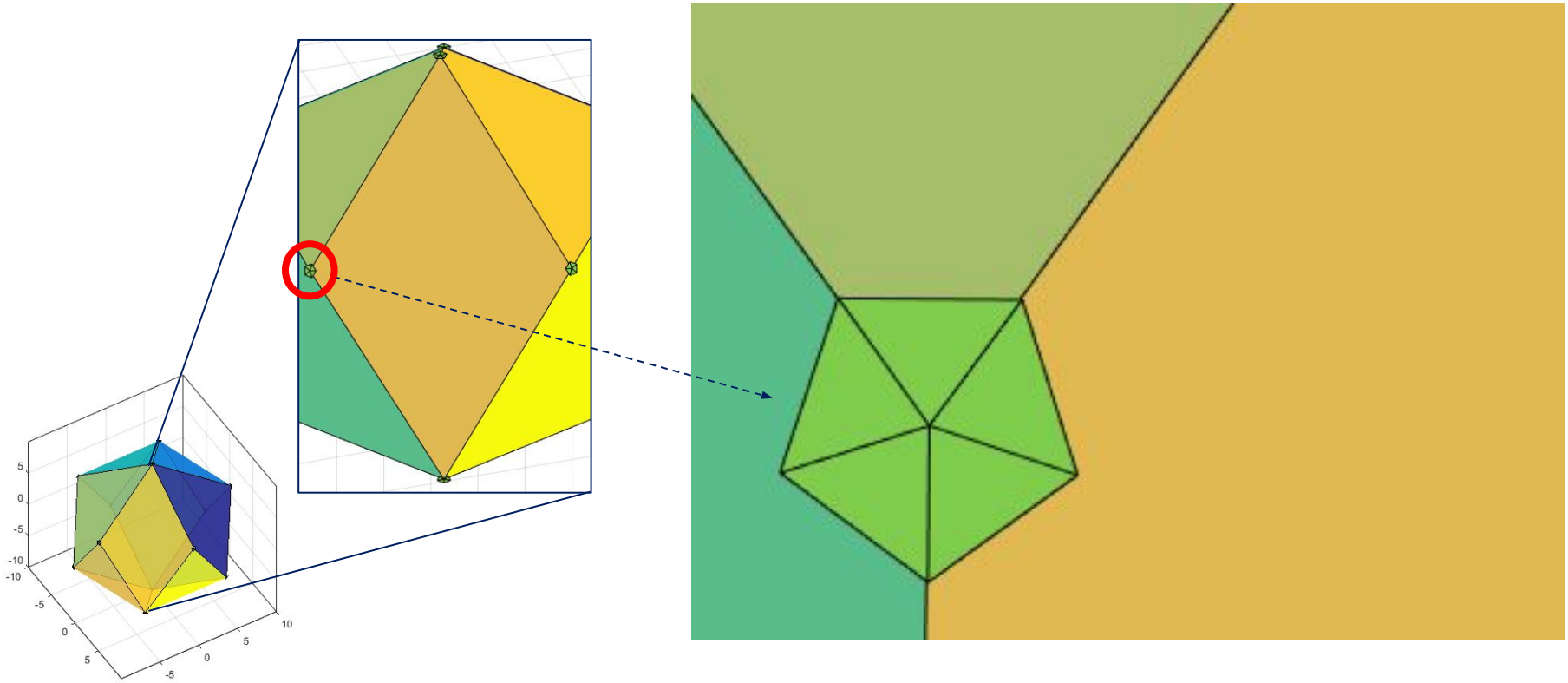
The ICONverter : *Generating Array for the Preliminary Diamond*

Initialize Columns and Rows



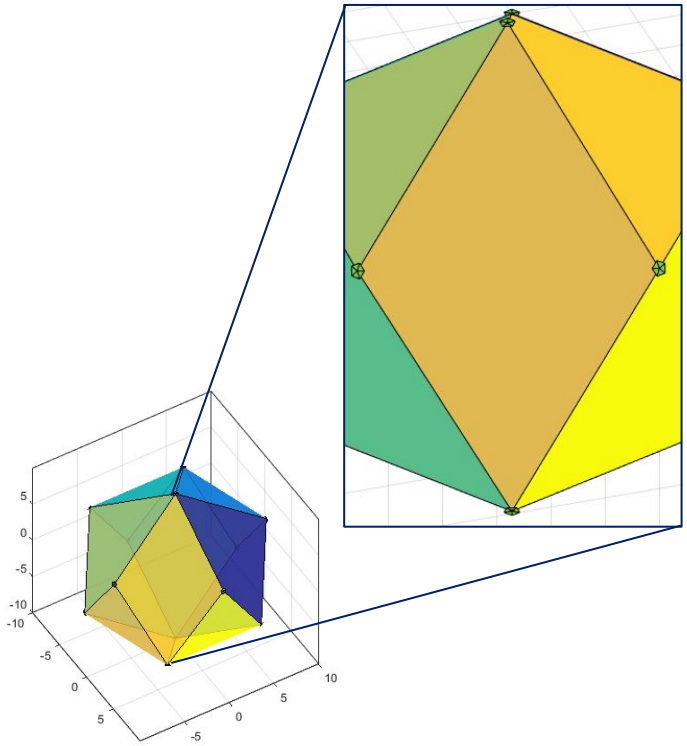
The ICONverter : *Generating Array for the Preliminary Diamond*

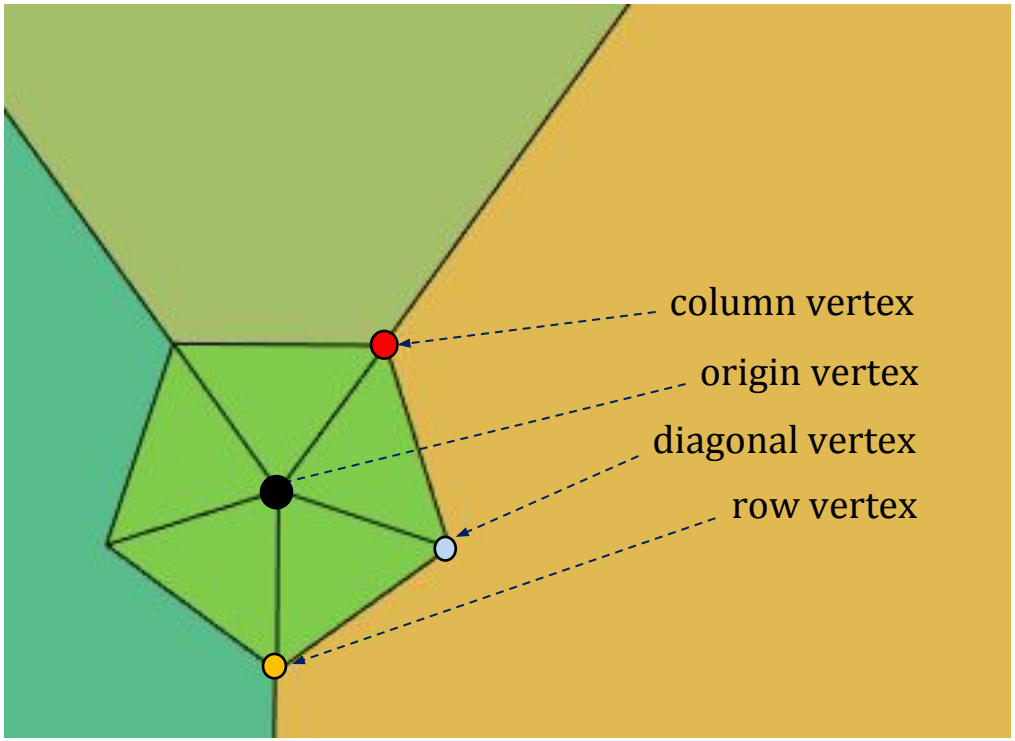
Initialize Columns and Rows



The ICONverter : *Generating Array for the Preliminary Diamond*

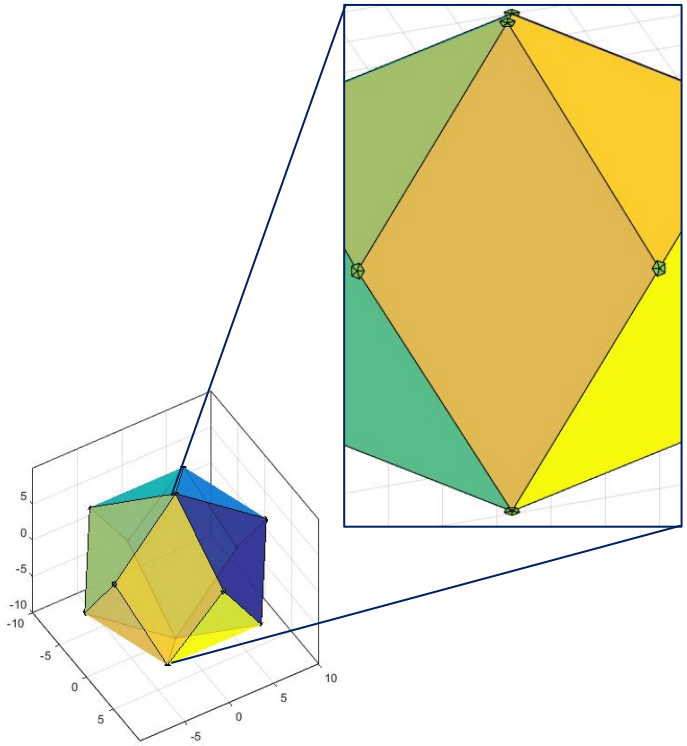
Initialize Columns and Rows

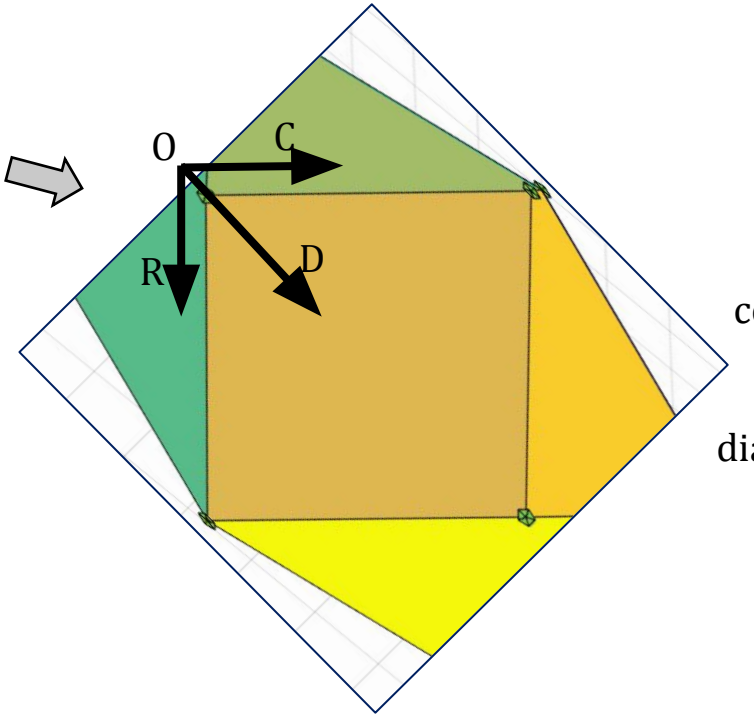




The ICONverter : *Generating Array for the Preliminary Diamond*

Initialize Columns and Rows





column vertex = C

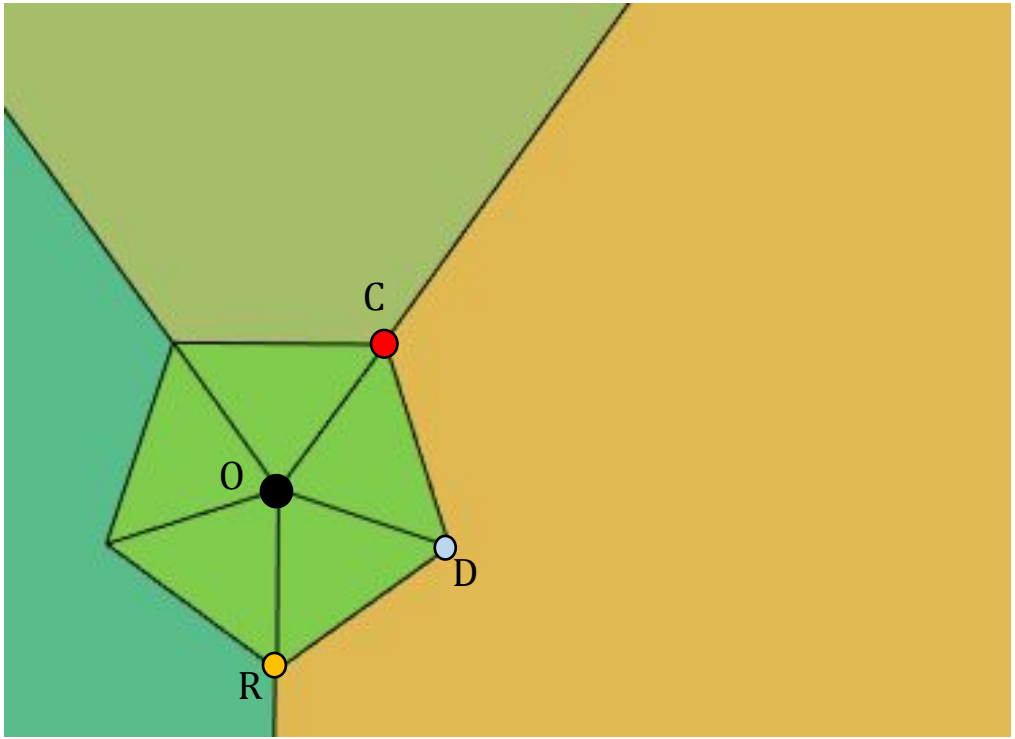
origin vertex = O

diagonal vertex = D

row vertex = R

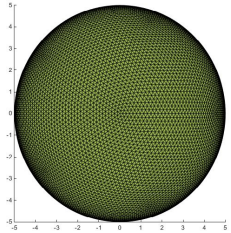
The ICONverter : *Generating Array for the Preliminary Diamond*

Initialize Columns and Rows

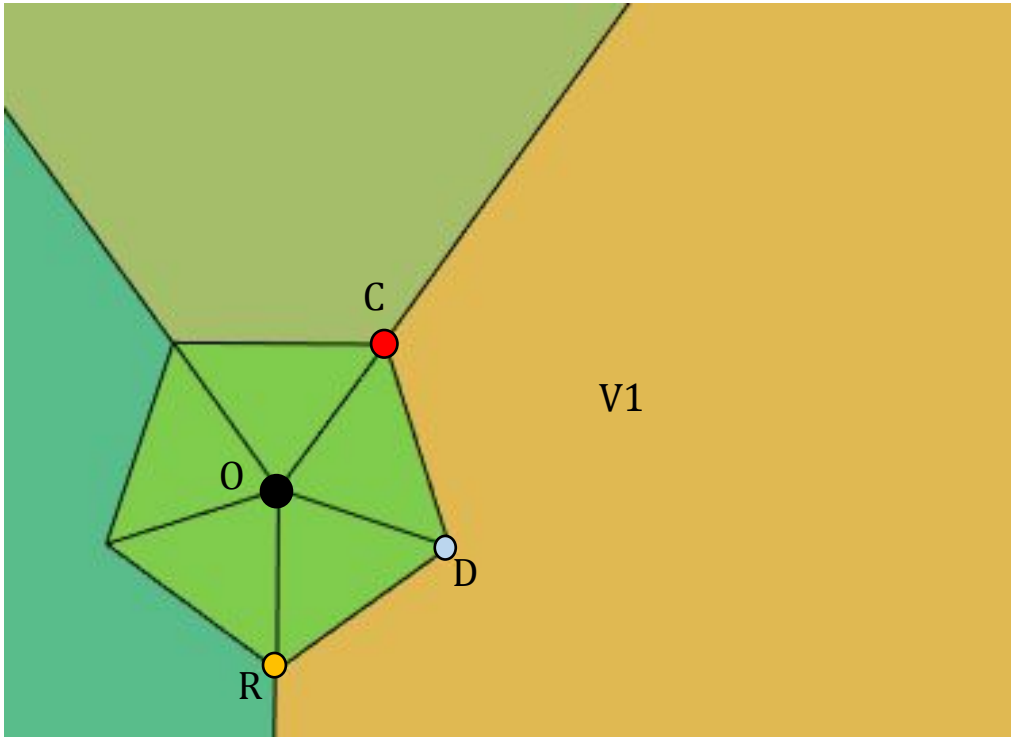


The ICONverter : *Generating Array for the Preliminary Diamond*

Initialize Columns and Rows

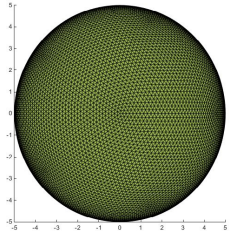


ICON geographic coordinate pool



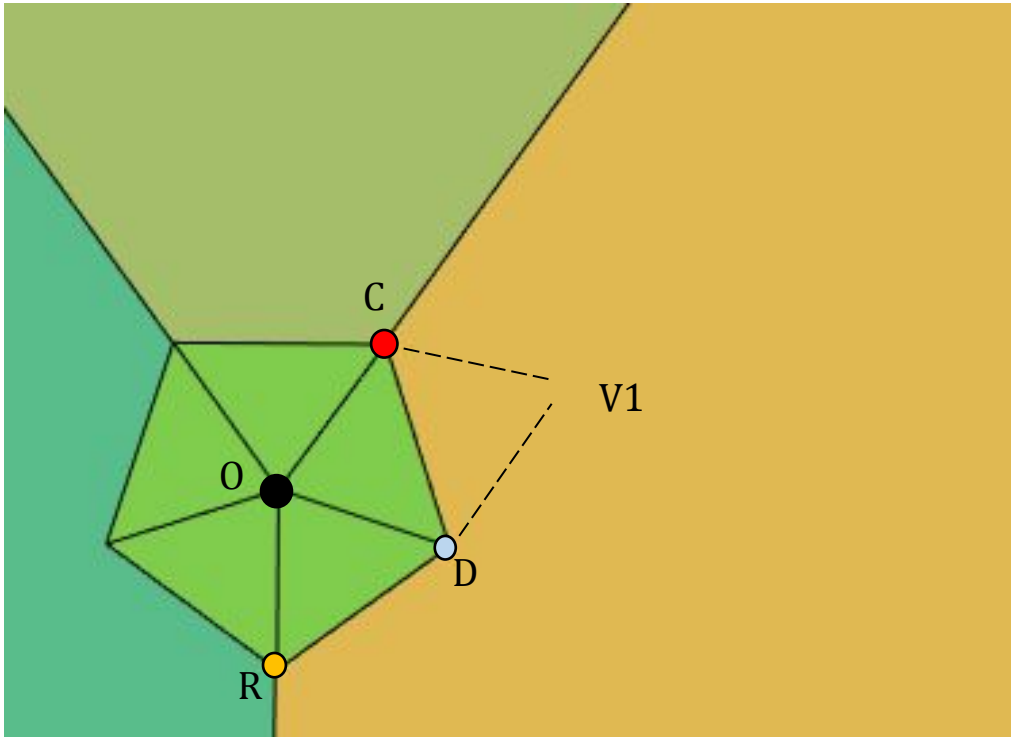
The ICONverter : *Generating Array for the Preliminary Diamond*

Initialize Columns and Rows



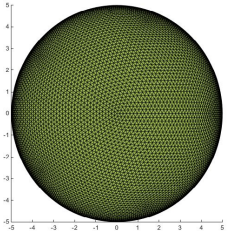
ICON geographic coordinate pool

O	C	D

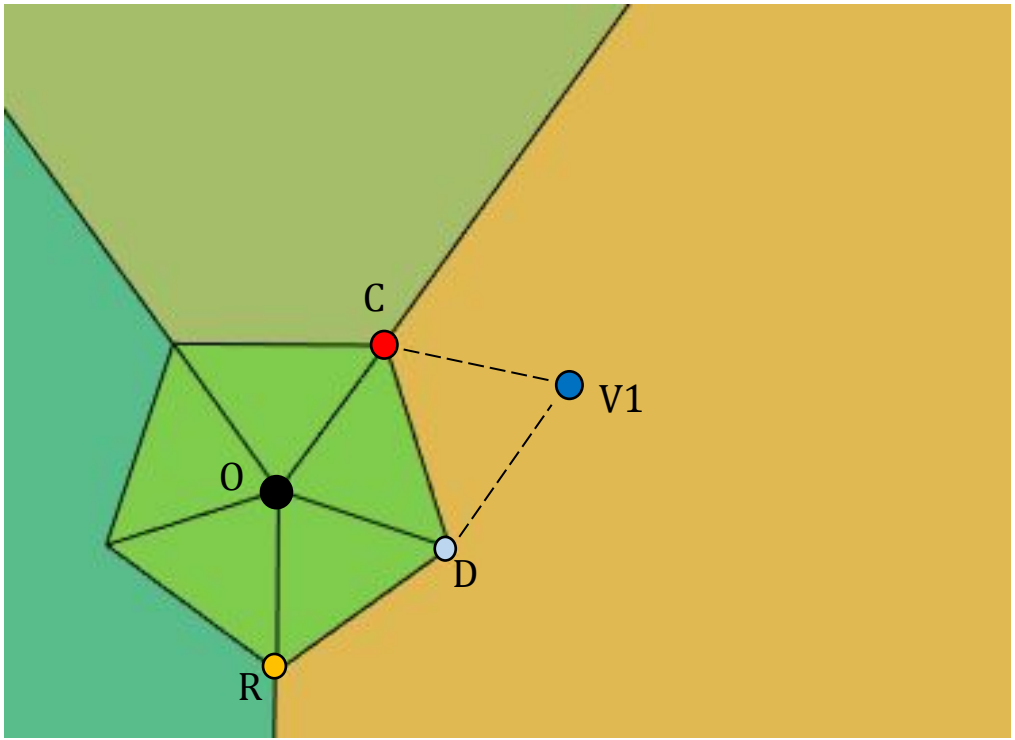


The ICONverter : *Generating Array for the Preliminary Diamond*

Initialize Columns and Rows

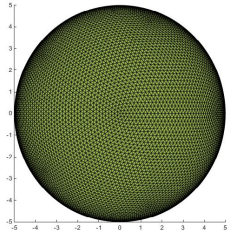


ICON geographic coordinate pool

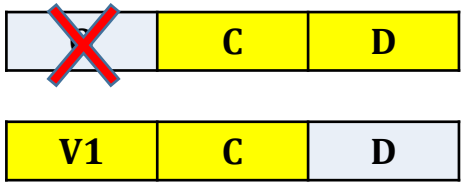


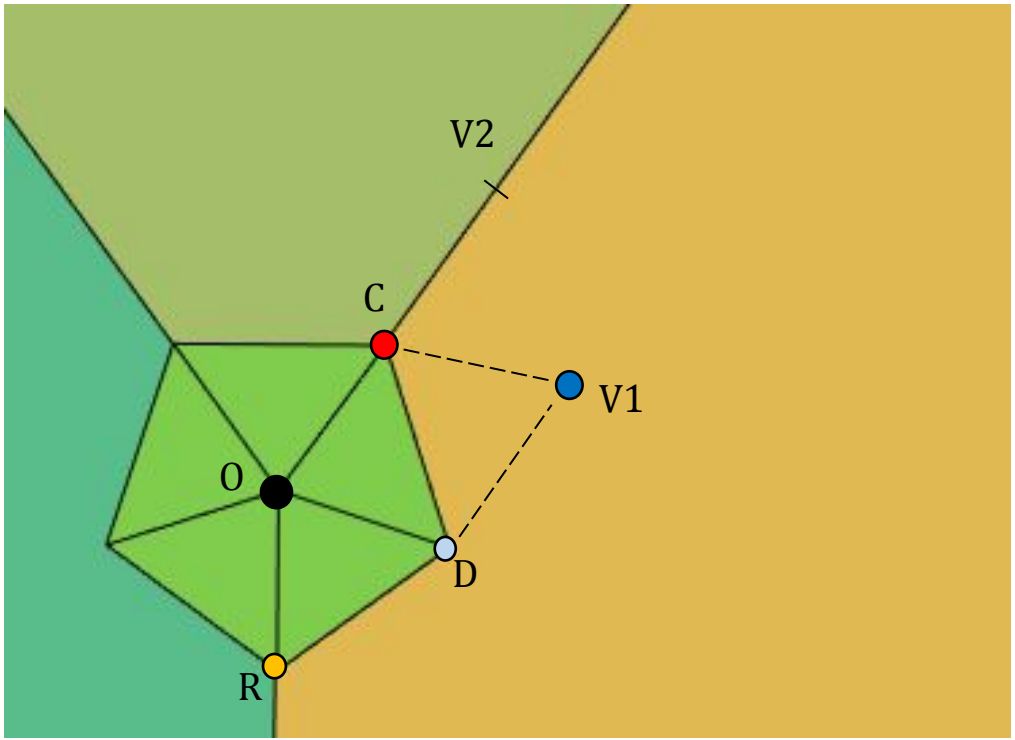
The ICONverter : *Generating Array for the Preliminary Diamond*

Initialize Columns and Rows



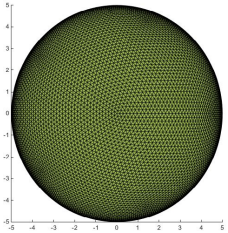
ICON geographic coordinate pool



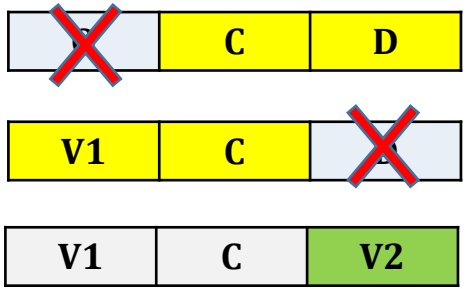


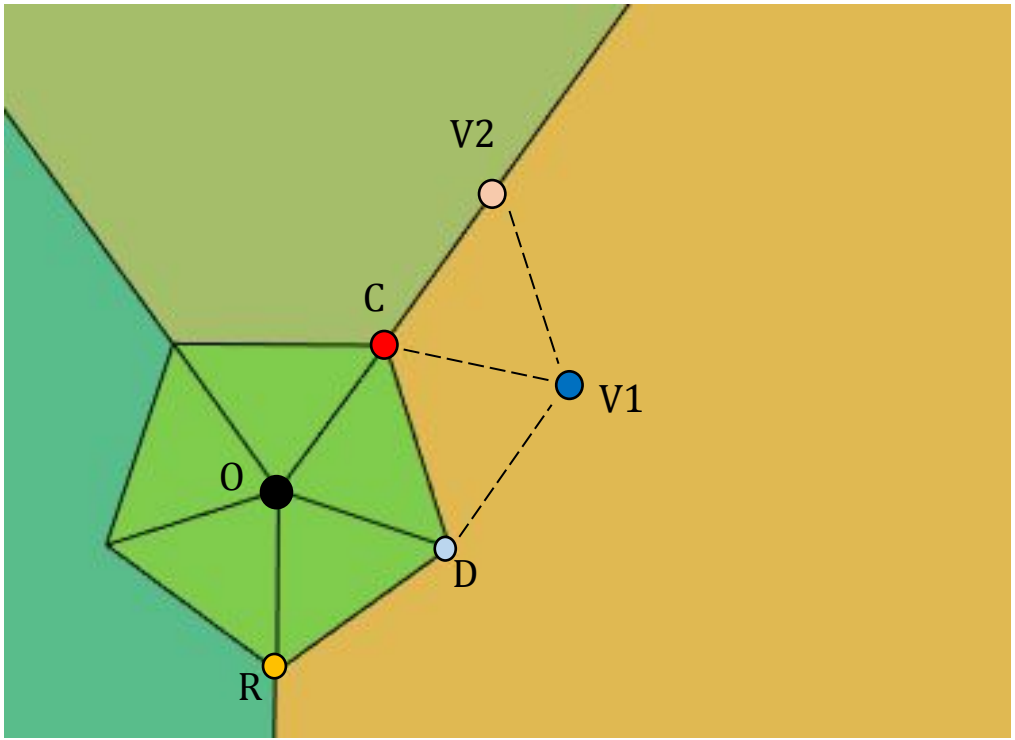
The ICONverter : *Generating Array for the Preliminary Diamond*

Initialize Columns and Rows



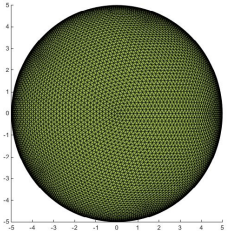
ICON geographic coordinate pool



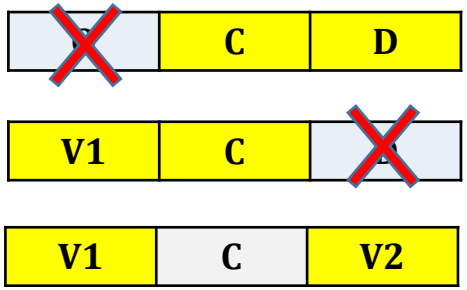


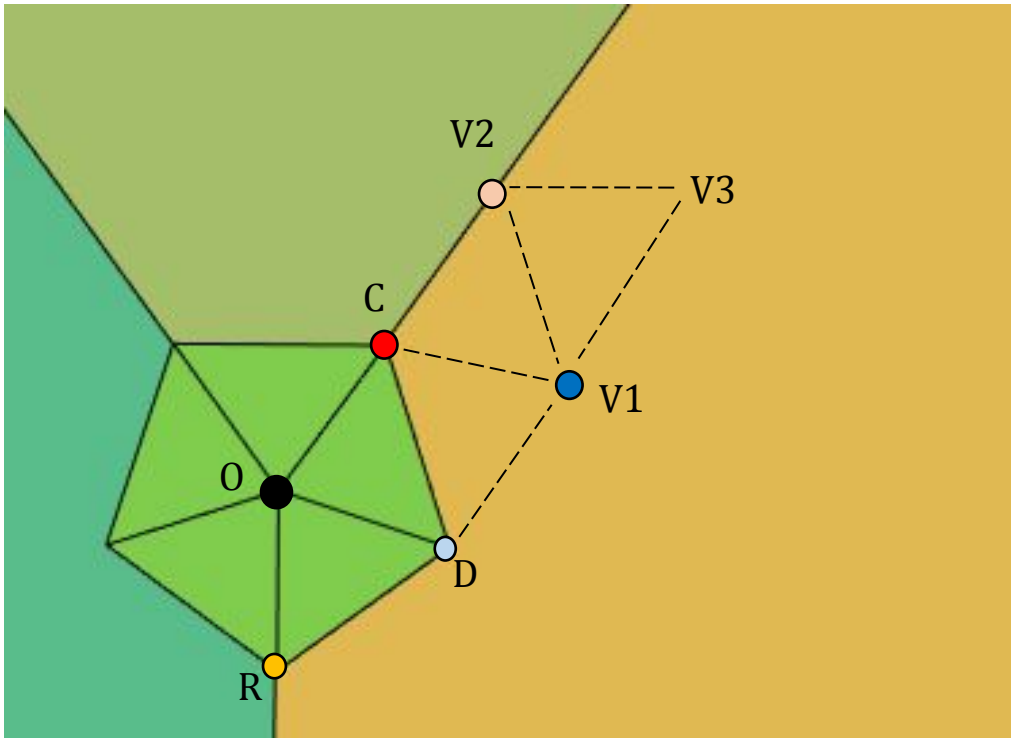
The ICONverter : *Generating Array for the Preliminary Diamond*

Initialize Columns and Rows



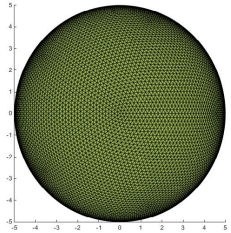
ICON geographic coordinate pool



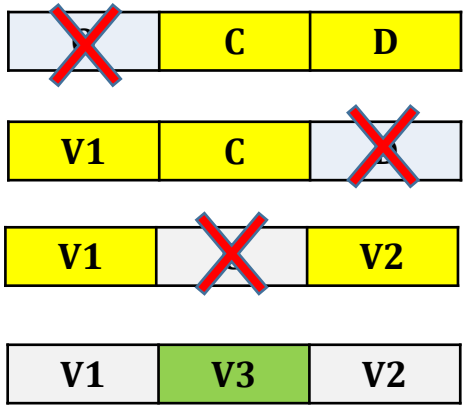


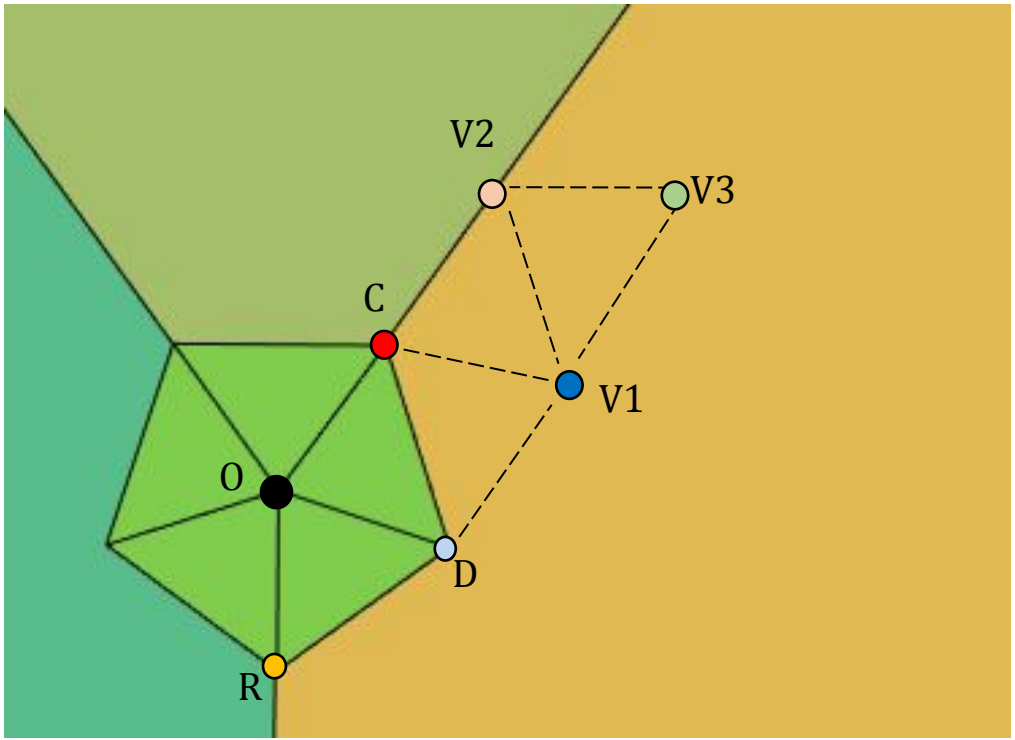
The ICONverter : *Generating Array for the Preliminary Diamond*

Initialize Columns and Rows



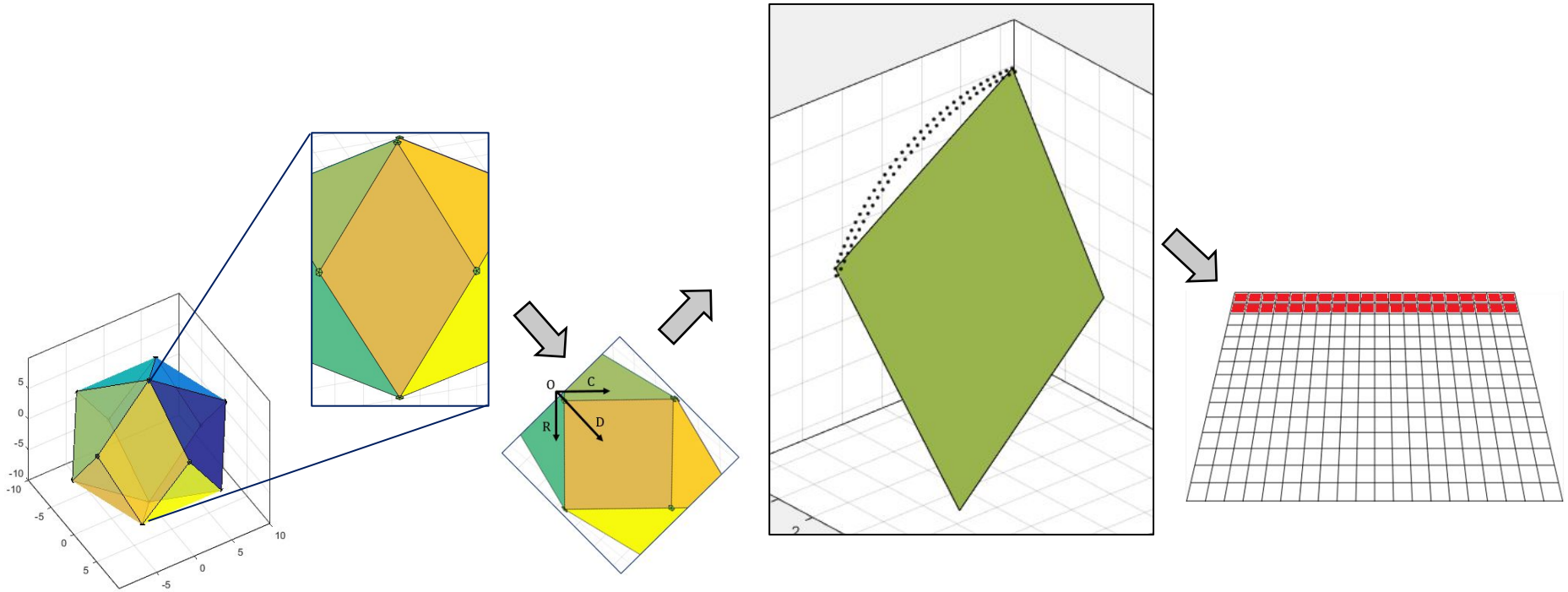
ICON geographic coordinate pool





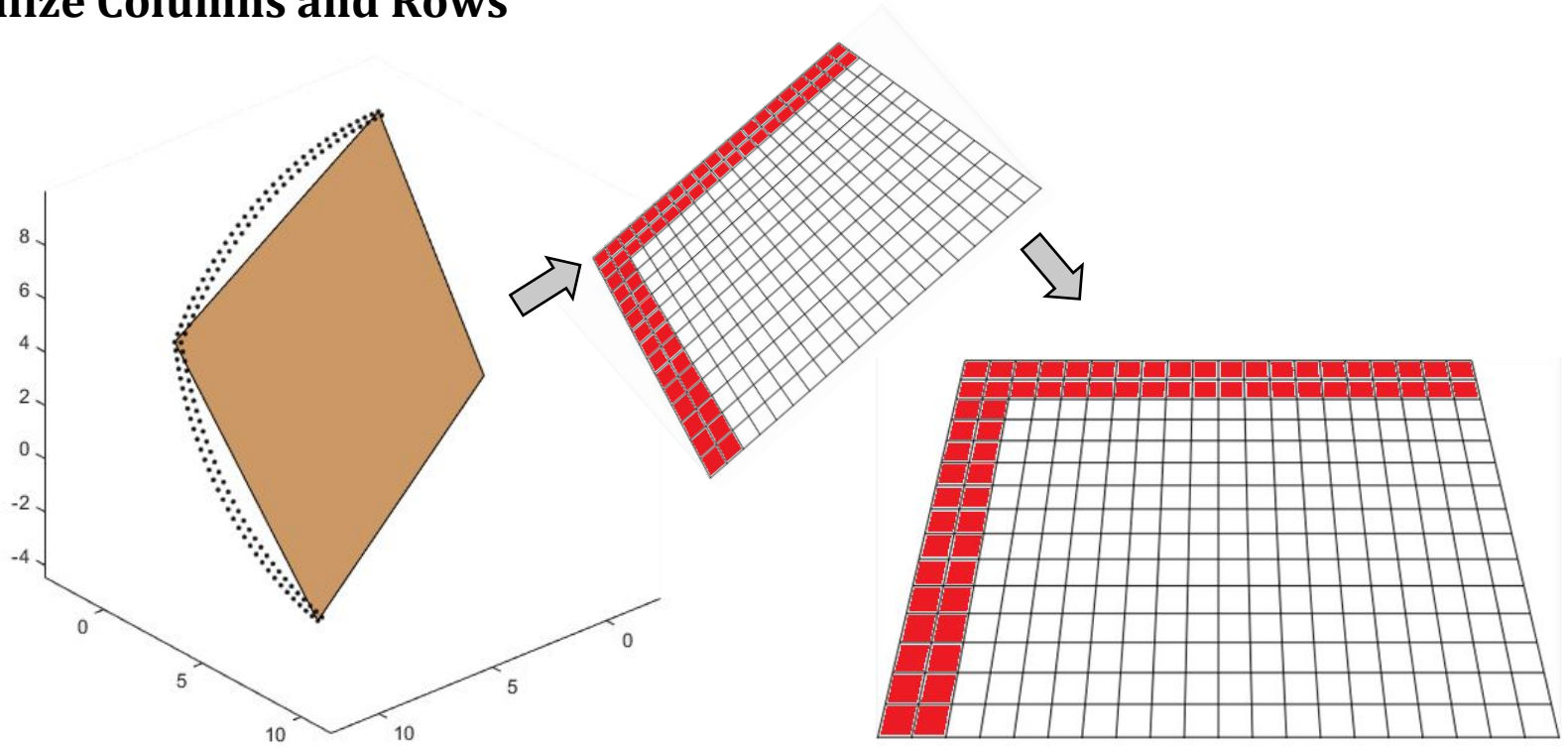
The ICONverter : *Generating Array for the Preliminary Diamond*

Initialize Columns and Rows



The ICONverter : *Generating Array for the Preliminary Diamond*

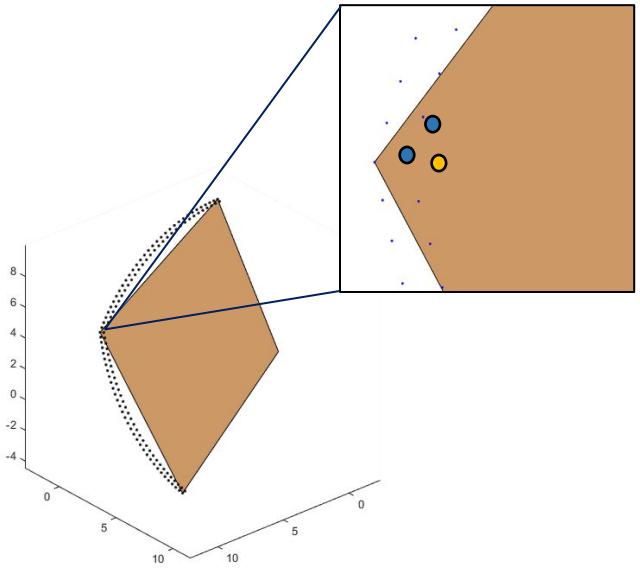
Initialize Columns and Rows

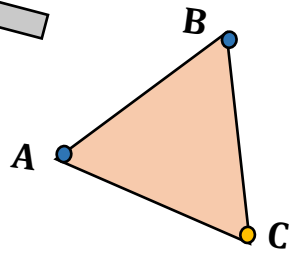


The ICONverter : *Generating Array for the Preliminary Diamond*

Filling up the entire array

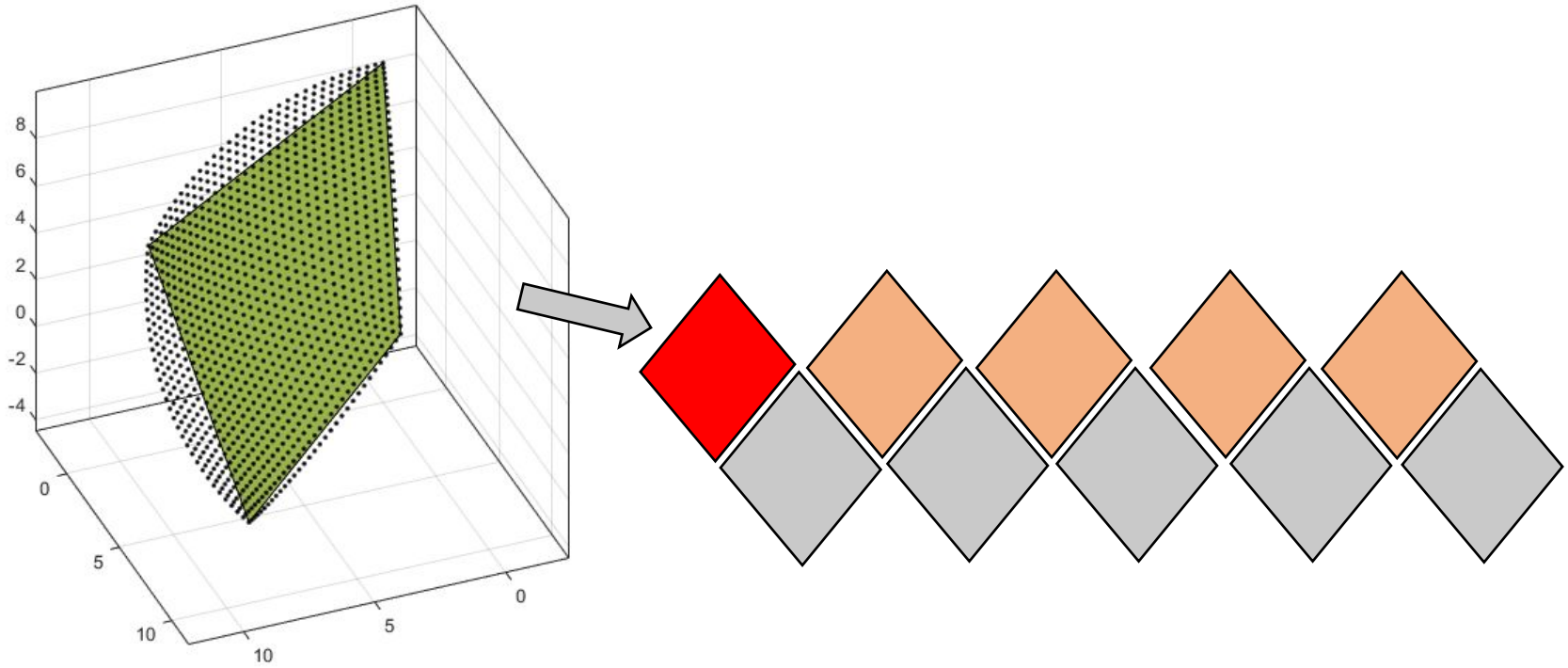
- To find vertices at (i, j) position of the array, we can use (i-1, j) and (i-1, j-1)
- Search in the variable pool for a vertex of a triangle which has two vertices **A** and **B** and is not already in the **array**



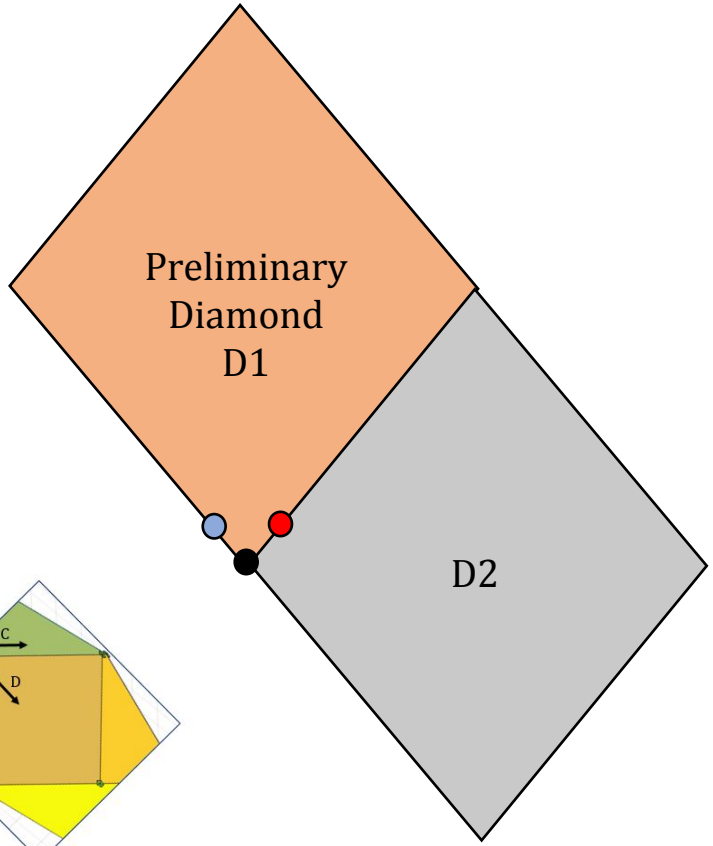


	A (i-1, j-1)	B (i-1, j)	
		C (i, j)	

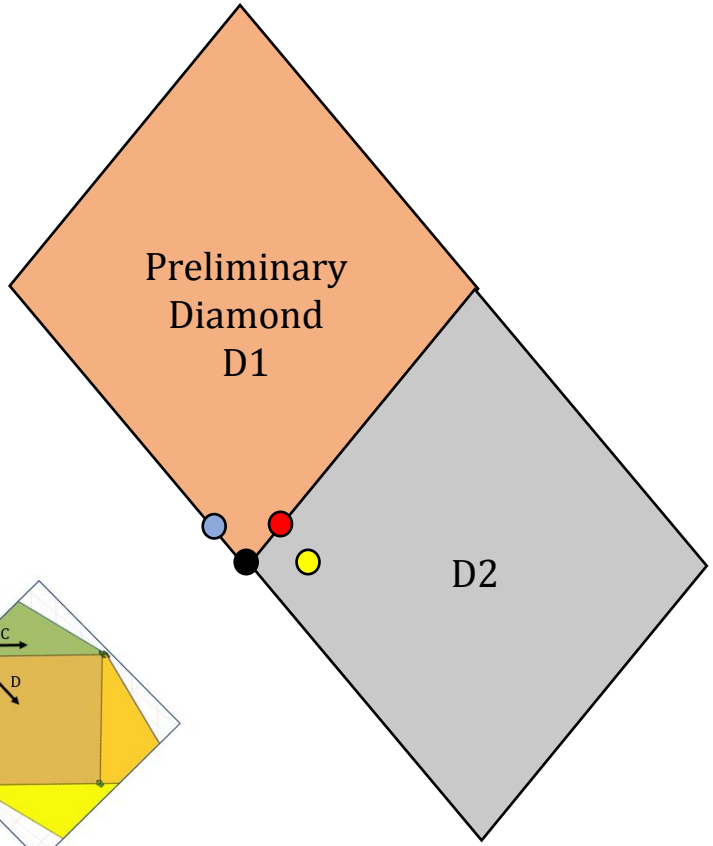
The ICONverter : *Generating Array for Other Diamonds*

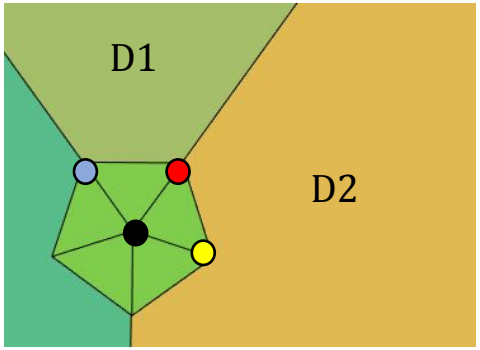


The ICONverter : *Generating Array for Other Diamonds*



The ICONverter : *Generating Array for Other Diamonds*





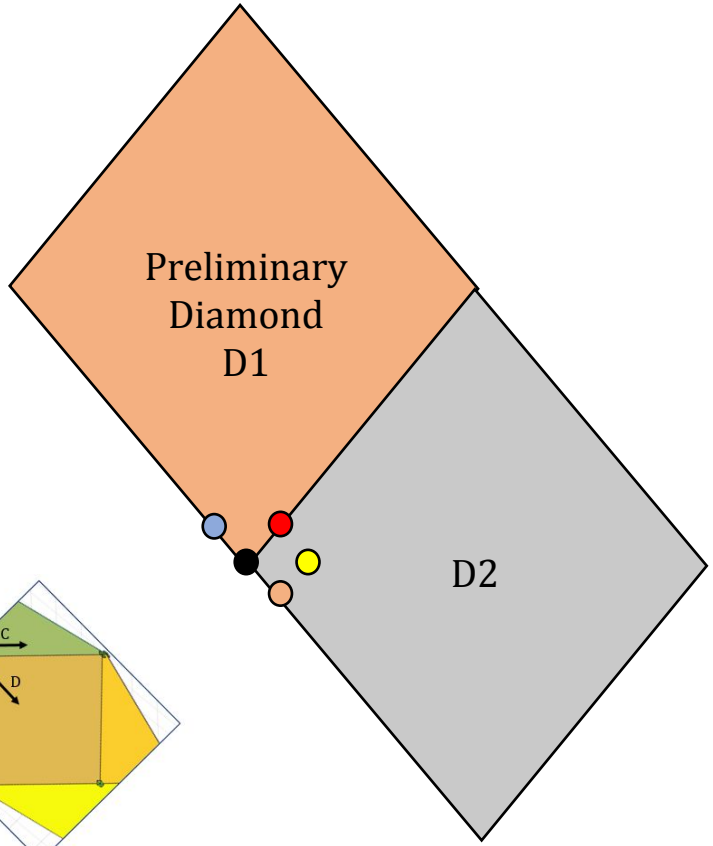
We have,

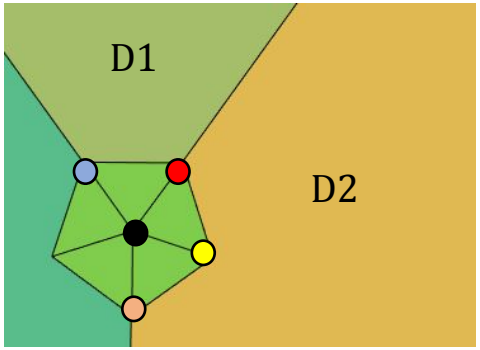
column vertex

origin vertex

We can find diagonal vertex (*not in D1*)

The ICONverter : *Generating Array for Other Diamonds*





We have,

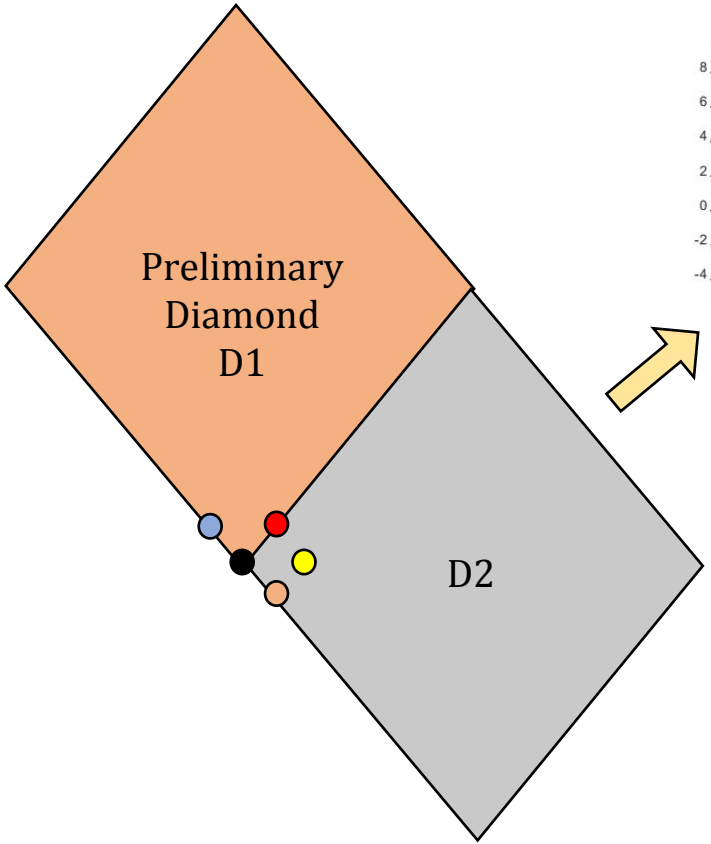
column vertex

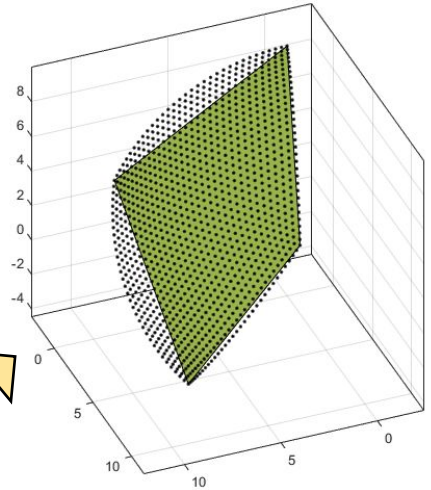
origin vertex

diagonal vertex

We can find row vertex

The ICONverter : *Generating Array for Other Diamonds*





We have,

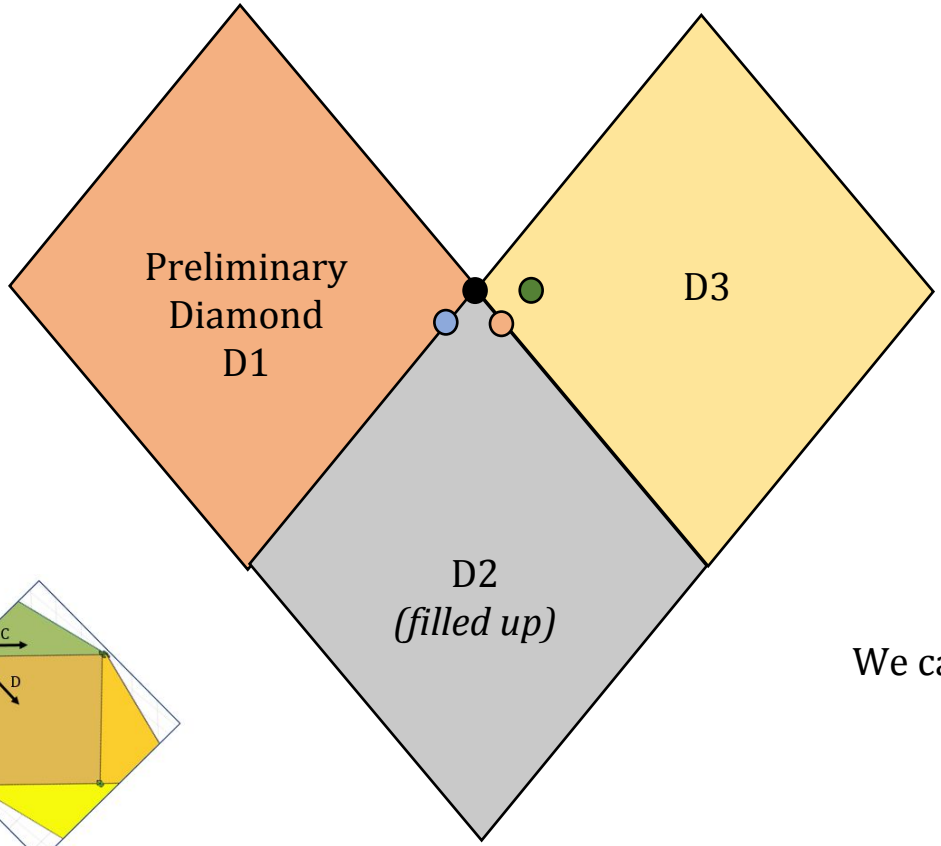
column vertex

origin vertex

diagonal vertex

We can find row vertex

The ICONverter : *Generating Array for Other Diamonds*



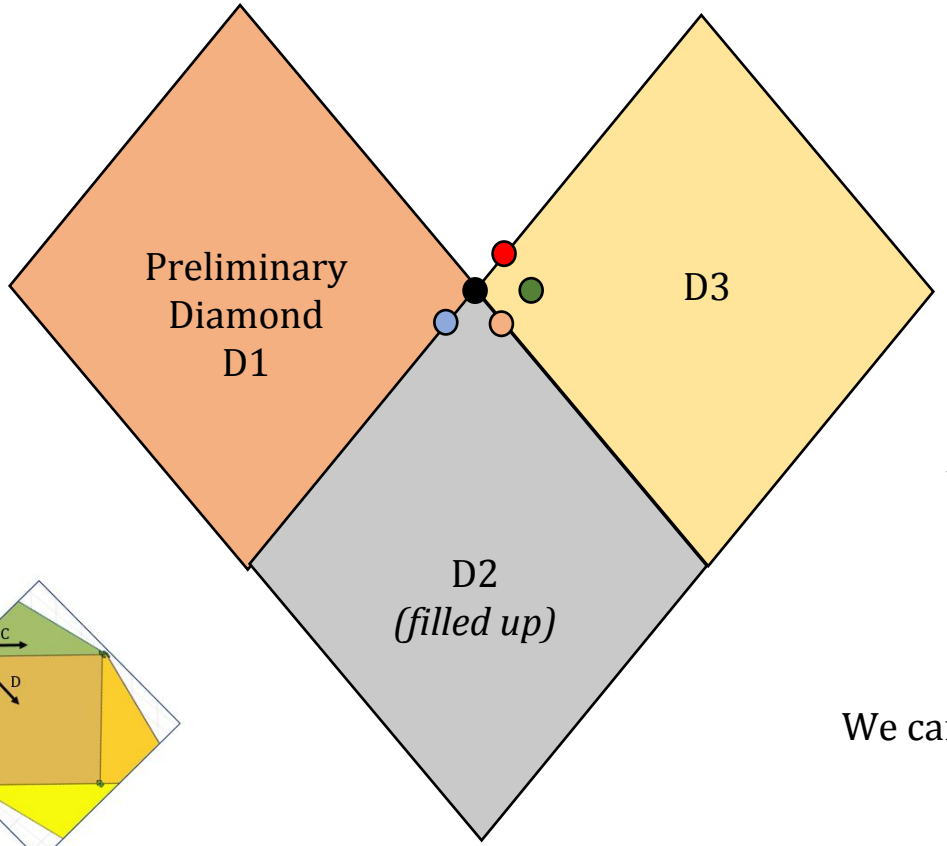
We have,

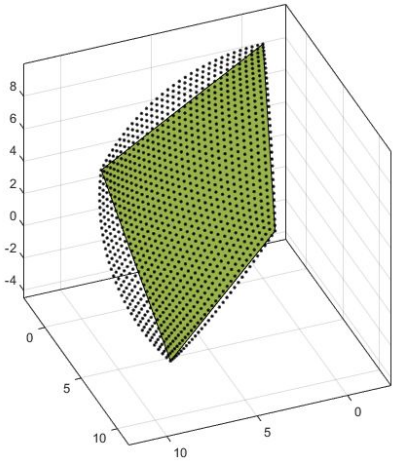
origin vertex ●

row vertex ●

We can find diagonal vertex ● *(not in D2* ● *)*

The ICONverter : *Generating Array for Other Diamonds*





We have,

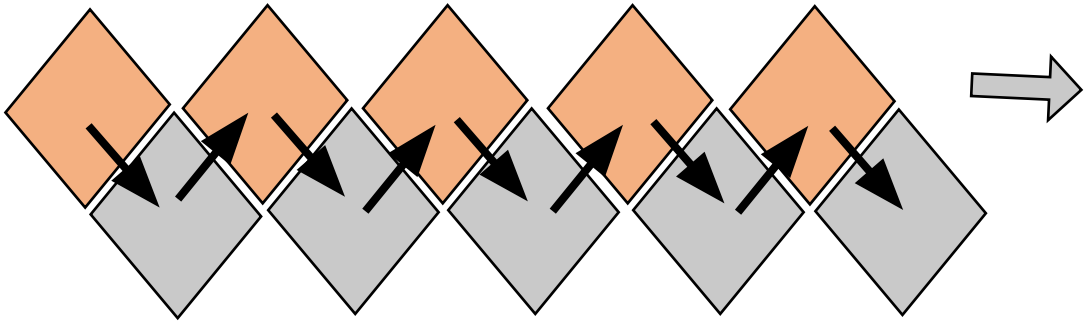
origin vertex

row vertex

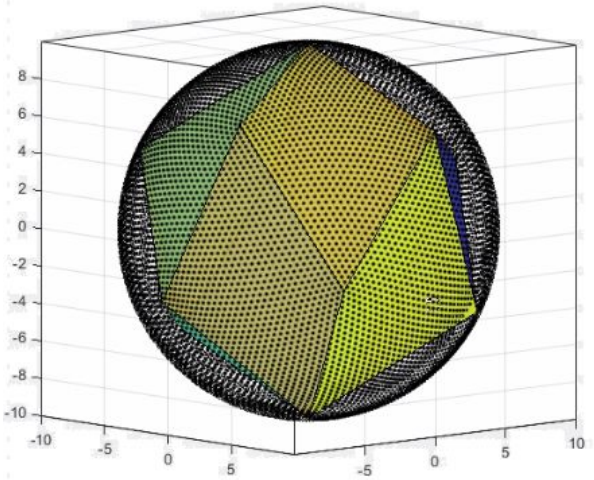
diagonal vertex

We can find column vertex

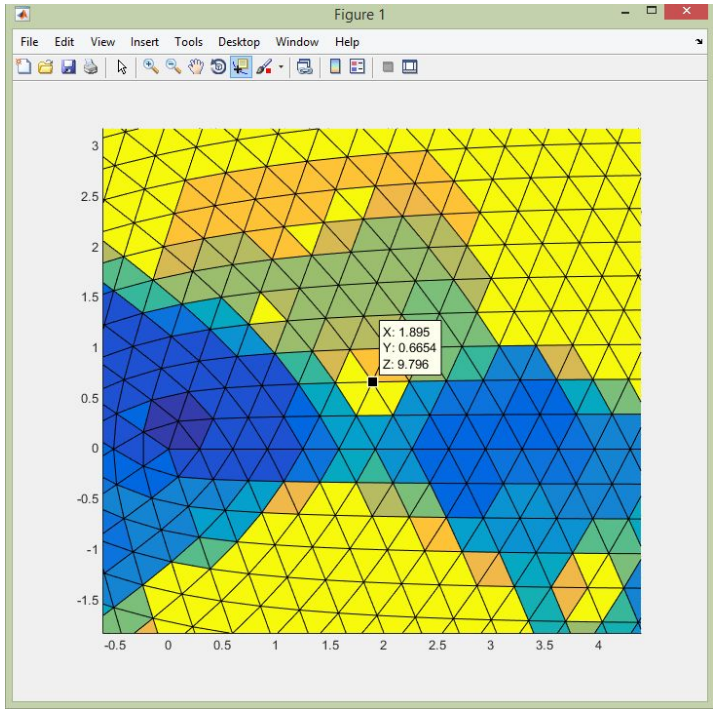
The ICONverter : *All the Arrays*

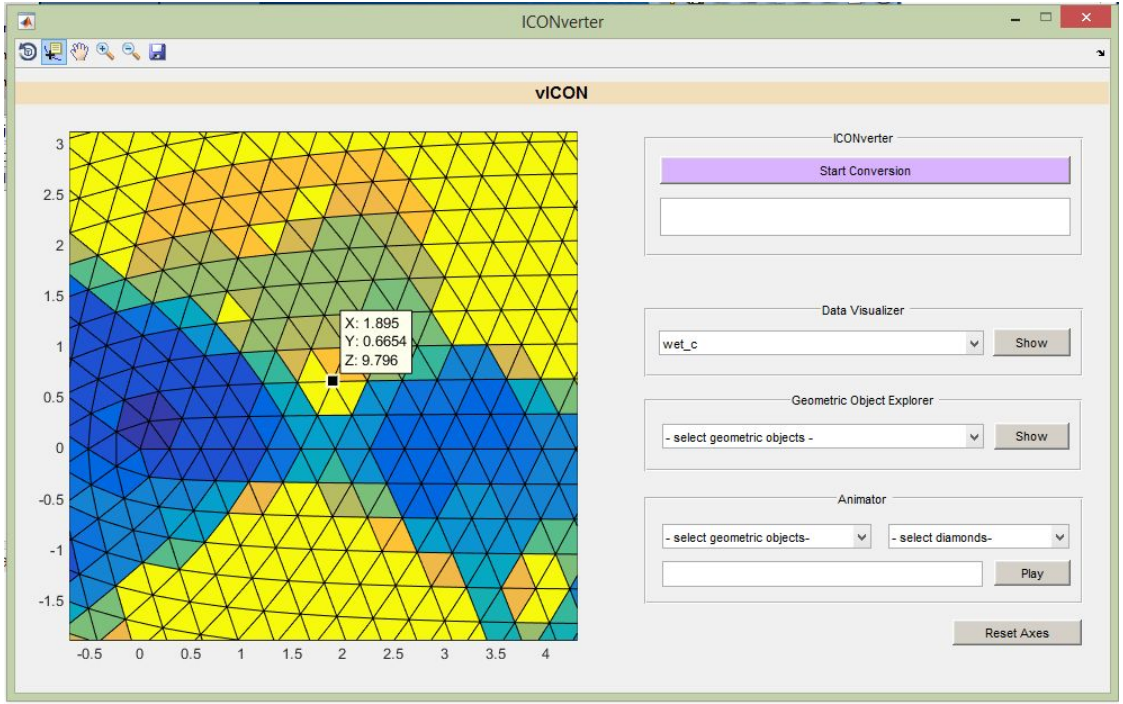


Processing all the Diamonds



The ICONverter : *Validation*





THANK YOU