What's In This Polyhedron?

by

Robert W. Gray
180-4 Poplar St.
Rochester, NY 14620

1. Introduction

In June of 1998 Lynnclaire Dennis and members of the Sequoia Symposium showed me a 120 triangular faced polyhedron that was important to her because she experienced this "watery" polyhedron during a near death experience. I was able to identify this polyhedron from my study of R. Buckminster Fuller's work. Fuller[1] described how the sphere can be broken into 120 LCD spherical triangles, which leads to a 120 triangular faced convex polyhedron that seemed to match the polyhedron she experienced. However, it was not until I understood that the polyhedron that she was describing was mixed, that is concave/convex (the polyhedron had bumps), that I began to explore other possibilities.

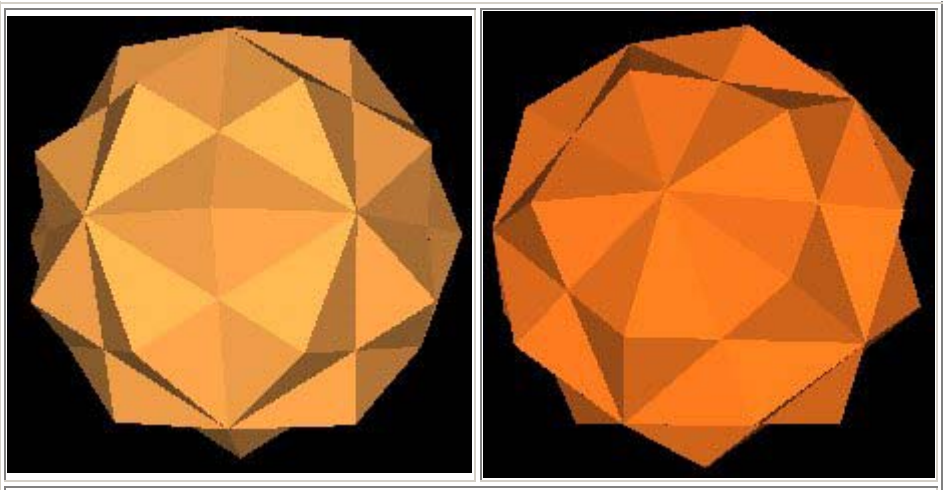
This article presents the current information I have developed for Lynnclaire's 120 "bubble" Polyhedron. It is my hope that this dynamic structure, with all its amazing relations and dynamics, turns out to be as important as Lynnclaire thinks it is. Whether it does or not, this is truely an amazing polyhedron worthy of further study.

See the Pattern web site at http://www.pattern.org for more information about Lynnclaire Dennis and the "Sequoia Team" investigating the information, implications and meaning of her near death experiences.

rwgray@rwgrayprojects.com

2. What It Looks Like

To start off, here are some illustrations of the 120 Polyhedron.



Figs. 2.1, 2.2 *The 144 Polyhedron*

This polyhedron has 120 triangular faces, 180 edges, and 62 vertices. To check that these numbers are consistent, we use Euler's formula for the topological features of polyhedra:

$$\text{Vertices} + \text{Faces} = \text{Edges} + 2$$
$$V + F = E + 2$$
$$62 + 120 = 180 + 2$$
$$182 = 182$$

3. Shared Vertex Inventory

It turns out that other polyhedra can be constructed using the 62 vertices of this polyhedron. For example, the next illustration shows the regular Dodecahedron positioned within the 120 Polyhedron.

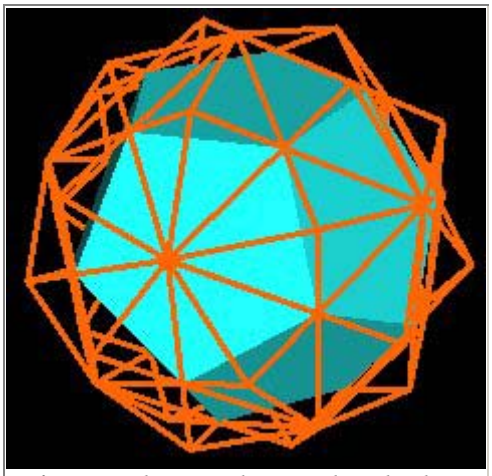


Fig 3.1 *The regular Dodecahedron within the 120 Polyhedron*

Notice that the regular Dodecahedron, which has 20 vertices, is positioned in such a way as to share 20 of the 120 Polyhedron's vertices.

In fact, ***all*** of the Platonic polyhdera can similarly be positioned within the 120 Polyhedron, sharing the 120 Polyhedron's vertices. This is cataloged in the next 4 illustrations.

rwgray@rwgrayprojects.com

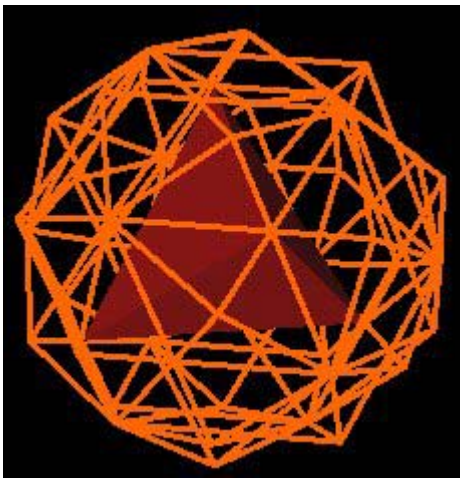


Fig 3.2 *Tetrahedron*

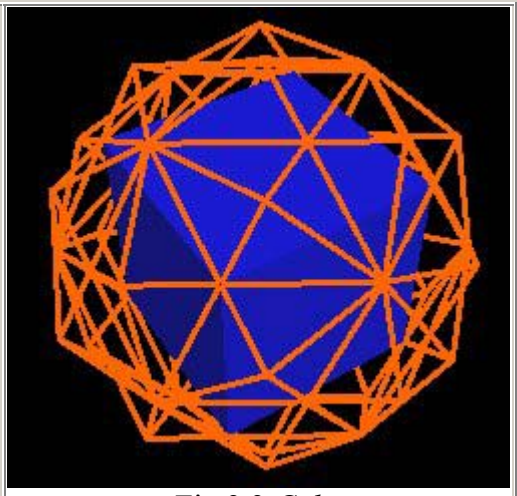


Fig 3.3 *Cube*

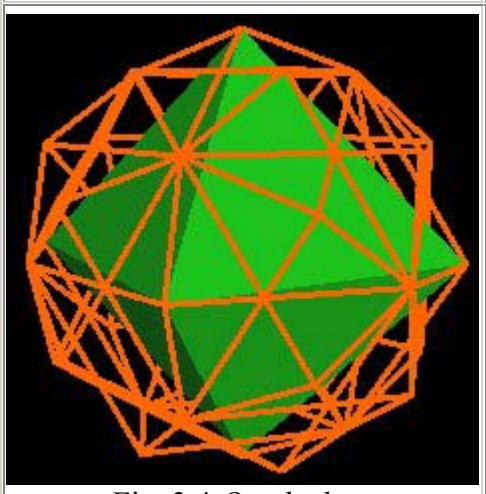


Fig. 3.4 *Octahedron*

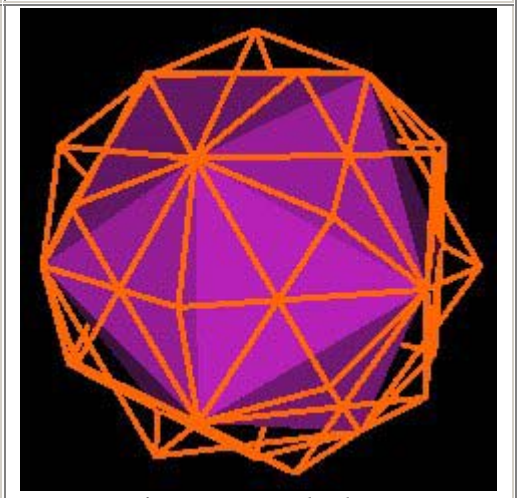


Fig. 3.5 *Icosahedron*

In Fuller's "Cosmic Hierarchy"[1], the regular Dodecahedron and Icosahedron do not share their vertices with any of the other polyhedra within the hierarchy. Their inability to "fit in" with the other polyhedra shows one of the limitations of Fuller's Cosmic Hierarchy. The problem arises, as will be shown below, because the model is too dependent on the concept of a single static vector matrix, which Fuller calls the Isotropic Vector Matrix (IVM), from which all polyhedra are to be measured. However, the polyhedra in the 120 Polyhedra arise from a ***single dynamic motion***, which Fuller was well aware of and for which he named the "Jitterbug" motion. (This will be shown below.)

rwgray@rwgrayprojects.com

As shown above, the 120 Polyhedron provides a way to include the regular Dodecahedron and the Icosahedron into a single polyhedral system. All the Platonic "solids" share their vertices with the 120 Polyhedron.

This is not the end to the inventory of polyhedra to be found in the 120 Polyhedron. There also occurs the rhombic Dodecahedron and the rhombic Triacontahedron.

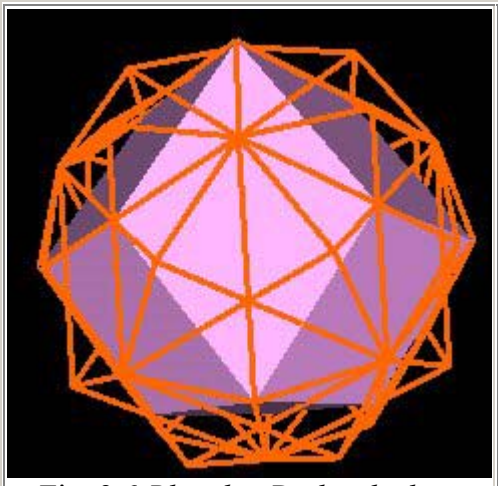


Fig. 3.6 *Rhombic Dodecahedron*

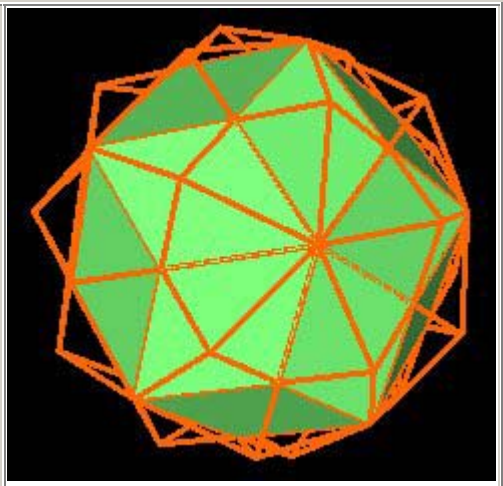


Fig. 3.7 *Rhombic Triacontahedron*

If we consider not only the 62 vertices of the 120 Polyhedron but also the intersection points resulting from the above inventory of polyhedra's edges intersecting with one another, there are even more well known polyhedra to be seen.

In particular, from the intersection of the Octahedron and the Cube, the Cubeoctahedron is defined. Fuller called the Cubeoctahedron the Vector Equilibrium (VE).

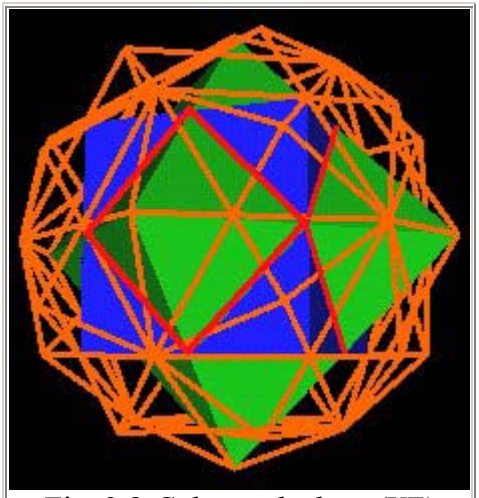


Fig. 3.8 *Cubeoctahedron (VE)*

rwgray@rwgrayprojects.com

4. Beginnings Of A Dynamic Creation

The cube and the regular Dodecahedron share the same vertices of the 120 Polyhedron, as shown in the next sequence of illustrations.

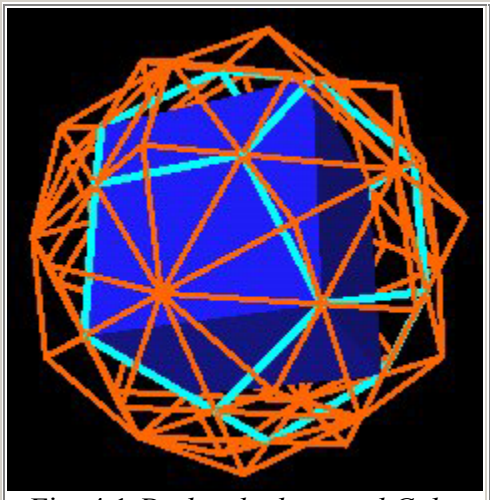


Fig. 4.1 *Dodecahedron and Cube in the 120 Polyhedron*

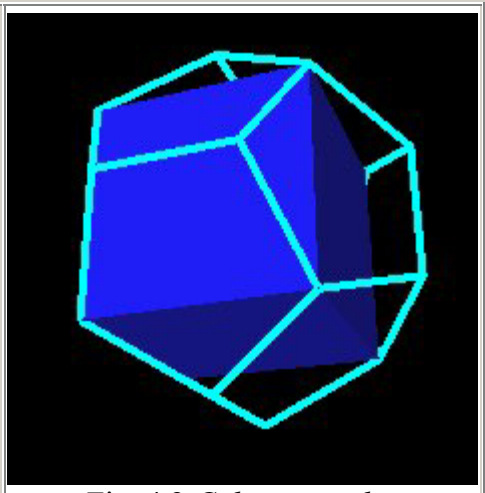


Fig. 4.2 *Cube in regular Dodecahedron*

In fact, we find that 5 different Cubes can be arranged inside the Dodecahedron. Each of the Dodecahedron's vertices is shared with 2 Cube vertices.

Fig. 4.3 *5 Cubes in Dodecahedron*

This led me to consider the possibility of 4 cubes rotating in such a way as to "spring forth" from a single 5th Cube and freezing into the regular Dodecahedron position.

Consider a single cube. It has 8 vertices.

We can use opposite vertices as axes of rotations for the cube. There are then 4 vertex-to-opposite-vertex rotation axes.

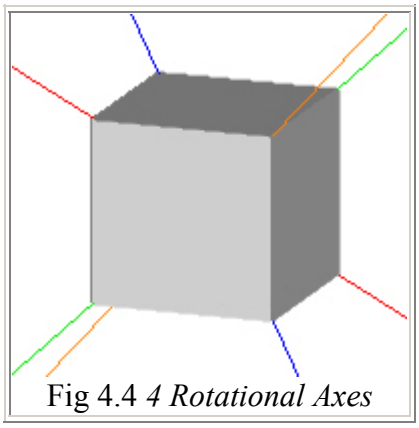
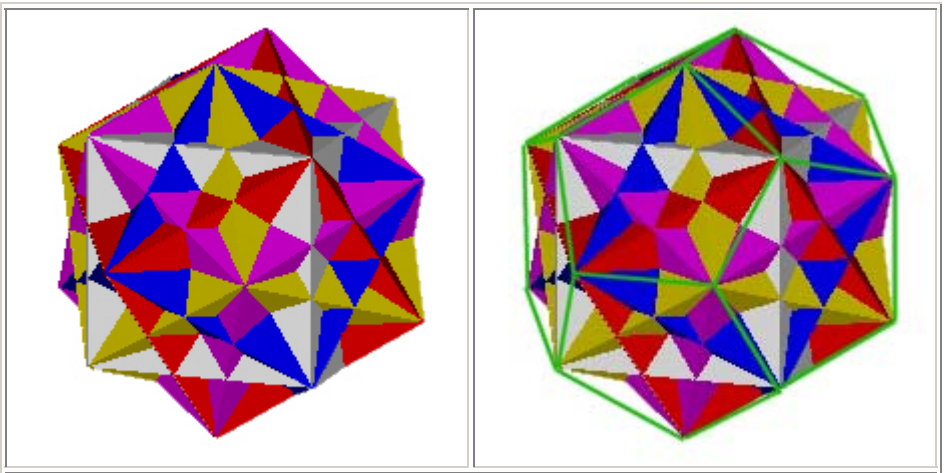


Fig 4.4 *4 Rotational Axes*

If we place 5 cubes, of the same size, in the same position, and leave one of the 5 cubes fixed, we can use these 4 rotation axes to rotate the other 4 cubes. We assign each of the 4 cubes to a different rotation axis.

When we rotate each of the 4 cubes by an angular amount of 44.47751219°(approximately), clockwise around the 1st axis, counter clockwise around the 2nd, clockwise around the 3rd, counter clockwise around the 4th, we find that the vertices of all the cubes define a regular Dodecahedron. Details for the calculation of this angle are given at http://www.rwgrayprojects.com/Lynn/NCH/nch/angle.html

rwgray@rwgrayprojects.com



Figs. 4.5, 4.6 *Regular Dodecahedron defined by rotating 4 of the 5 cubes*

Notice that the cube edges define pentagrams in the Dodecahedron's pentagonal faces.

Fig. 4.7 *Cubes' edges define 12 Pentagrams*

Here is a gif movie showing the 4 rotating cubes, with one cube (gray in color) stationary. (See web version for animation.)

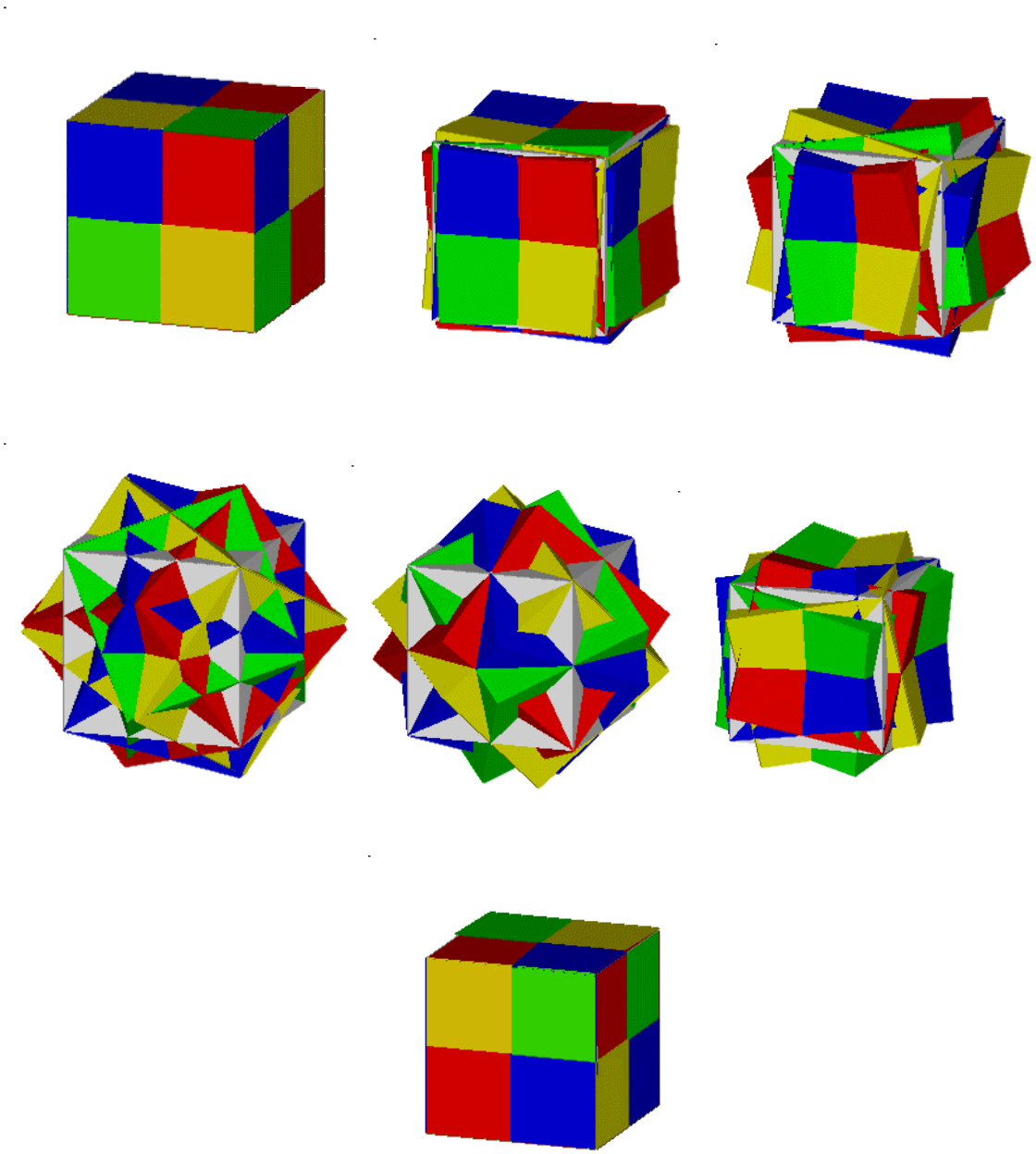
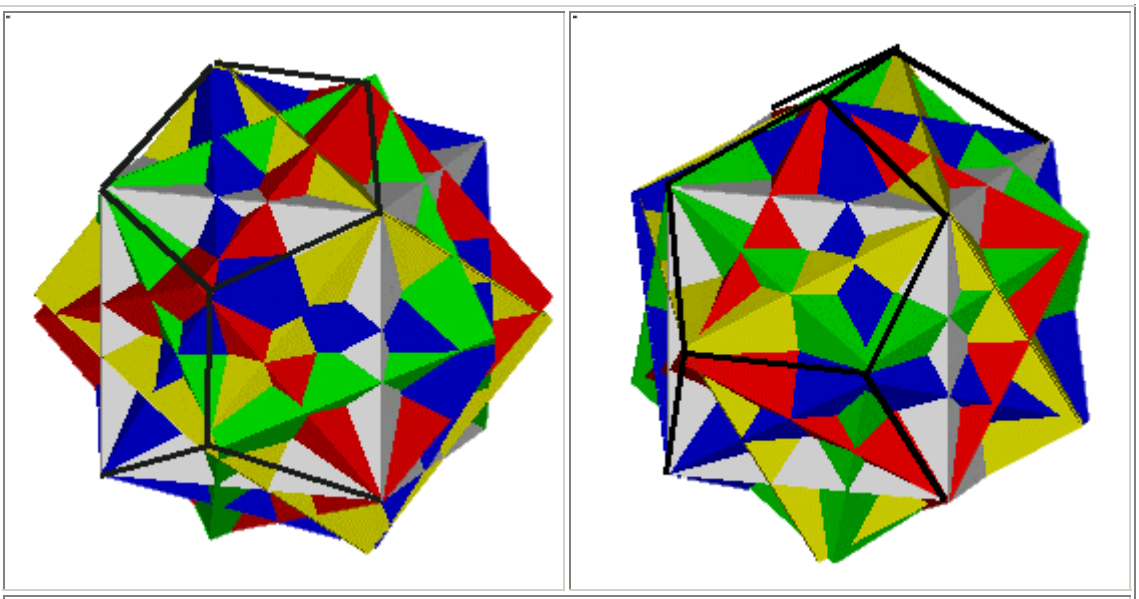


Fig. 4.8 *Animation of 4 rotating Cubes, fifth Cube stationary*

An interesting feature of this dynamic definition of the regular Dodecahedron, using the rotation of the Cubes, is that 2 orientations of the Dodecahedron appear. Here are 2 frames from the above gif movie in which I have added some black lines to outline some of the Dodecahedron's edges to show the 2 orientations of the regular Dodecahedron.



Figs. 4.9, 4.10 *Two orientations of the Dodecahedron*

For each Cube an Octahedron is can be defined. The Octahedron is the "dual" polyhedron of the Cube. A "dual" of a polyhedron is (loosely) defined by replacing all vertices with faces and all faces by vertices. So, a cube having 8 vertices and 6 faces will have a dual polyhedron consisting of 8 faces and 6 vertices. This is the Octahedron.

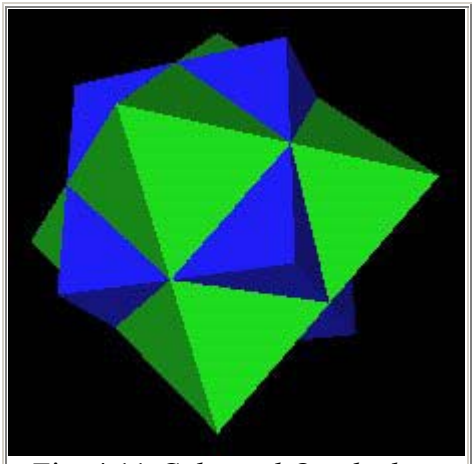
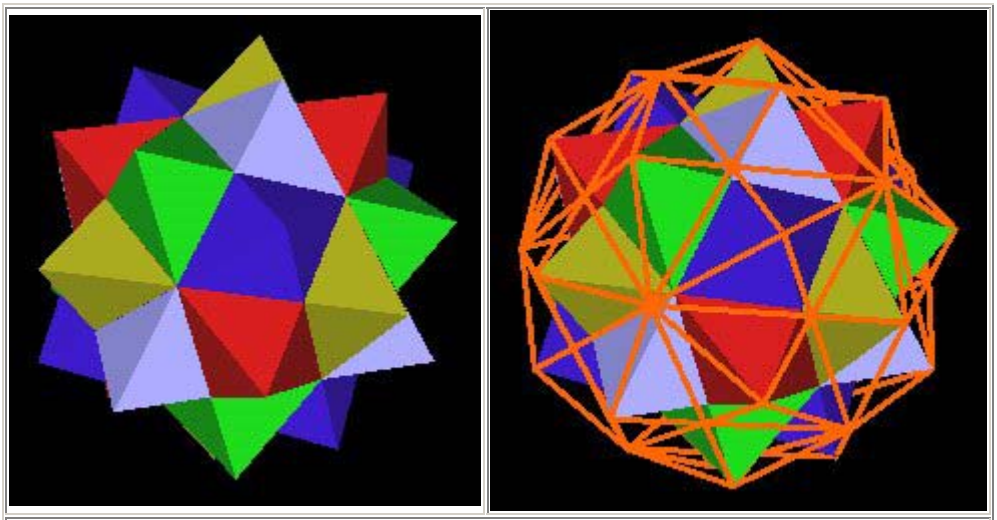


Fig. 4.11 *Cube and Octahedron*

So, there are 5 Octahedra for the 5 Cubes we have been considering.

If we now rotate the 4 Octahedra with the 4 Cubes (keeping one Cube and its Octahedron fix) by the angular amount of 44.47751219°(approximately) mentioned above, both the Octahedra and Cube vertices will coincide with vertices of the 120 Polyhedron.



Figs. 4.12, 4.13 *30 vertices of 5 rotated Octahedra*

The 5 Cube's vertices coincide with the regular Dodecahedron's 20 vertices, which coincide with 20 vertices of the 120 Polyhedron. The 5 Octahedra have a total of 5x6=30 vertices and these coincide with 30 vertices of the 120

rwgray@rwgrayprojects.com

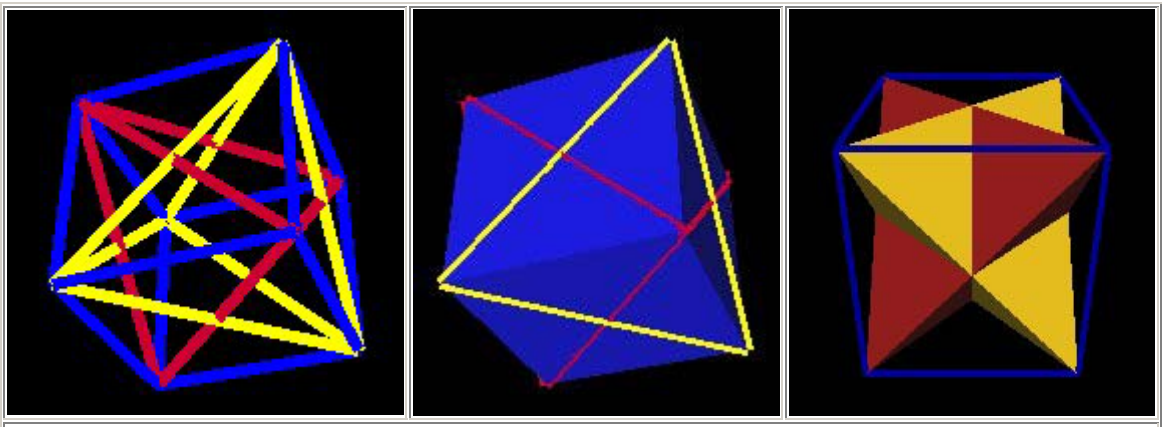
Polyhedron. This leaves 62-20-30=12 vertices of the 120 Polyhedron unassigned. These 12 vertices correspond to the 12 vertices of an Icosahedron, which is the dual of the regular Dodecahedron.

Before continuing with this rotational development of the 120 Polyhedron, let us take a look at another polyhedron associated with the Cube and the regular Dodecahedron.

rwgray@rwgrayprojects.com

5. Asymmetry In Symmetry

As Fuller often pointed out, the Cube consists of 2 intersecting Tetrahedron.



Figs. 5.1, 5.2, 5.3 *Cube defined by 2 intersecting Tetrahedra*

When the regular Dodecahedron is defined by the 5 cubes, as shown above, each of the Dodecahedron's vertices coincided with 2 different Cube's vertices. By considering the Tetrahedra defined by these Cubes, we can eliminate this redundancy. That is, we can define the regular Dodecahedron by 5 intersecting Tetrahedra in such a way as to assign a single Tetrahedron's vertex to a single Dodecahedron vertex. In this way a "minimal" constuction definition for the regular Dodecahedron is achieved.

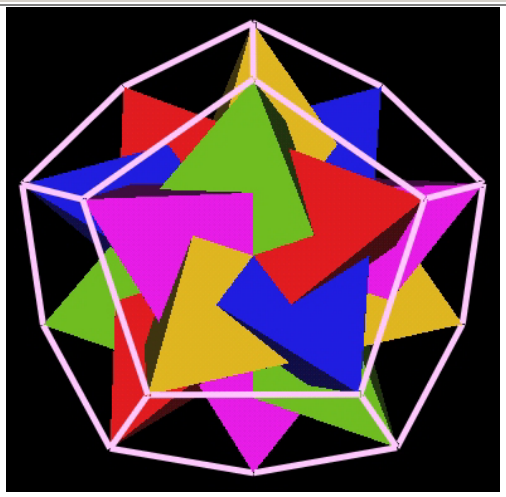


Fig. 5.4 *Regular Dodecahedron defined by 5 intersecting Tetrahedra*

However, there are actually 2 ways to assign the Tetrahedra within the regular Dodecahedron. In one case, the Tetrahedra intersect each other to define what appears to be a "clockwise layering" and in the other case, the 5 Tetrahedra intersect each other in a "counter-clockwise layering" manner.

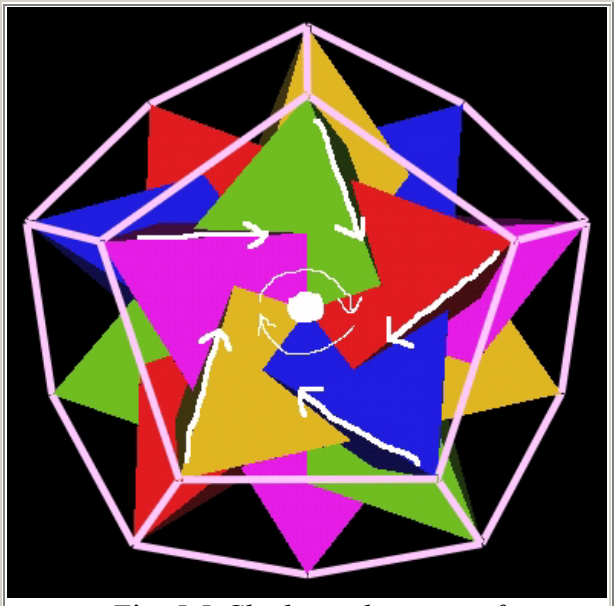


Fig. 5.5 *Clockwise layering of the 5 Tetrahedra*

This gives the 120 Polyhedron either a clockwise or counter-clockwise orientation in its construction while remaining globally invariant with respect to its external vertex orientation.

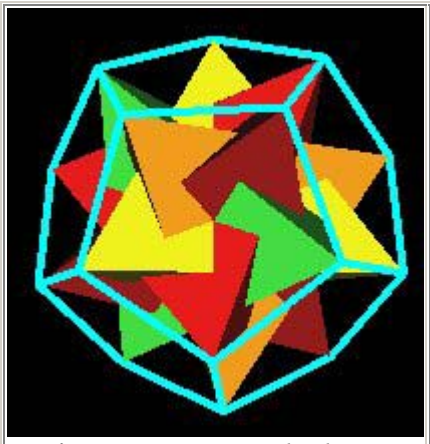


Fig. 5.6 *Counter Clockwise*

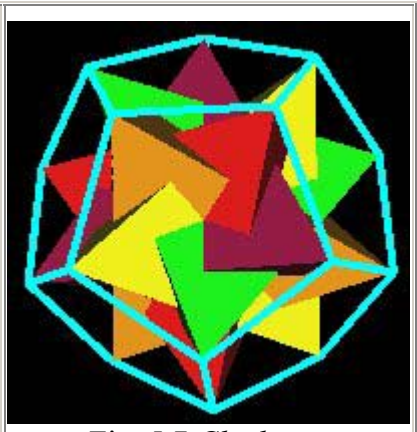


Fig. 5.7 *Clockwise*

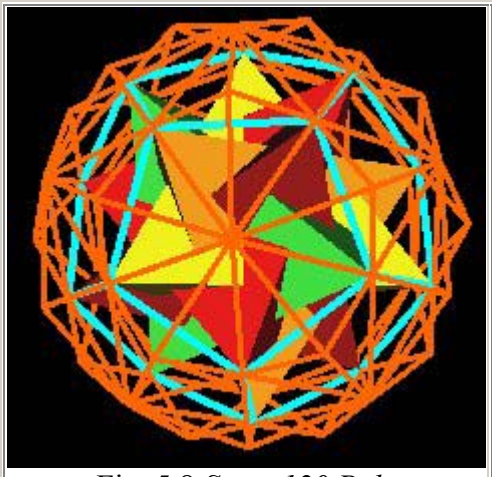


Fig. 5.8 *Same 120 Poly orientation*

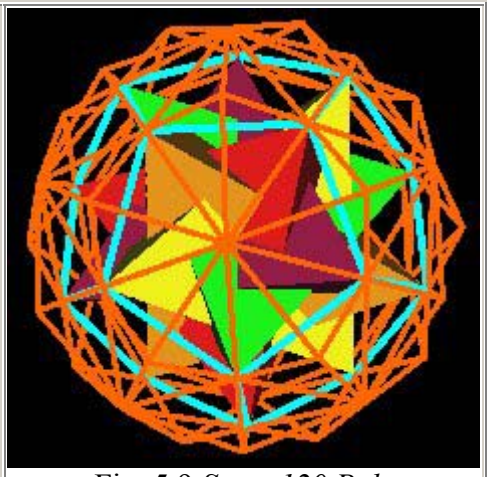


Fig. 5.9 *Same 120 Poly orientation*

6. The Jitterbug Does It All

With a Cubeoctahedron (VE) constructed out of sticks and rubber vertices, Fuller often demonstrated what he called the "Jitterbug" motion. The Jitterbug shows how the VE can fold up into an Octahedron as well as how an Octahedron can expand in the VE.

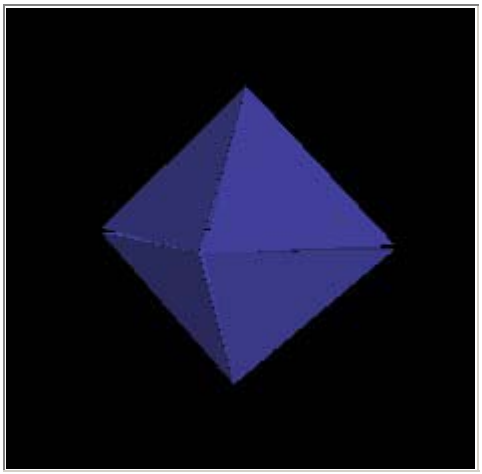


Fig. 6.1 *Octahedron position*

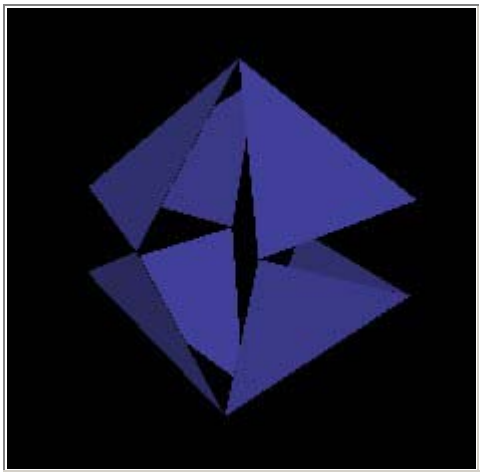


Fig. 6.2 *Jitterbug in motion*

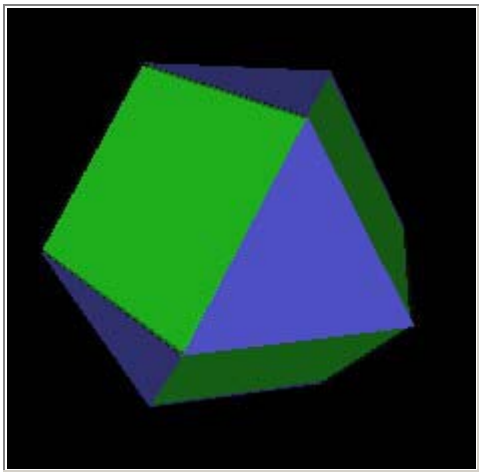


Fig. 6.3 *VE position*

The Jitterbug has 8 triangular faces. As these 8 faces rotate, they also move radially inward or outward from the center of volume along its 4 rotation axes.

rwgray@rwgrayprojects.com

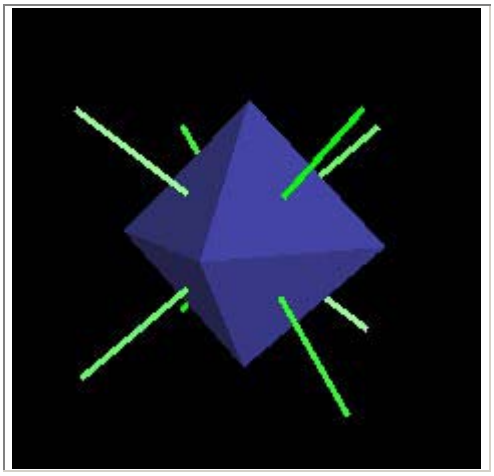


Fig. 6.4 *Motion along 4 rotation axes*

These are the same 4 rotation axes that we used to rotate the 4 cubes.

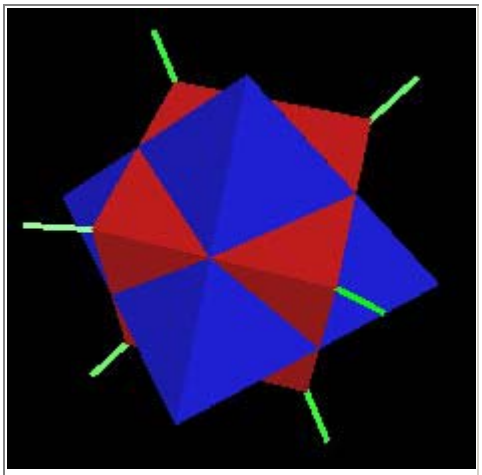


Fig. 6.5 *Jitterbug rotation axes*

Fuller pointed out that between the VE and the Octahedron positions, the Jitterbug will pass through an Icosahedron position.

rwgray@rwgrayprojects.com

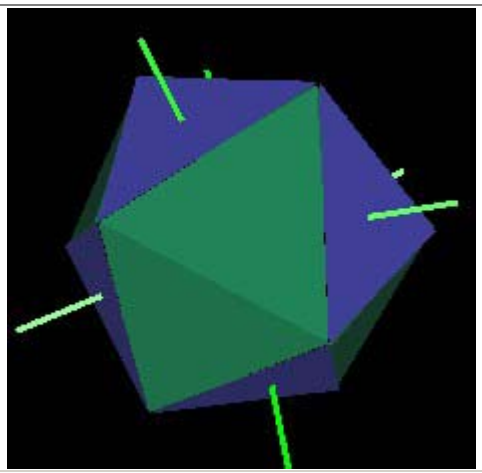


Fig. 6.6 *Jitterbug defines the Icosahedron*

If we allow the 8 rotating triangles to interpenatrate each other, then the Jitterbug can rotate and contract into two intersecting tetrahedra to define a cube.

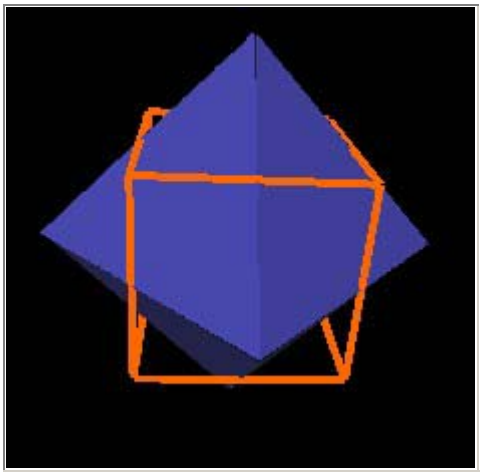


Fig. 6.7 *Jitterbug in Octahedron position*

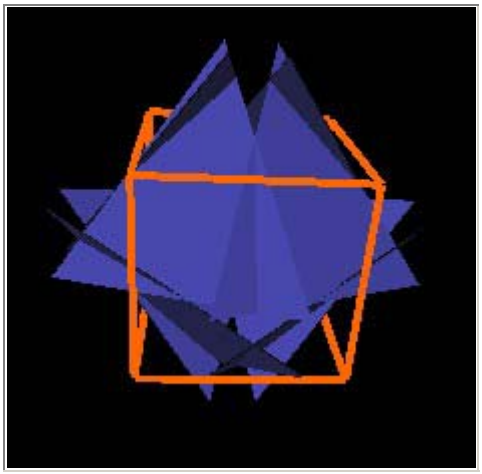


Fig. 6.8 *Jitterbug in motion*

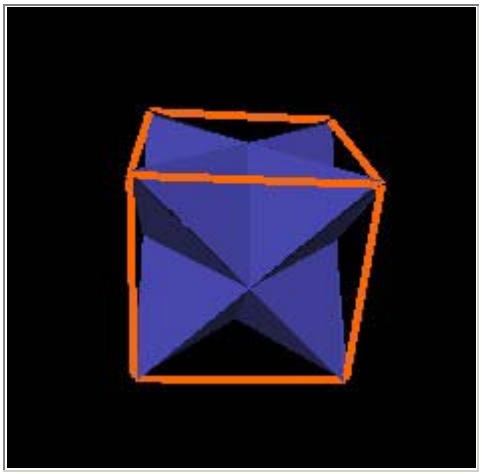


Fig. 6.9 *Jitterbug in Cube position*

Further rotation and contraction results in the definition of another Icosahedron.

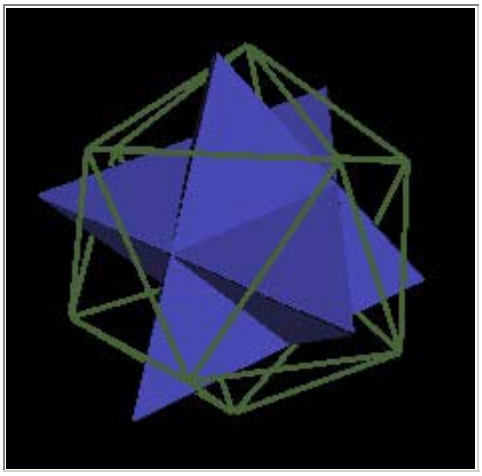


Fig. 6.10 *Jitterbug in Cube position*

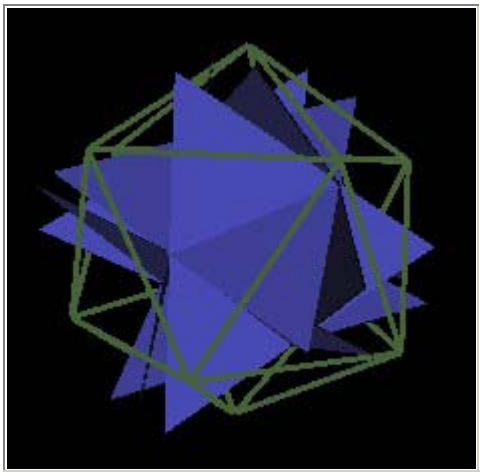


Fig. 6.11 *Jitterbug defines the Icosahedron*

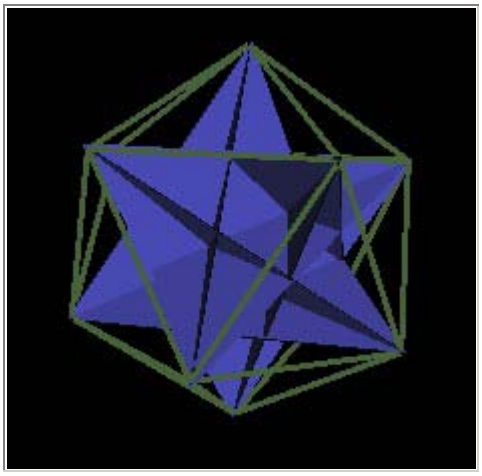


Fig. 6.12 *Jitterbug in Icosahedron position*

When the 120 Polyhedron is considered with all of its defining, internal polyhedra, many Jitterbugs can easily be identified. These Jitterbugs are not all in the same open position, nor of the same scale. Here is an illustration looking into the 120 Polyhedron through a regular Dodecahedron vertex. (The edges of the 120 Polyhedron are not shown.)

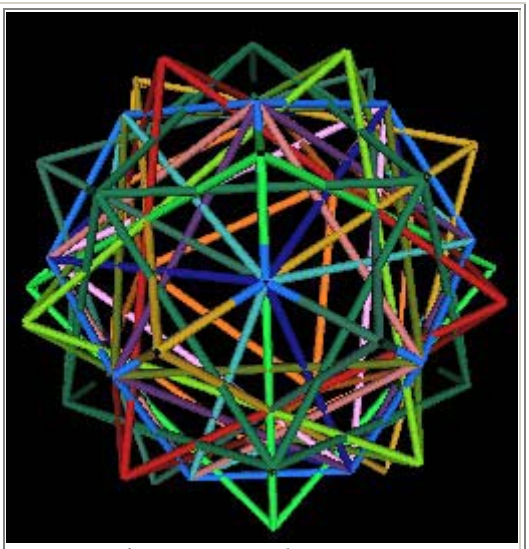


Fig. 6.13 *Looking into a Dodecahedron vertex*

In the next sequence of illustrations, I display various Jitterbugs by changing the associated polyhedron into a solid appearance. All of these Jitterbugs have the same face centered rotation axis passing through the regular Dodecahedron's vertex.

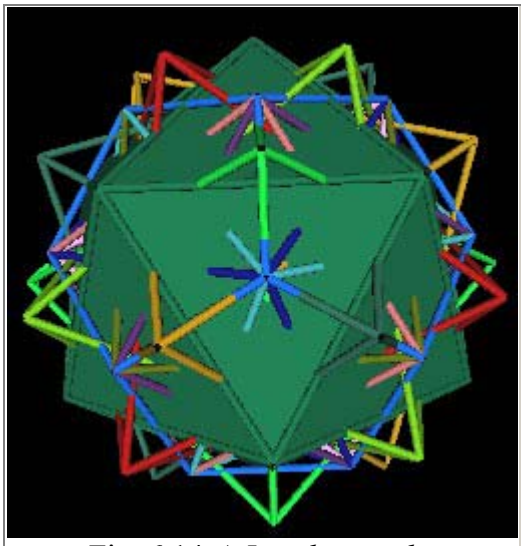


Fig. 6.14 *A Jitterbug in the Icosahedron position*

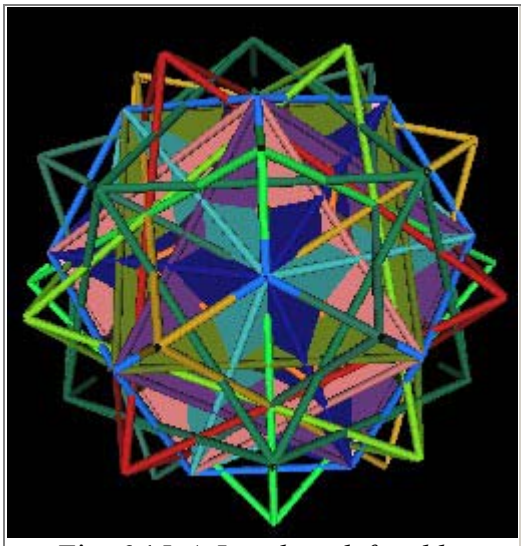


Fig. 6.15 *A Jitterbug defined by Cube edges*

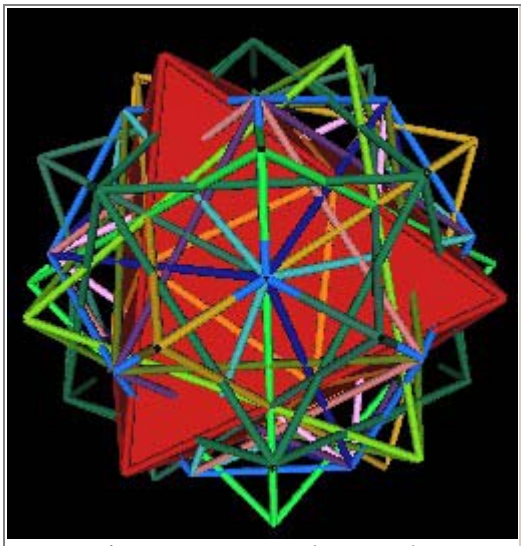


Fig. 6.16 *A Jitterbug in the Octahedron position*

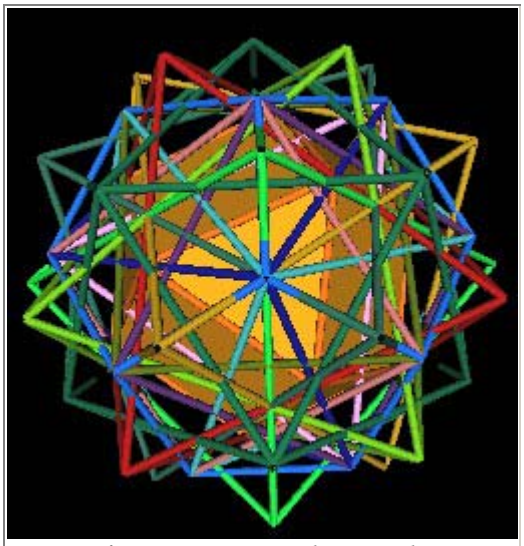


Fig. 6.17 *A Jitterbug in the VE position*

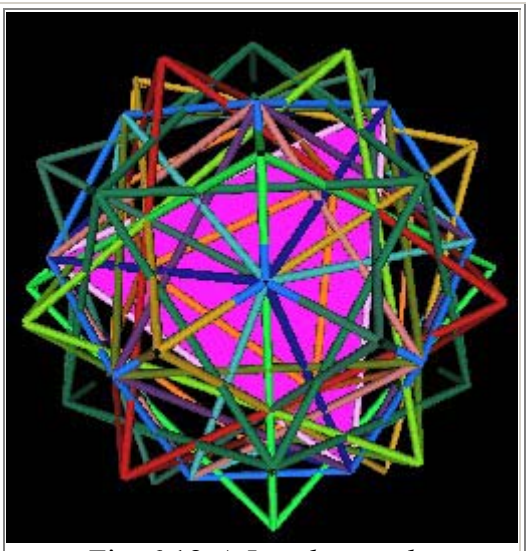
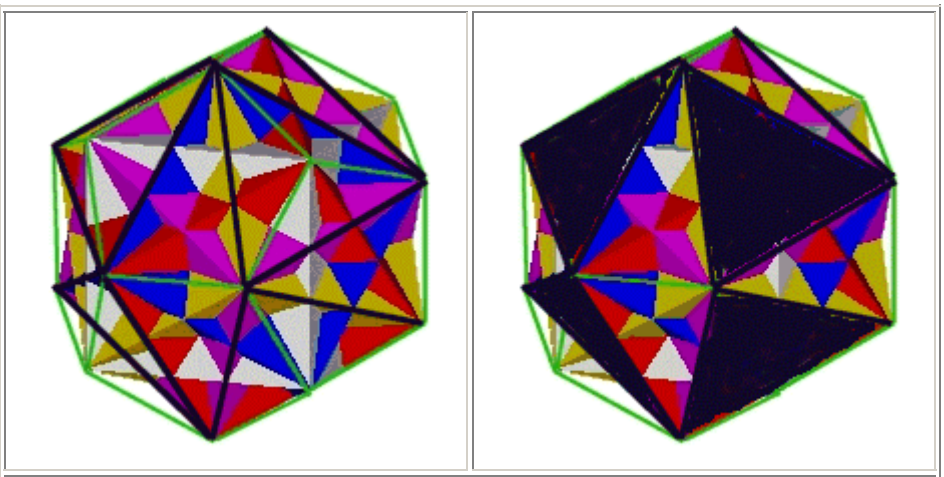


Fig. 6.18 *A Jitterbug in the Tetrahedron position*

It is difficult to see the Jitterbug defined by the Cube's edges in the above illustration, so here is a different perspective. The Dodecahedron is shown with the 5 Cubes. Some of the edges of the Cubes are outlined in black. Filling in the triangular faces in black reveals the Jitterbug.



Figs. 6.19, 6.20 *A Jitterbug defined by the Cube edges*

Notice the variation in the triangular face sizes and orientations. These variations show that there are many ***different*** Jitterbugs operating within the 120 Polyhedron.

Recall that we first introduced the rotation of 4 Cubes, plus a stationary Cube, as the defining motion for the 120 Polyhedron. But, as we have shown here, a single Jitterbug uses the same 4 axes of rotation, can be used to define Cubes (and therefore, regular Dodecahedra), Octahedra, Icosahedra, and adds an additional expansion/contraction motion which the 4 rotating Cubes and Octahedra did not have. By using 5 Jitterbugs and their expansion/contraction/rotation motion instead of the 5 Cubes and Octahedra, all the 120 Polyhedron vertices are defined.

rwgray@rwgrayprojects.com

7. Vortices

Lynnclaire reports that energy emerges from the 120 Polyhedron's Icosahedron type vertices and returns into the 120 Polyhedron through the regular Dodecahedron type vertices. This occurs when the 120 Polyhedron is deformed into an ellipsoid. (See the Pattern web site at http://www.pattern.org for more information on this deformation and other energy dynamics.)

When considering the energy flow through the 120 Polyhedra, we should consider the possibility of "vortex" like motion. That is, rotational motion in or out of the 120 Polyhedron. This is the kind of motion we showed with the Jitterbugs above.

Recall that the regular Dodecahedron defined by 5 intersecting Tetrahedra also defined what appears to be either a clockwise or counter clockwise vortex through the Dodecahedron's 12 pentagonal faces. The center of the Dodecahedron's faces is aligned radially with the Icosahedron type vertices of the 120 Polyhedron.

rwgray@rwgrayprojects.com

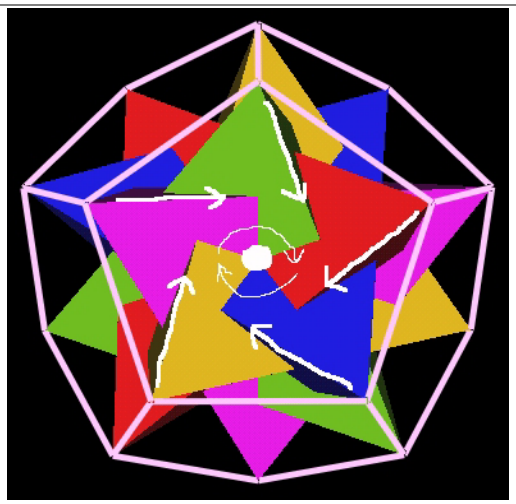


Fig. 7.1 *Vortex through a Dodecahedron face*

Each face appears to have a vortex with the same rotational direction.

Notice that since there are 3 pentagonal faces around every regular Dodecahedron's vertex, the vortex circular motions will be in conflict with each other.

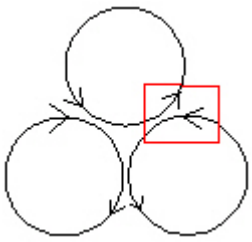


Fig. 7.2 *3 Vortex rotational conflict*

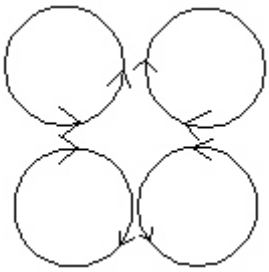


Fig. 7.3 *4 Vortex rotational synch.*

Because of this conflict, it is not clear that vortices are ever actually realized through the 12 face centers of the regular Dodecahedron.

If we consider the 5 intersecting Octahedra in the 120 Polyhedron as Jitterbugs, then we find that during the Jitterbug motion holes do appear at each Icosahedron type vertex.

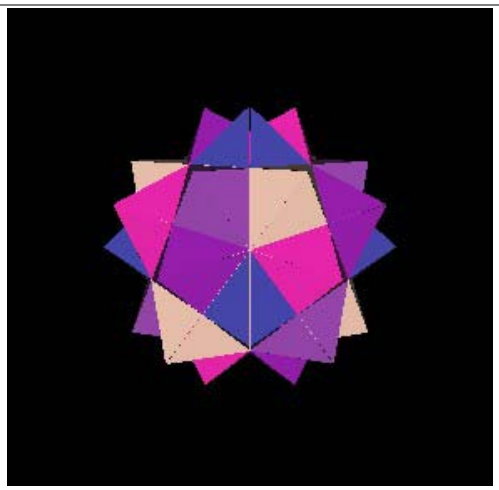


Fig. 7.4 *The 5 intersecting Octahedra*

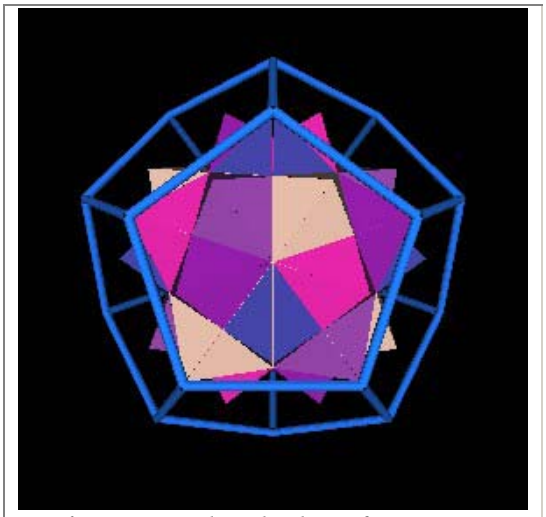


Fig. 7.5 *Dodecahedron face center*

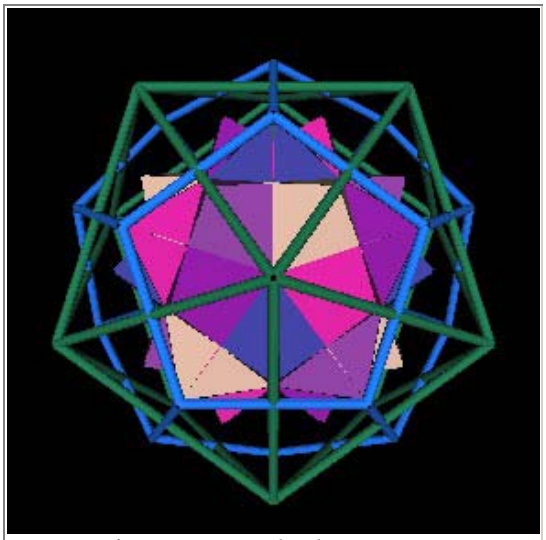


Fig. 7.6 *Icosahedron vertex*

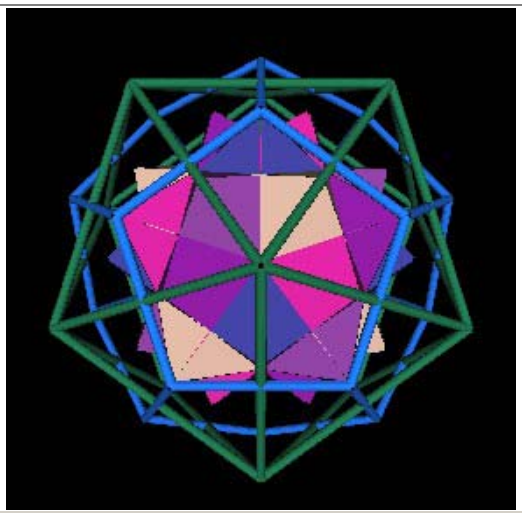


Fig. 7.7 *5 Jitterbug motions allows vortices at Icosahedron type vertices*

During this motion, the Jitterbugs pass through a regular Dodecahedron and Icosahedron position.

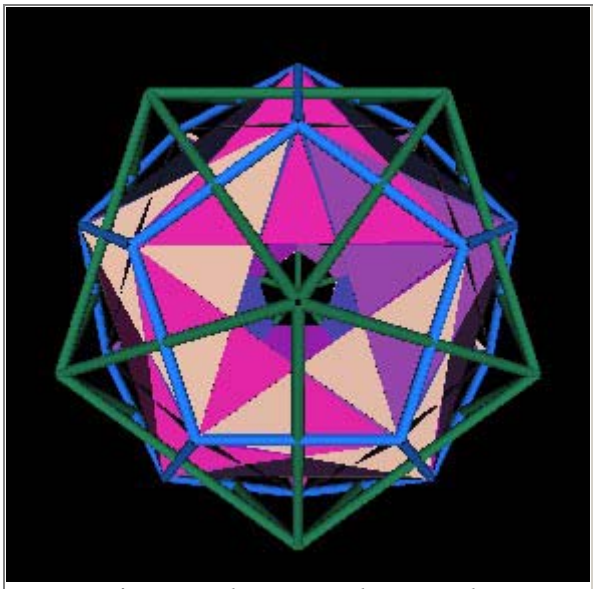


Fig. 7.8 *The 5 Jitterbugs in the Dodecahedron position*

rwgray@rwgrayprojects.com

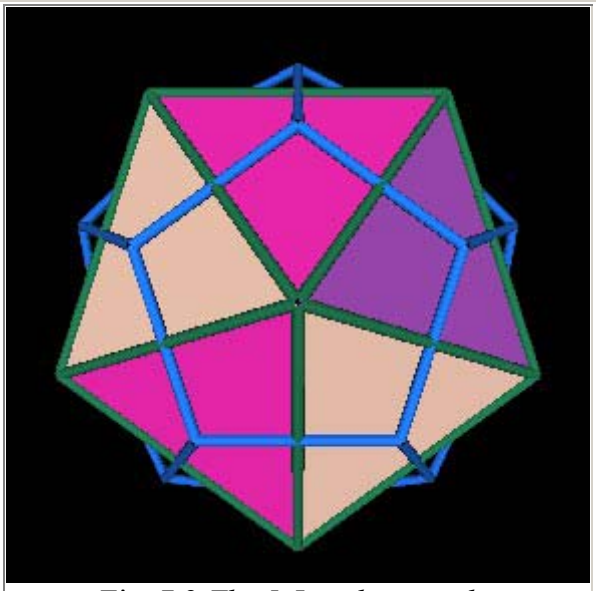


Fig. 7.9 *The 5 Jitterbugs in the Icosahedron position*

The "vortex" holes at the Icosahedron's 12 vertices are clearly visible when the 5 Jitterbugs are in the Dodecahedron position.

The other energy flow points that Lynnclaire describes, as mentioned above, occurs through the regular Dodecahedron's vertices. As we have described before, the vertices of the regular Dodecahedron coincide with the vertices of 5 Cubes. They are also radially inline with the Icosahedron face centers. The rotation axes of the 5 Jitterbugs also coincide with the 20 vertices of the Dodecahedron.

rwgray@rwgrayprojects.com

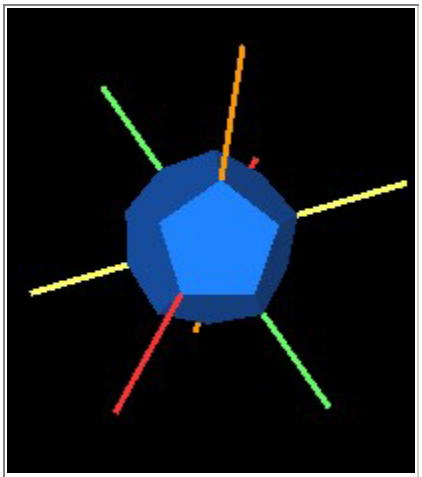
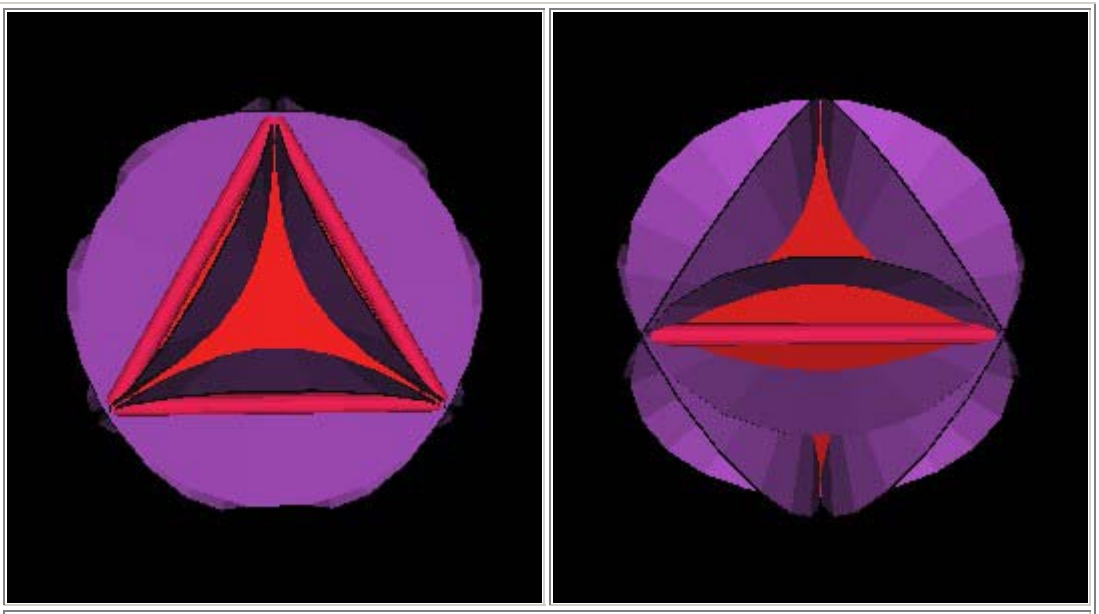


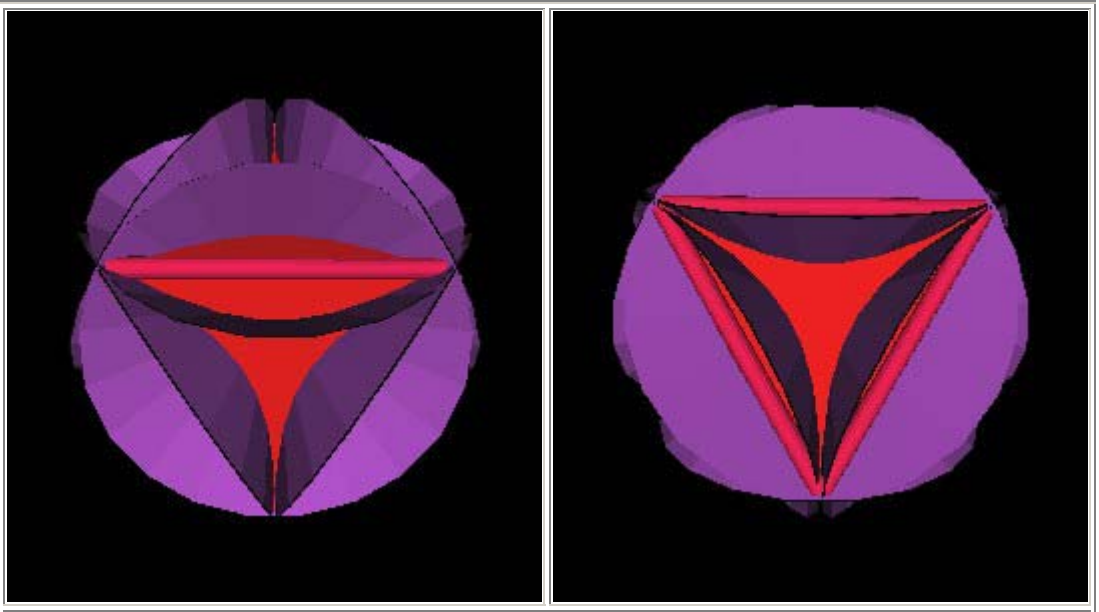
Fig. 7.10 *One set of Jitterbug axes through Dodecahedron*

Each of the 5 Jitterbugs of the 120 Polyhedron has 8 triangular faces. In the closed position, a Jitterbug is an Octahedron. The face center of any of the Jitterbugs' faces is radially in line with the regular Dodecahedron's vertices.

If we define the domain of a "vortex" at each of the Dodecahedron's vertices to be the cones with there apex at the center of volume and such that the cone completely encircles the triangular face of a Jitterbug in the Octahedron position, then we see in the next figure that even for a single Jitterbug, the vortex domains overlap each other.



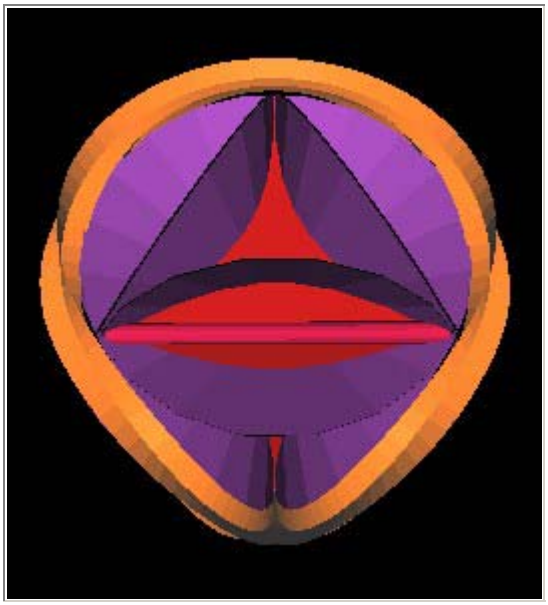
Figs. 7.11, 7.12 *Cones=Vortex Domains: Completely Encircle Triangles Overlap*



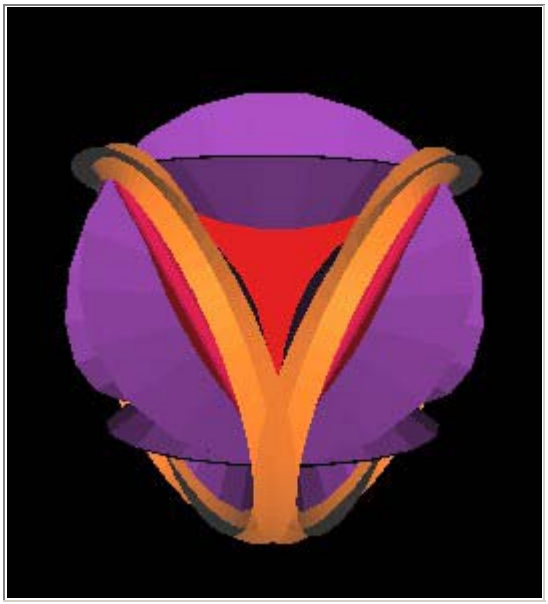
Figs. 7.13, 7.14 *Cones=Vortex Domains: Completely Encircle Triangles Overlap*

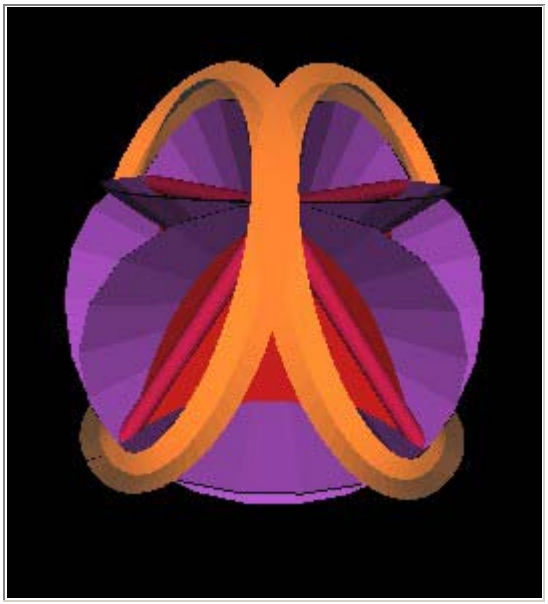
As we will see shortly, we can change the "open" angle of the 8 cones so that for a single Jitterbug the cones do not overlap each other. But I wanted to start with the cones open to such a degree as to completely enclose the Jitterbug's triangles because with these cones the circular rims of the cones can be used to trace out the knot pattern that Lynnclaire describes.

rwgray@rwgrayprojects.com



Figs. 7.15, 7.16 *The Pattern Knot Traced On Rim Of Cones*

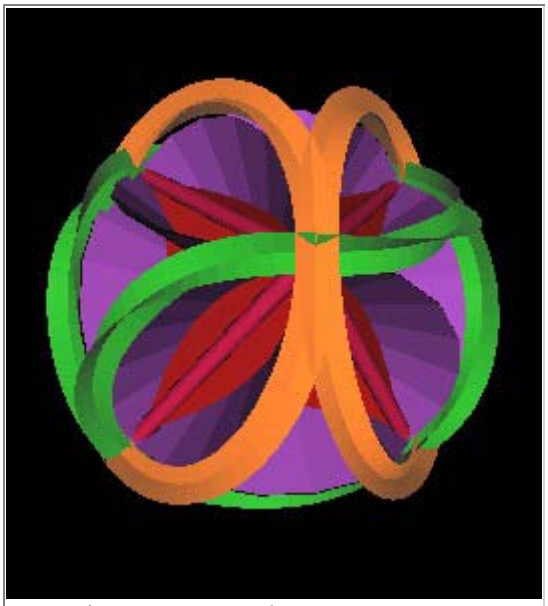




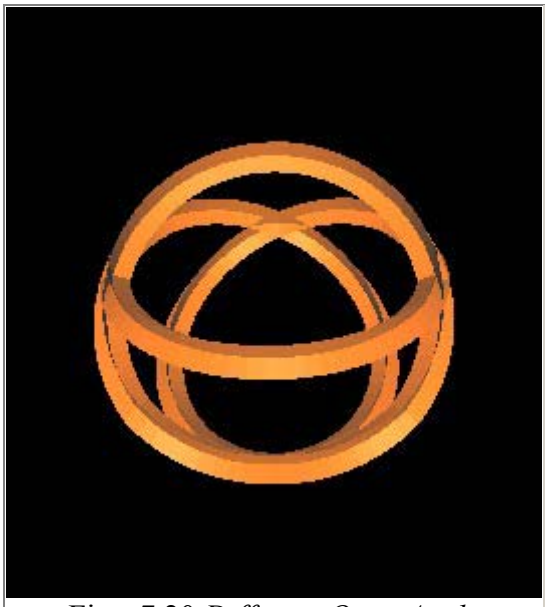
Figs. 7.17, 7.18 *The Pattern Knot Traced On Rim Of Cones*

Actually, several different "Pattern" knots can be traced on the cone's rims.

rwgray@rwgrayprojects.com



Figs. 7.19 *Another Pattern Knot On The Cone Rims*

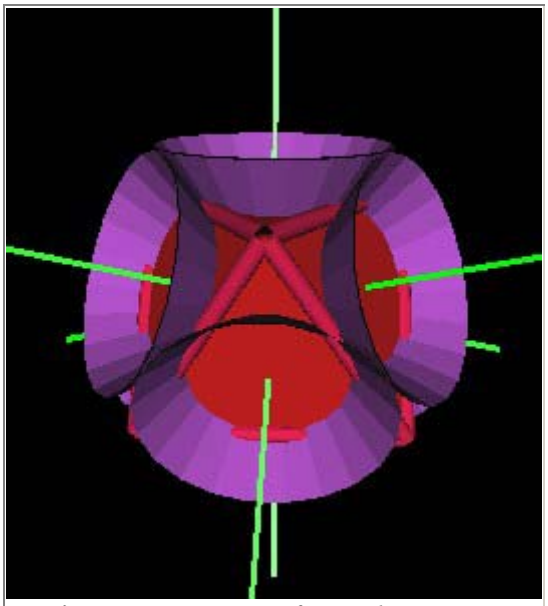


Figs. 7.20 *Different Open Angle Of The Pattern Knot*

Details about this knot can be found at Lynnclaire's web site www.pattern.org as well as my web pages at www.rwgrayprojects.com/Lynn/Lynn01.html

We can lessen the open angle of the 8 cones so that the cones do not overlap each other but are tangent to one another.

rwgray@rwgrayprojects.com



Figs. 7.21 *Cones Of Jitterbug Faces Just Touch*

Recall that the Jitterbug triangles rotate as they move radially. The 8 cones could rotate in the same directions as the Jitterbug triangles rotate without any conflict. The cones would not interfere with one another. This seems like a good choice for the domains of vortex-like motion.

However, there are many Jitterbugs in the 120 Polyhedron. When we look at the 5 Jitterbugs/Octahedra (4 rotated as described above) we see that the cones do intersect one another.

rwgray@rwgrayprojects.com

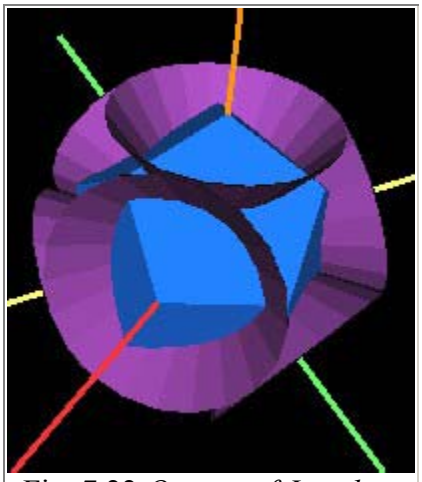


Fig. 7.22 *One set of Jitterbug cones.*

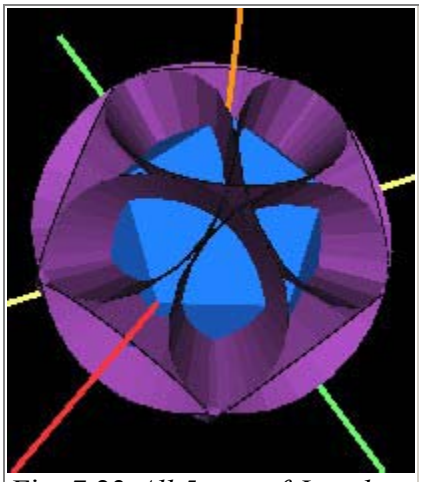


Fig. 7.23 *All 5 sets of Jitterbug cones*

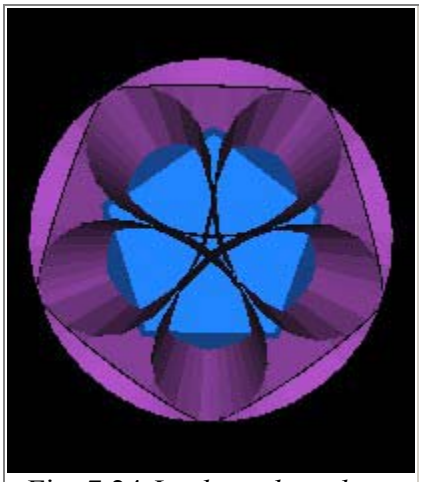


Fig. 7.24 *Looking directly at Dodecahedron's face center*

Each "petal" of the intersecting cones is centered on one of the 120 Polyhedron's "diamonds" (a rhombic Triacontahedron's face.)

Fig. 7.25 *The 120 Polyhedron's 30 "diamonds" in the cone's "petal" domains*

We can, of course, set the cone angle so that all 20 cones are tangent to one another.

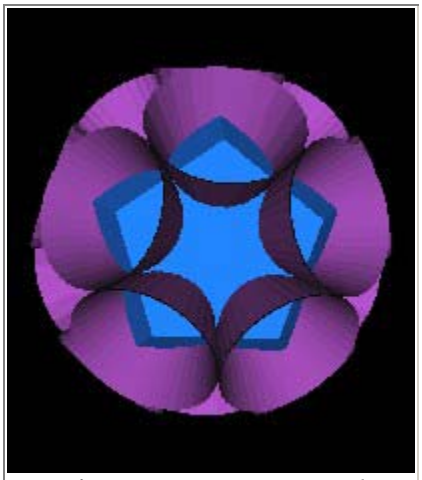


Fig. 7.26 *Cones around Dodecahedron's vertices*

However, because there are three cones touching every cone, there is no way to assign a rotation direction to each cone without some conflict in direction. (See Fig. 7.2 above.)

Each Jitterbug has 8 faces. With 5 Jitterbugs, this gives a total of 5x8=40 triangular faces. Therefore, each of the Icosahedron's 20 faces is covered by 2 Jitterbug faces. It can be seen in the movies above that these 2 Jitterbug triangles per Icosahedron face (Dodecahedron vertex) rotate in opposite directions. This, again, raises the question as to whether or not this Jitterbug motion, and vortices through the regular Dodecahedron's 20 vertices, are ever realized.

rwgray@rwgrayprojects.com

8. Vertex Coordinates

In order to calculate the properties of the 120 Polyhedron, it is helpful to first calculate the coordinates to its vertices. (I will be using Cartesian (x, y, z) coordinates.) But what orientation and scale of the 120 Polyhedron should be used? Is there a preferred orientation and scale which will make calculations easier or which will highlight some important features of the 120 Polyhedron?

In a note published on "synergetics-l@teleport.com", Gerald de Jong showed that the regular Dodecahedron could be assigned simple coordinates expressed in terms of the Golden ratio. This Golden ratio is often represented by the Greek letter phi. However, I will use the letter "p" in this text. The Golden ratio is

$$p = (1 + \sqrt{5}) / 2$$

which is approximately p = 1.618033989.

Gerald showed that the Dodecahedron's 20 vertices could all be assign numbers from the set {0, -p, p, -p^2, p^2, -p^3, p^3}. This is a remarkable set of numbers. For example, it can easily be shown that

$$p + p^2 = p^3$$

In general, it can be shown that (for n an integer)

$$p^n + p^{(n+1)} = p^{(n+2)}$$

rwgray@rwgrayprojects.com

Additionally, using these numbers for the coordinates of the regular Dodecahedron highlights the Golden ratio aspects of the polyhedron.

Since the regular Dodecahedron's vertices are the same as 20 of the 120 Polyhedron's vertices, I will use Gerald's 20 coordinates to fix the orientation and scale of the 120 Polyhedron. I will then calculate and fill in the remaining 62-20=42 coordinates.

rwgray@rwgrayprojects.com

Vertex	Type	X	Y	Z
1	A	0	0	2p^2
2	B	p^2	0	p^3
3	A	p	p^2	p^3
4	C	0	p	p^3
5	A	-p	p^2	p^3
6	B	-p^2	0	p^3
7	A	-p	-p^2	p^3
8	C	0	-p	p^3
9	A	p	-p^2	p^3
10	A	p^3	p	p^2
11	C	p^2	p^2	p^2
12	B	0	p^3	p^2
13	C	-p^2	p^2	p^2
14	A	-p^3	p	p^2
15	A	-p^3	-p	p^2
16	C	-p^2	-p^2	p^2
17	B	0	-p^3	p^2
18	C	p^2	-p^2	p^2
19	A	p^3	-p	p^2
20	C	p^3	0	p
21	A	p^2	p^3	p
22	A	-p^2	p^3	p
23	C	-p^3	0	p
24	A	-p^2	-p^3	p
25	A	p^2	-p^3	p

Vertex	Type	X	Y	Z
26	A	2p^2	0	0
27	B	p^3	p^2	0
28	C	p	p^3	0
29	A	0	2p^2	0
30	C	-p	p^3	0
31	B	-p^3	p^2	0
32	A	-2p^2	0	0
33	B	-p^3	-p^2	0
34	C	-p	-p^3	0
35	A	0	-2p^2	0
36	C	p	-p^3	0
37	B	p^3	-p^2	0

Vertex	Type	X	Y	Z
38	C	p^3	0	-p
39	A	p^2	p^3	-p
40	A	-p^2	p^3	-p
41	C	-p^3	0	-p
42	A	-p^2	-p^3	-p
43	A	p^2	-p^3	-p
44	A	p^3	p	-p^2
45	C	p^2	p^2	-p^2
46	B	0	p^3	-p^2
47	C	-p^2	p^2	-p^2
48	A	-p^3	p	-p^2
49	A	-p^3	-p	-p^2
50	C	-p^2	-p^2	-p^2
51	B	0	-p^3	-p^2
52	C	p^2	-p^2	-p^2
53	A	p^3	-p	-p^2
54	B	p^2	0	-p^3
55	A	p	p^2	-p^3
56	C	0	p	-p^3
57	A	-p	p^2	-p^3
58	B	-p^2	0	-p^3
59	A	-p	-p^2	-p^3
60	C	0	-p	-p^3
61	A	p	-p^2	-p^3
62	A	0	0	-2p^2

rwgray@rwgrayprojects.com

This is a remarkable set of coordinates. Remember that the 10 Tetrahedra, 5 Cubes, 5 Octahedra, 5 rhombic Dodecahedra, the regular Dodecahedron, Icosahedron and the rhombic Triacontahedron all share their vertices with the 120 Polyhedron. This means that ***all their vertex coordinates are a subset of the coordinates given above.***

Click here for tables defining all these polyhedra. (See Appendix.) These tables give all the vertex coordinates, edge maps and face maps for these polyhedra.

The combinations of 0, p, p^2, p^3 are also very interesting.

rwgray@rwgrayprojects.com

9. Basic Data For The 120 Polyhedron

Using the above coordinates, the basic data for the 120 Polyhedron can easily be calculated.

The 120 Polyhedron has 3 types of vertices. Each type of vertex is defined by the other polyhedra that share the vertex.

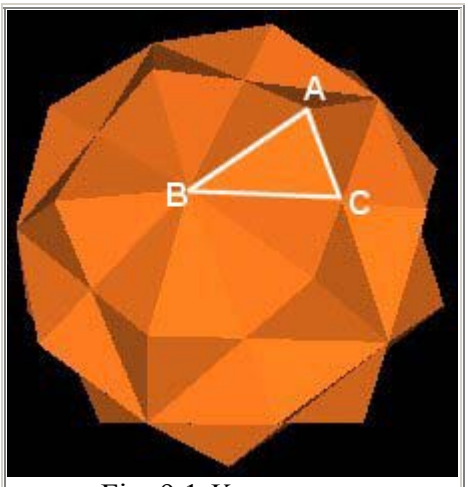


Fig. 9.1 *Vertex types*

Vertex Type	Shared With The Vertices Of
A	Octahedra, Rhombic Dodecahedra
B	Icosahedron, Rhombic Triacontahedron
C	Regular Dodecahedron, Rhombic Dodecahedra, Cubes, Tetrahedra, Rhombic Triacontahedron

rwgray@rwgrayprojects.com

The 3 different vertex types of the 120 Polyhedron are at different distances from the Polyhedron's center of volume.

Vertex Label	Radius	Approx.
A	2p^2	5.236067977
B	sqrt(2+p)p^2	4.97979657
C	sqrt(3)p^2	4.534567884

The edge lengths of the triangular face ABC are

Edge	Length	Approx.
AB	sqrt(3)p	2.802517077
AC	sqrt(2+p)	1.902113033
BC	sqrt(2+p)p	3.077683537

The face angles are calculated to be

Angle Label	Angle	Approx.
BAC	arccos(1/(sqrt(6+3p)p))	79.18768304°
ABC	arccos((p^2)/sqrt(6+3p))	37.37736814°
ACB	arccos(p/(2+p))	63.43494882°

The triangular face data is summarized in the following diagram.

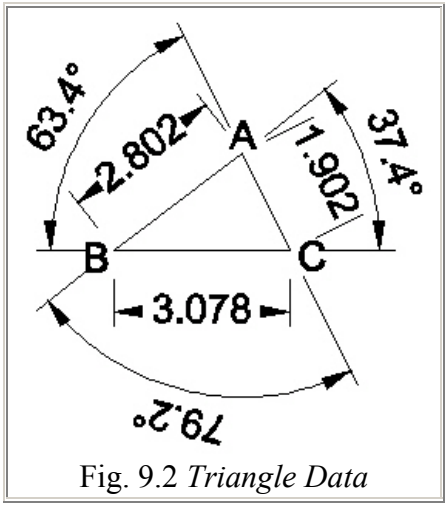


Fig. 9.2 *Triangle Data*

rwgray@rwgrayprojects.com

In calculating the volumes of the polyhedra in the 120 Polyhedron, I will use, as Fuller does, the Tetrahedron as unit volume.

Polyhedron	Coordinate Distance	Normalized Length	Volume	Approx.
Tetrahedron	Edge 2sqrt(2)p^2	1	1	1.0
Cube	Face Diagonal 2sqrt(2)p^2	1	3	3.0
Octahedron	Edge 2sqrt(2)p^2	1	4	4.0
Rhombic Dodecahedron	Long Face Diagonal 2sqrt(2)p^2	1	6	6.0
Regular Dodecahedron	Edge 2p	1/(sqrt(2)p)	(3/2)(2+p)	5.427050983
Icosahedron	Edge 2p^2	1/sqrt(2)	(5/2)p^2	6.545084972
Rhombic Triacontahedron	Long Face Diagonal 2p^2	1/sqrt(2)	15/2	7.5
120 Polyhedron	Long Face Diagonal of R. Triaconta. 2p^2	1/sqrt(2)	15/p	9.270509831

It is interesting to note that in the 120 Polyhedron, the Icosahedron edge length is equal to the Cube edge length. The Icosahedron edge length is also equal to the distance from the center of volume to an Octahedron vertex.

rwgray@rwgrayprojects.com

10. The Golden Ratio

The appearance of the Golden Ratio p=(1+sqrt(5))/2 in this polyhedron is not at all surprising. After all, the coordinates are given in terms of p and its powers. But what is surprising is that the Golden Ratio provides a way to organize so many polyhedra within a single polyhedron: The 120 Polyhedron that Lynnclaire experienced.

This leads to many questions.

Of particular interest, to me, is "If the Golden Ratio provides such an organizing system for these fundamental polyhedra, which is a reflection of the properties of the Space we live in, then why isn't it taught in Physics courses? Why is it that Physics seems not to have any need for the Golden Ratio when clearly Space (and time) seems to be organized around this ratio?"

The 120 Polyhedron clearly demonstrates that Space is organized around the Golden Mean. The Platonic polyhedra, which are the "fundamental" polyhedra, tell us what kind of space we live in. Physics does take note of this by cataloging the symmetries that the polyhedra exhibit. But Physics seems not to take note of nor utilize the Golden Ratio which can organize all these polyhedron into a single unit.

Time seems to be organized through the Golden Ratio as well. Or, perhaps a better statement of this would be that ***our perception*** of time is ordered by the Golden Ratio.

This was pointed out in Huntley's book "The Divine Proportion." Huntley points out that our perception of distances may be related to our brain's

rwgray@rwgrayprojects.com

perception of time. So, the statement that people find the Golden Rectangle (a rectangle whose edges are in the ratio of p:1) to be must pleasing of various rectangles is a statement about distances as well as time.

In music, the major 6th interval is considered, by most people, to be a very pleasing sound. The major 6th is in the ratio 8:5 which is 8/5=1.6, This is very close to the Golden Ratio p=1.618 (approx.). And music, sound waves, greatly depends on our perception of time.

The Golden Ratio occurs quite frequently in biology. Many growth patterns exhibit the Fibonacci numbers (1, 1, 2, 3, 5, 8, 13, 21, etc.) in which the next number is the sum of the previous 2 numbers. The Fibonacci sequence is known to be connected with the Golden Ratio (see comments below.)

What is different about Biology and Physics is that in Biology, systems grow. They increase and decrease in their constituent components. Physics seems to consider "stable" systems which, although dynamic, do not grow or multiply in their aggregate components.

This is not a very satisfying answer to the question "Where is the Golden Ratio in Physics", particularly since, at a fundamental level, Biology arises from Physics. They really can not be separated one from the other.

It would also be interesting to know where the Golden Ratio is in Chemistry. How does the Golden Ratio emerge from the Chemical processes? Where is the Golden Ratio in the molecules and atoms which Chemistry uses? Again, it doesn't seem to emerge until there is a time sequence of aggregate growth.

There are lots of questions to be asked and answered.

rwgray@rwgrayprojects.com

Before leaving this section, I'd like to point out an interesting equation from Huntley's book.

Although it is often pointed out that the Golden Ratio is the limit of successive Fibonacci numbers

$$\text{Lim}(n\text{->infinity})\ (f(n+1)/f(n)) = p$$

this is also true for ***any*** sequence f(n) defined by

$$f(n+1) = f(n) + f(n-1)$$

where f(n) is an integer for all n. The Fibonacci sequence (1, 1, 2, 3, 5, 8, 13, etc.) is but one such sequence. Any sequence as defined above will work.

The important point is that the Golden Ratio is ***not*** exclusively associated with the Fibonacci sequence, as many writings might lead you to beleive.

For example, try f(1)=-23, f(2)=15, then f(3)=-8, f(4)=7, f(5)=-1, f(6)=6, f(7)=5, f(8)=11, f(9)=16, f(10)=27, etc. Then the limit as n approaches infinity of f(n+1)/f(n) will equal the Golden Ratio. (f(10/f(9)=27/16=1.6875 which already starts to show the 1.6... of the Golden Ratio.)

So the Golden Ratio is connected with how the series is constructed and not a particular example of that construction (i.e. the Fibonacci sequence.) The Fibonacci sequence happens to be the most well known example of the construction rule

$$f(n+1) = f(n) + f(n-1)$$

rwgray@rwgrayprojects.com

11. Planes of the 120 Polyhedron

Returning to consider the coordinates of the 120 Polyhedron's vertices as listed above, it is obvious, from the z-coordinate, that the 62 vertices divide themselves into 9 groups. Each group defines a plane passing through the polyhedron. The spacing between the planes is shown in the next illustration.

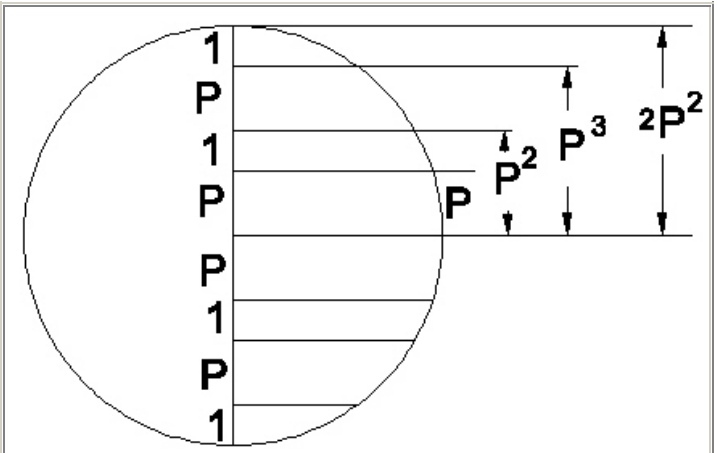


Fig. 11.1 *Vertex layers in the 120 Polyhedron*

The central angles of the intersection of these planes with a circumsphere are given in the next table The results illustrate interesting relations between the angles and the Golden Ratio p. The radius of the sphere is 2p^2 = 2 + 2p.

cos(18°)	sqrt(2+p)/2	sin(72°)
cos(30°)	sqrt(3)/2	sin(60°)
cos(36°)	p/2	sin(54°)
cos(45°)	1/sqrt(2)	sin(45°)
cos(54°)	sqrt(1+(1/p^2))/2	sin(36°)
cos(60°)	1/2	sin(30°)
cos(72°)	1/(2p)	sin(18°)

rwgray@rwgrayprojects.com

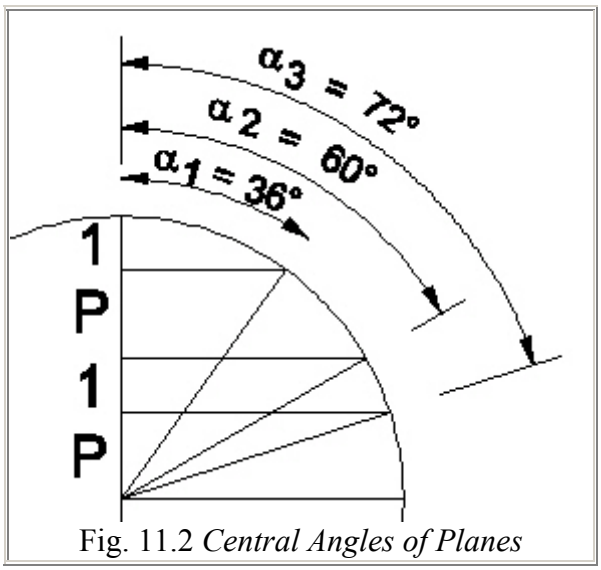


Fig. 11.2 *Central Angles of Planes*

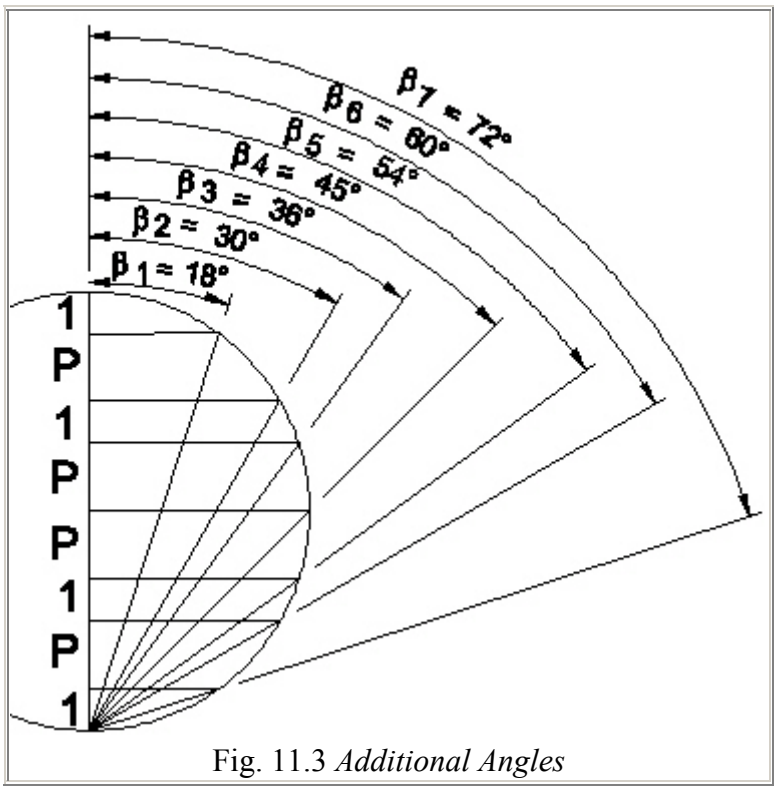


Fig. 11.3 *Additional Angles*

However, as pointed out above, not all of the 120 Polyhedron's vertices are at the same radial distance from the center of volume. This means that not all of the vertices lay on the circumsphere. I am only illustrating the planes defined by these vertices and not the vertices themselves.

rwgray@rwgrayprojects.com

12. Connection to the 144 Polyhedron

The other major polyhedron that Lynnclaire describes from her near death experiences is a 144 triangular faced polyhedron.

Fig. 12.1 *The 144 Polyhedron*

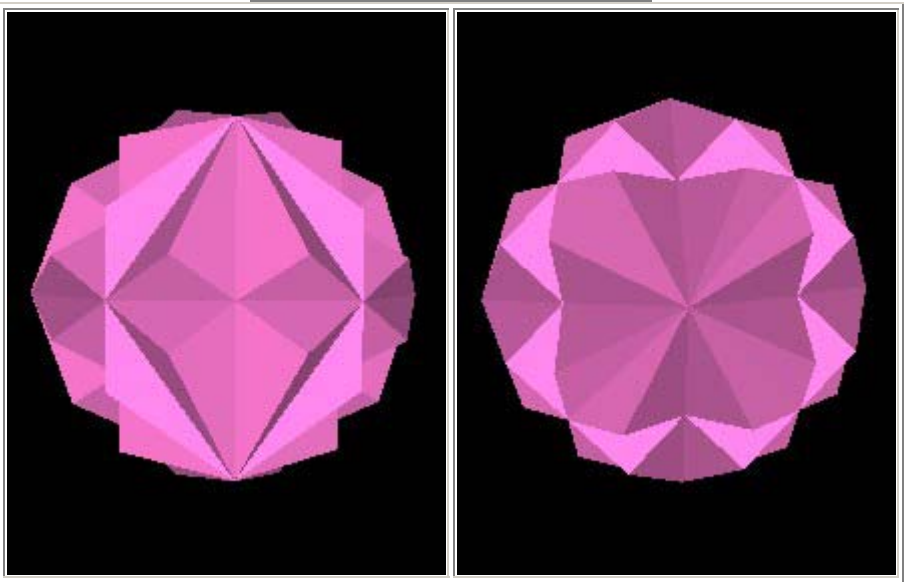


Fig. 12.2, 12.3 *The 144 Polyhedron*

It is interesting to note that the 144 Polyhedron is also based on the Cube-Octahedron dual pair.

rwgray@rwgrayprojects.com

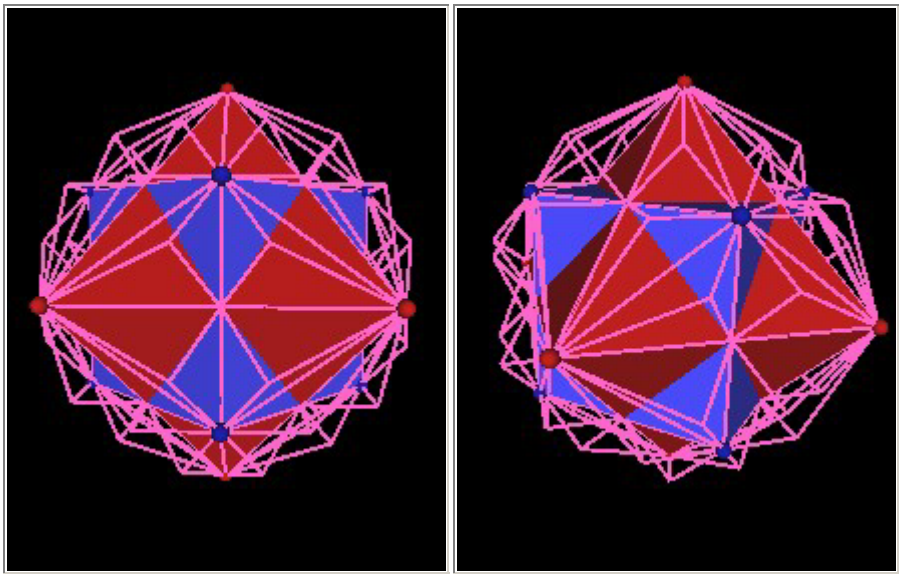


Fig. 12.4, 12.5 *The Cube and Octahedron in the 144 Polyhedron*

The next illustration shows the 144 Polyhedron (red) with one of the Octahedron's faces outlined in white and part of the cube face outlined in green.

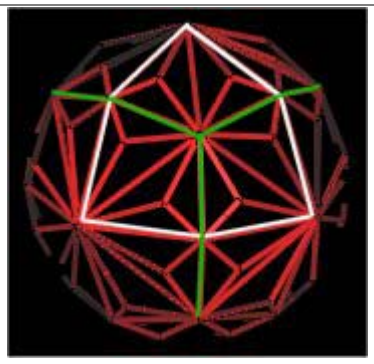


Fig. 12.6 *Octa and Cube outlined on the 144 Polyhedron*

In fact, this 144 Polyhedron can be constructed from the Face Centered Cubic (FCC) lattice, which Fuller called the Isotropic Vector Matrix (IVM). That is, all of the 144 Polyhedron's vertices coincide with FCC vertex points.

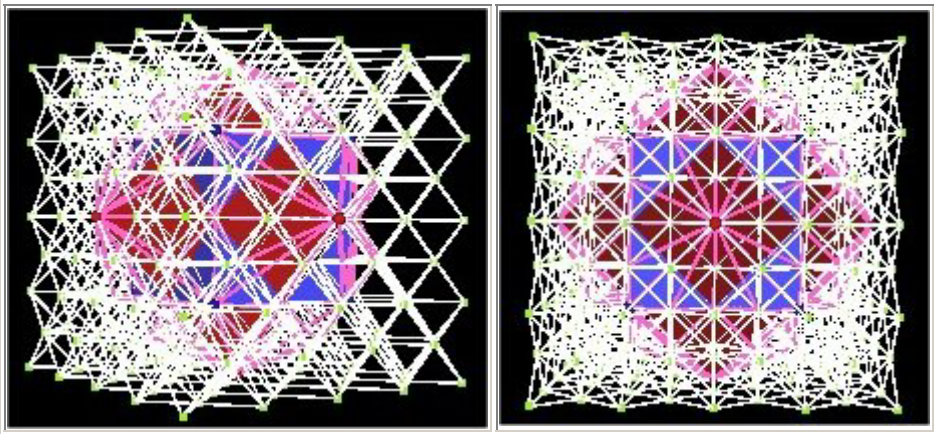


Fig. 12.7, 12.8 *The 144 Polyhedron in the IVM (FCC lattice)*

This also means that this polyhedron can be constructed from the closest packing of spheres.

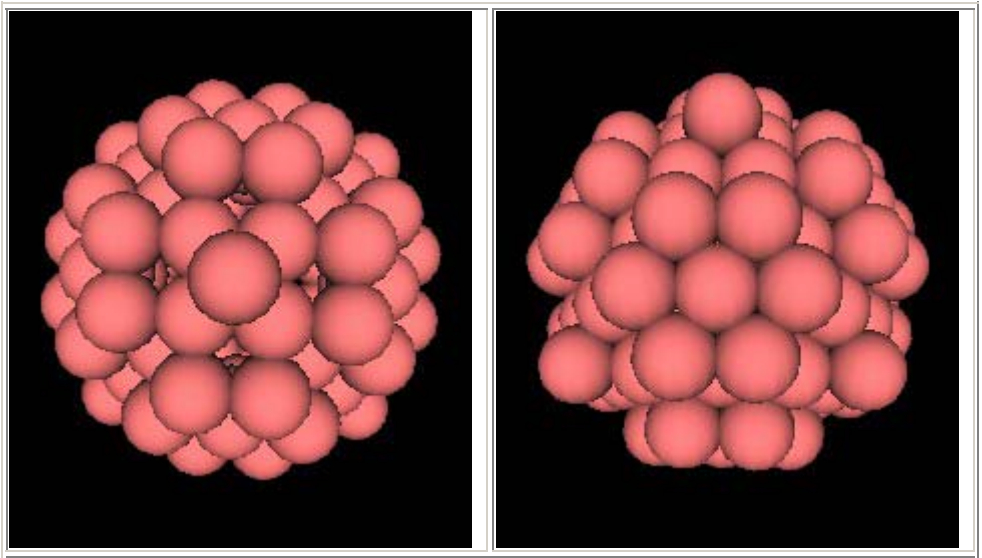


Fig. 12.9, 12.10 *The 144 Polyhedron made from closest packing of spheres*

Recall that when we first introduced the 120 Polyhedron, we rotated 4 cubes, leaving a 5th cube stationary. We then added the 5 Cube's duals: the Octahedron. The 144 Polyhedron has only one Cube-Octahedron dual pair.

What happens when we take 5 of these 144 Polyhedra and rotate 4 of them, as we did with the cubes, by an angular amount of 44.47751219°(approximately)? As before, we get 62-12=50 vertices of the 120 Polyhedron, the "missing" 12

vertices corresponding to the Icosahedron. However, there are also 120 additional external vertices from the 5 144 Polyhedra which do not coincide with any of the 120 Polyhedron's vertices. The next illustration shows these additional vertices.

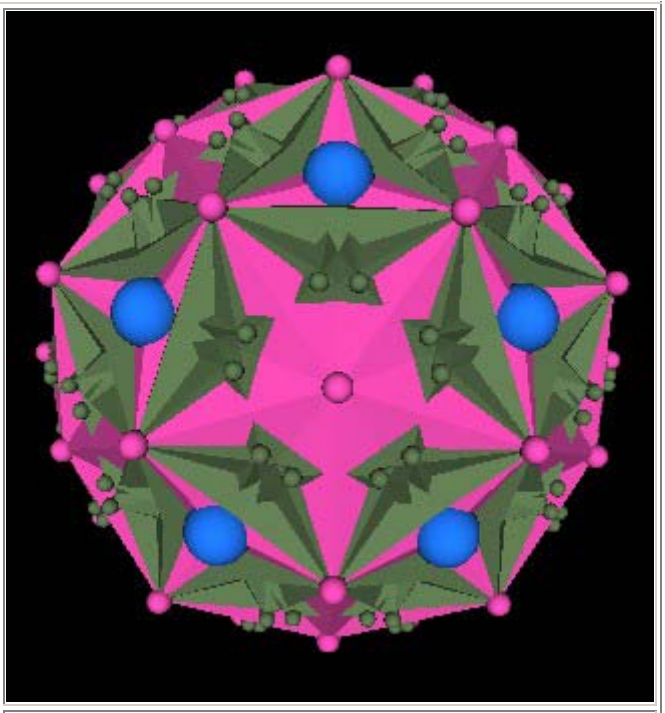


Fig. 12.11 *The 120 additional external vertices (green)*

In this illustration, the pink (polyhedron and spheres) mark the 120 Polyhedron. The green marks the 5 144 Polyhedron, and the blue marks the vertices of the regular Dodecahedron.

Since there are 10 green spheres within the regular Dodecahedron face region, and knowing that there are 12 Dodecahedron faces, there are then 12x10=120 vertices of the 144 Polyhedra showing through the 120 Polyhedron.

If we let the 4 144 Polyhedron continuously rotate, we get a movie in which it appears that surface waves are moving over the 120 Polyhedron.

rwgray@rwgrayprojects.com

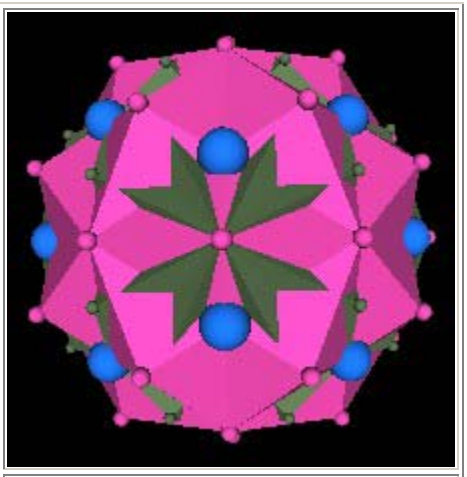


Fig. 12.12 *Surface waves over the 120 Polyhedron*

As with the 120 Polyhedron, the 144 clearly defines 9 planes on which all the vertices occur. The orientation for this counting of planes is the same as for the 120 Polyhedron. That is, the z-axis runs through opposite vertices of the Octahedron. (Of course other planes are also defined by the model.) Unlike the 120 Polyhedron case, each of the 9 planes are separated by an equal distance from its neighbor planes.

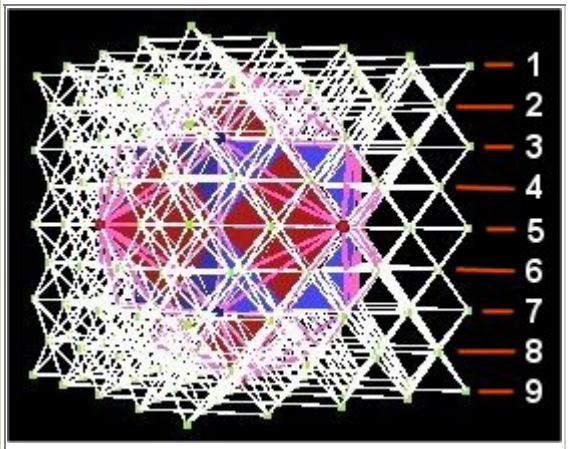


Fig. 12.13 *The 9 vertex planes of the 144 Polyhedron*

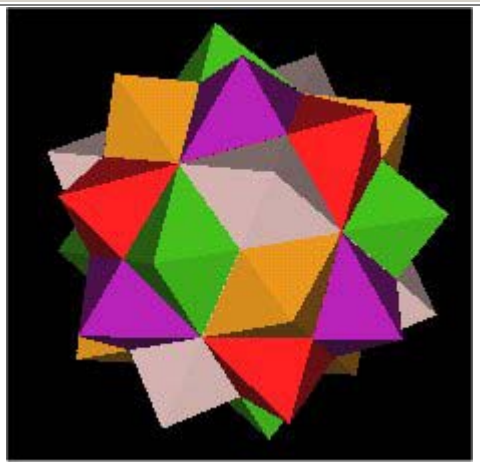
The distance between each plane, at the same scale as the 120 Polyhedron, so that the Octahedra in the 2 polyhedra are the same size, is given by

$$2p^2 / 4 = p^2 / 2 = (1 + p) / 2$$

rwgray@rwgrayprojects.com

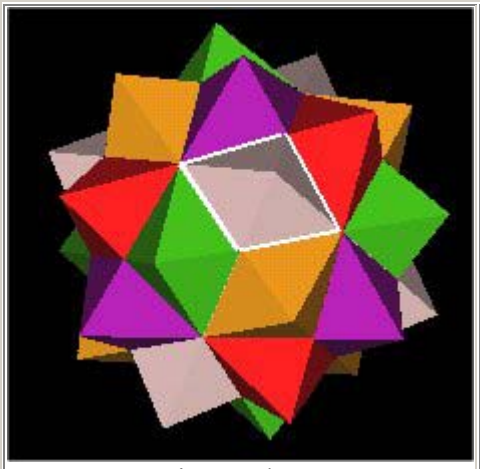
13. What Else can be Found

Earlier, I described how 5 Octahedra/Jitterbugs (4 of which are rotated with respect to the 5th) play a part in the construction of the 120 Polyhedron. It should be noted that the 5 rotated Octahedra ***alone*** defines another, different 120 triangular faced polyhedron. The image below shows the 5 rotated Octahedra with a slightly different orientation and color scheme. We can see that around each vertex of the Octahedra are 4 triangles. An Octahedron has 6 vertices. Adding up all the triangular faces we get a total of 5x6x4=120.

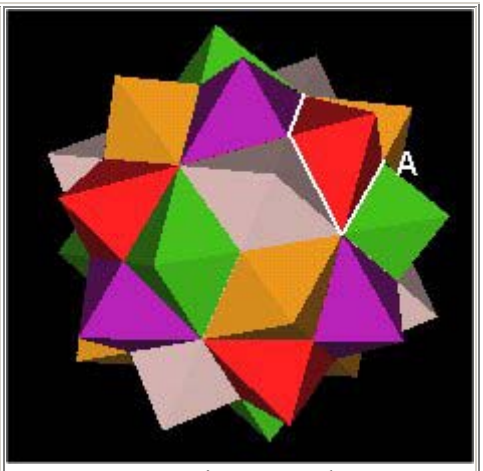


5 Intersecting Octa define a 120 polyhedron.

It looks like there are again 30 diamond "bases" (outlined in white) as there is with the original 120 Polyhedron. However, this is not the case because the diamond base is not planer. We can see this by looking at another diamond base. The point "A" is not in the same plane as the other 3 vertices of the "diamond".

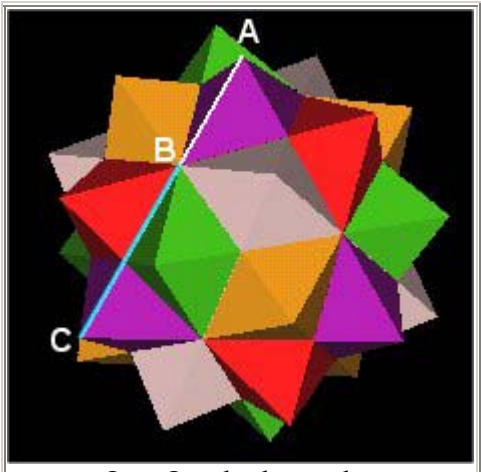


A Diamond Base?



No. Not in same plane.

Consider one edge of one of the Octahedra (purple Octahedron). It is intersected in 2 places by 4 Octahedra. We consider one intersection point (B) along the Octahedron's edge (AC).



One Octahedron edge
and intersection point.

It turns out that the ratio AC:BC is the Golden Ratio p.

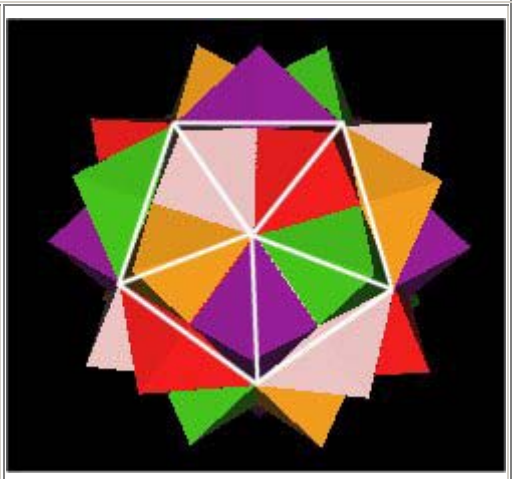
We can construct 2 equilateral triangles by connecting the intersection points. One triangle is rotated clockwise and the other is rotated counter clockwise with respect to the Octahedron's triangle.

Small triangle rotated CW.

Small triangle rotated CCW.

Each of these 2 cases define another Icosahedron.

Several edges if the Icosahedron.

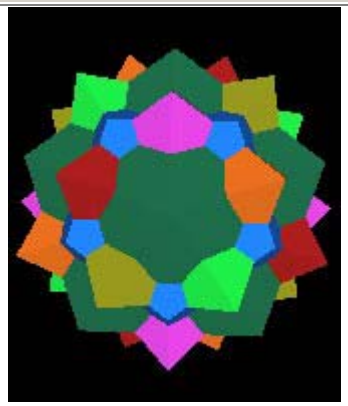


Several edges of the Icosahedron.

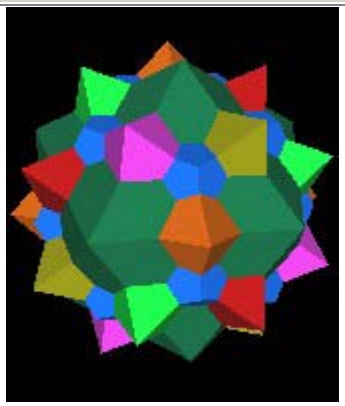
If we cut these 5 Jitterbugs in half and remove the front half, we can see into the system. We discover that during the Jitterbug motion a truncated Icosahedron position occurs. However, not all of the edges are of equal length. This polyhedron has 12 regular pentagons, 20 irregular hexagons and 60 vertices. Molecules in this polyhedral configuration are call "Buckminsterfullerenes".

Fig. 7.10 *Buckyball defined during the 5 Jitterbug's motion*

When we take an Octahedron, add in 5 intersecting Tetrahedra and the regular Dodecahedron, we get the following illustrations.



5 intersecting Cubes and Octa.

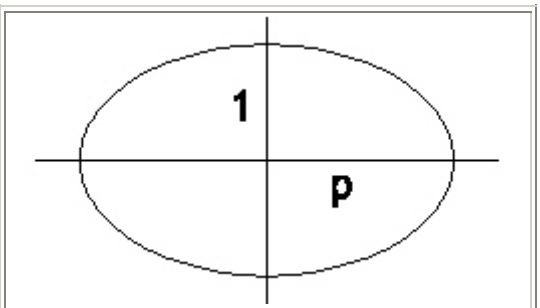


5 intersecting Cubes and Octa.

It appears that there are 12 new mini regular Dodecahedra defined (shown here in blue). (This needs to be confirmed through calculation.)

14. Other Dynamics

Lynnclaire reports that the 120 Polyhedron she experienced is not a static polyhedron. It "breaths" or expands into an ellipsoid shape. Could it be that the ellipsoid she experienced is the Golden Ellipsoid having its axes in the Golden Ratio?



Golden Ellipsoid

rwgray@rwgrayprojects.com

Appendix: Java Code For Polyhedra

Here is the Java code (tables) that I use to define the 10 Tetrahedra, 5 Octahedra, 5 Cubes, 5 Rhombic Dodecahedra, Icosahedron, regular Dodecadron, Triacontahedron, and the 120 Polyhedron.

First is a table of all the vertex coordinates to be used to define all these polyhedra.

Then for each polyhedron, a table of the vertices, edge maps, and face maps are given. Each such map is a list of the indices into the coordinate table to obtain the coordinate for the vertex being referenced.

Note that the Java object Point3d is just a 3-tuple. It is part of the Java 3D package in the javax.vecmath library.

```
double p = (1.0 + Math.sqrt(5.0)) / 2.0;
double p2 = p * p;
double p3 = p * p * p;

Point3d[] p120v = {new Point3d(    0.0,     0.0,     0.0),  // 00 center of volume

                   new Point3d(    0.0,     0.0,  2.0*p2),  // 01

                   new Point3d(     p2,     0.0,      p3),  // 02
                   new Point3d(      p,      p2,      p3),  // 03
                   new Point3d(    0.0,       p,      p3),  // 04
                   new Point3d(     -p,      p2,      p3),  // 05
                   new Point3d(    -p2,     0.0,      p3),  // 06
                   new Point3d(     -p,     -p2,      p3),  // 07
                   new Point3d(    0.0,      -p,      p3),  // 08
                   new Point3d(      p,     -p2,      p3),  // 09

                   new Point3d(     p3,       p,      p2),  // 10
                   new Point3d(     p2,      p2,      p2),  // 11
                   new Point3d(    0.0,      p3,      p2),  // 12
```


rwgray@rwgrayprojects.com

```
        new Point3d(    -p2,      p2,      p2),  // 13
        new Point3d(    -p3,       p,      p2),  // 14
        new Point3d(    -p3,      -p,      p2),  // 15
        new Point3d(    -p2,     -p2,      p2),  // 16
        new Point3d(    0.0,     -p3,      p2),  // 17
        new Point3d(     p2,     -p2,      p2),  // 18
        new Point3d(     p3,      -p,      p2),  // 19

        new Point3d(     p3,     0.0,       p),  // 20
        new Point3d(     p2,      p3,       p),  // 21
        new Point3d(    -p2,      p3,       p),  // 22
        new Point3d(    -p3,     0.0,       p),  // 23
        new Point3d(    -p2,     -p3,       p),  // 24
        new Point3d(     p2,     -p3,       p),  // 25

        new Point3d( 2.0*p2,     0.0,     0.0),  // 26
        new Point3d(     p3,      p2,     0.0),  // 27
        new Point3d(      p,      p3,     0.0),  // 28
        new Point3d(    0.0,  2.0*p2,     0.0),  // 29
        new Point3d(     -p,      p3,     0.0),  // 30
        new Point3d(    -p3,      p2,     0.0),  // 31
        new Point3d(-2.0*p2,     0.0,     0.0),  // 32
        new Point3d(    -p3,     -p2,     0.0),  // 33
        new Point3d(     -p,     -p3,     0.0),  // 34
        new Point3d(    0.0, -2.0*p2,     0.0),  // 35
        new Point3d(      p,     -p3,     0.0),  // 36
        new Point3d(     p3,     -p2,     0.0),  // 37

        new Point3d(     p3,     0.0,      -p),  // 38
        new Point3d(     p2,      p3,      -p),  // 39
        new Point3d(    -p2,      p3,      -p),  // 40
        new Point3d(    -p3,     0.0,      -p),  // 41
        new Point3d(    -p2,     -p3,      -p),  // 42
        new Point3d(     p2,     -p3,      -p),  // 43

        new Point3d(     p3,       p,     -p2),  // 44
        new Point3d(     p2,      p2,     -p2),  // 45
        new Point3d(    0.0,      p3,     -p2),  // 46
        new Point3d(    -p2,      p2,     -p2),  // 47
        new Point3d(    -p3,       p,     -p2),  // 48
        new Point3d(    -p3,      -p,     -p2),  // 49
        new Point3d(    -p2,     -p2,     -p2),  // 50
        new Point3d(    0.0,     -p3,     -p2),  // 51
        new Point3d(     p2,     -p2,     -p2),  // 52
        new Point3d(     p3,      -p,     -p2),  // 53

        new Point3d(     p2,     0.0,     -p3),  // 54
        new Point3d(      p,      p2,     -p3),  // 55
        new Point3d(    0.0,       p,     -p3),  // 56
        new Point3d(     -p,      p2,     -p3),  // 57
        new Point3d(    -p2,     0.0,     -p3),  // 58
        new Point3d(     -p,     -p2,     -p3),  // 59
        new Point3d(    0.0,      -p,     -p3),  // 60
        new Point3d(      p,     -p2,     -p3),  // 61

        new Point3d(    0.0,     0.0, -2.0*p2)   // 62
    };
```


rwgray@rwgrayprojects.com

```
// 10 different tetrahedra
int[][] TetrahedronVertices   = { { 4, 34, 38, 47},
                                  {18, 23, 28, 60},

                                  { 4, 36, 41, 45},
                                  {16, 20, 30, 60},

                                  { 8, 28, 41, 52},
                                  {13, 20, 34, 56},

                                  { 8, 30, 38, 50},
                                  {11, 23, 36, 56},

                                  {11, 16, 47, 52},
                                  {13, 18, 45, 50}

                                };

int[][] TetrahedronEdgeMap  = { { 4, 34,  4, 38,  4, 47, 34, 38, 34, 47, 38, 47},
                                {18, 23, 18, 28, 18, 60, 23, 28, 23, 60, 28, 60},
                                { 4, 36,  4, 41,  4, 45, 36, 41, 36, 45, 41, 45},
                                {16, 20, 16, 30, 16, 60, 20, 30, 20, 60, 30, 60},
                                { 8, 28,  8, 41,  8, 52, 28, 41, 28, 52, 41, 52},
                                {13, 20, 13, 34, 13, 56, 20, 34, 20, 56, 34, 56},

                                { 8, 30,  8, 38,  8, 50, 30, 38, 30, 50, 38, 50},
                                {11, 23, 11, 36, 11, 56, 23, 36, 23, 56, 36, 56},

                                {11, 16, 11, 47, 11, 52, 16, 47, 16, 52, 47, 52},
                                {13, 18, 13, 45, 13, 50, 18, 45, 18, 50, 45, 50}
                              };

int[][] TetrahedronFaceMap  = { { 4, 34, 47,
                                  4, 38, 34,
                                  4, 47, 38,
                                 34, 38, 47},
                                {18, 23, 28,
                                 18, 23, 60,
                                 18, 28, 60,
                                 23, 28, 60},

                                { 4, 41, 36,
                                  4, 36, 45,
                                  4, 45, 41,
                                 36, 41, 45},
                                {16, 20, 30,
                                 16, 20, 60,
                                 16, 30, 60,
                                 20, 30, 60},

                                { 8, 28, 41,
                                  8, 28, 52,
                                  8, 41, 52,
                                 28, 41, 52},
```


rwgray@rwgrayprojects.com

```
                                {13, 20, 34,
                                 13, 20, 56,
                                 13, 34, 56,
                                 20, 34, 56},

                                { 8, 30, 38,
                                  8, 30, 50,
                                  8, 38, 50,
                                 30, 38, 50},
                                {11, 23, 36,
                                 11, 23, 56,
                                 11, 36, 56,
                                 23, 36, 56},

                                {47, 16, 11,
                                 52, 47, 11,
                                 16, 52, 11,
                                 47, 52, 16},
                                {13, 18, 45,
                                 13, 18, 50,
                                 13, 45, 50,
                                 18, 45, 50}
                               };


// 5 different cubes
int[][] CubeVertices = { { 4, 18, 23, 28, 34, 38, 47, 60},
                         { 4, 16, 20, 30, 36, 41, 45, 60},
                         { 8, 13, 20, 28, 34, 41, 52, 56},
                         { 8, 11, 23, 30, 36, 38, 50, 56},
                         {11, 13, 16, 18, 45, 47, 50, 52}
                       };

int[][] CubeEdgeMap  = { { 4, 18, 18, 38, 38, 28, 28,  4,
                           4, 23, 18, 34, 28, 47, 38, 60,
                          23, 34, 34, 60, 60, 47, 47, 23},
                         { 4, 16, 16, 36, 36, 20, 20,  4,
                           4, 30, 16, 41, 20, 45, 36, 60,
                          30, 41, 41, 60, 60, 45, 45, 30},
                         { 8, 13, 13, 28, 28, 20, 20,  8,
                           8, 34, 13, 41, 28, 56, 20, 52,
                          34, 41, 41, 56, 56, 52, 52, 34},
                         { 8, 11, 11, 30, 30, 23, 23,  8,
                           8, 36, 11, 38, 23, 50, 30, 56,
                          36, 38, 38, 56, 56, 50, 50, 36},
                         {11, 13, 13, 16, 16, 18, 18, 11,
                          11, 45, 13, 47, 16, 50, 18, 52,
                          45, 47, 47, 50, 50, 52, 52, 45}
                       };

int[][] CubeFaceMap  = { { 4, 18, 38,
                          38, 28,  4,

                           4, 23, 18,
                          18, 23, 34,

                           4, 28, 47,
```


rwgray@rwgrayprojects.com

```
  4, 47, 23,

 28, 38, 60,
 28, 60, 47,

 23, 47, 34,
 47, 60, 34,

 38, 18, 60,
 18, 34, 60
},

{ 4, 16, 20,
 16, 36, 20,

  4, 20, 45,
  4, 45, 30,

  4, 30, 41,
  4, 41, 16,

  45, 41, 30,
  45, 60, 41,

  20, 36, 60,
  20, 60, 45,

  36, 16, 41,
  36, 41, 60},

{ 8, 13, 20,
 13, 28, 20,

  8, 20, 52,
  8, 52, 34,

  8, 34, 41,
  8, 41, 13,

 20, 28, 56,
 20, 56, 52,

 52, 56, 41,
 52, 41, 34,

 28, 13, 41,
 28, 41, 56},

{ 8, 11, 23,
 11, 30, 23,

  8, 23, 50,
  8, 50, 36,

  8, 36, 38,
  8, 38, 11,
```


rwgray@rwgrayprojects.com

```
                            23, 30, 56,
                            23, 56, 50,

                            50, 56, 38,
                            50, 38, 36,

                            30, 11, 56,
                            11, 38, 56},

                          {11, 13, 16,
                           11, 16, 18,

                           11, 45, 47,
                           11, 47, 13,

                           13, 47, 50,
                           13, 50, 16,

                           16, 50, 52,
                           16, 52, 18,

                           18, 52, 45,
                           18, 45, 11,

                           45, 50, 52,
                           45, 47, 50}
                        };

// 5 differnet octahedra
int[][] OctahedronVertices = {{ 7, 10, 22, 43, 49, 55},
                              { 9, 14, 21, 42, 53, 57},
                              { 3, 15, 25, 40, 44, 59},
                              { 5, 19, 24, 39, 48, 61},
                              { 1, 26, 29, 32, 35, 62}
                             };

int[][] OctahedronEdgeMap  = { { 7, 10,  7, 22,  7, 43,  7, 49,
                                 10, 22, 10, 43, 22, 49, 43, 49,
                                 10, 55, 22, 55, 43, 55, 49, 55},
                               { 9, 14,  9, 21,  9, 42,  9, 53,
                                 14, 21, 14, 42, 21, 53, 42, 53,
                                 14, 57, 21, 57, 42, 57, 53, 57},

                               { 3, 15,  3, 25,  3, 40,  3, 44,
                                 15, 25, 15, 40, 40, 44, 25, 44,
                                 25, 59, 15, 59, 40, 59, 44, 59},


                               { 5, 19,  5, 24,  5, 39,  5, 48,
                                 19, 24, 19, 39, 24, 48, 39, 48,
                                 19, 61, 24, 61, 39, 61, 48, 61},

                               { 1, 26,  1, 29,  1, 32,  1, 35,
                                 26, 29, 29, 32, 32, 35, 35, 26,
                                 62, 26, 62, 29, 62, 32, 62, 35}

                               };
```


rwgray@rwgrayprojects.com

```
int[][] OctahedronFaceMap  = {  { 7, 10, 43,
                                  7, 22, 10,
                                  7, 43, 49,
                                  7, 49, 22,

                                 55, 10, 43,
                                 55, 22, 10,
                                 55, 43, 49,
                                 55, 49, 22},

                                { 9, 14, 21,
                                  9, 21, 53,
                                  9, 42, 14,
                                  9, 53, 42,

                                 57, 14, 21,
                                 57, 21, 53,
                                 57, 42, 14,
                                 57, 53, 42},

                               { 3, 15, 25,
                                 3, 25, 44,
                                 3, 40, 15,
                                 3, 44, 40,

                                 59, 25, 15,
                                 59, 15, 40,
                                 59, 40, 44,
                                 59, 44, 25},

                                { 5, 19, 39,
                                  5, 24, 19,
                                  5, 39, 48,
                                  5, 48, 24,

                                 61, 19, 39,
                                 61, 24, 19,
                                 61, 39, 48,
                                 61, 48, 24},

                               { 1, 26, 29,
                                 1, 29, 32,
                                 1, 32, 35,
                                 1, 35, 26,

                                62, 29, 26,
                                62, 32, 29,
                                62, 35, 32,
                                62, 26, 35}


                              };
```


rwgray@rwgrayprojects.com

```
// 1 regular dodecahedron
int[] RegDodecahedronVertices = { 4,  8, 11, 13, 16, 18, 20, 23, 28,
                                  30, 34, 36, 38, 41, 45, 47, 50, 52,
                                  56, 60};
int[] RegDodecahedronEdgeMap  = { 4,  8,  4, 11, 4, 13,
                                   8, 16,  8, 18,
                                  11, 20, 11, 28,
                                  13, 30, 13, 23,
                                  16, 23, 16, 34,
                                  18, 36, 18, 20,
                                  20, 38,
                                  23, 41,
                                  28, 30, 28, 45,
                                  30, 47,
                                  34, 50, 34, 36,
                                  36, 52,
                                  38, 45, 38, 52,
                                  41, 47, 41, 50,
                                  45, 56,
                                  47, 56,
                                  50, 60,
                                  52, 60,
                                  56, 60
                                 };
int[] RegDodecahedronFaceMap  = { 4,  8, 11, 11,  8, 18, 11, 18, 20,
                                   4, 13, 23,  4, 23,  8,  8, 23, 16,
                                   4, 11, 28,  4, 28, 30,  4, 30, 13,
                                   8, 16, 34,  8, 34, 18, 18, 34, 36,
                                  11, 20, 28, 20, 45, 28, 20, 38, 45,
                                  13, 30, 23, 23, 30, 41, 41, 30, 47,
                                  16, 23, 34, 34, 23, 50, 50, 23, 41,
                                  18, 36, 52, 18, 52, 38, 18, 38, 20,
                                  28, 45, 56, 28, 56, 47, 28, 47, 30,
                                  34, 50, 60, 34, 60, 36, 36, 60, 52,
                                  38, 52, 60, 38, 60, 56, 38, 56, 45,
                                  41, 47, 56, 41, 56, 60, 41, 60, 50
                                 };

// 5 rhombic dodecahedra
int[][] RhDodecahedronVertices = { { 4,  7, 10, 18, 22, 23, 28, 34, 38, 43, 47, 49, 55, 60},
                                   { 4,  9, 14, 16, 20, 21, 30, 36, 41, 42, 45, 53, 57, 60},
                                   { 3,  8, 13, 15, 20, 25, 28, 34, 40, 41, 44, 52, 56, 59},
                                   { 5,  8, 11, 19, 23, 24, 30, 36, 38, 39, 48, 50, 56, 61},
                                   { 1, 11, 13, 16, 18, 26, 29, 32, 35, 45, 47, 50, 52, 62}
                                 };
int[][] RhDodecahedronEdgeMap  = {
                                   { 7,  4,  7, 18,  7, 23,  7, 34,
                                    10,  4, 10, 18, 10, 28, 10, 38,
                                    22,  4, 22, 23, 22, 28, 22, 47,
                                    43, 18, 43, 34, 43, 38, 43, 60,
                                    49, 23, 49, 34, 49, 47, 49, 60,
                                    55, 28, 55, 38, 55, 47, 55, 60},
                                   { 9,  4,  9, 16,  9, 20,  9, 36,
                                    14,  4, 14, 16, 14, 30, 14, 41,
                                    21,  4, 21, 20, 21, 30, 21, 45,
                                    42, 16, 42, 36, 42, 41, 42, 60,
                                    53, 20, 53, 36, 53, 45, 53, 60,
```


rwgray@rwgrayprojects.com

```
                                    57, 30, 57, 41, 57, 45, 57, 60},
                                   { 3,  8,  3, 13,  3, 20,  3, 28,
                                    15,  8, 15, 13, 15, 34, 15, 41,
                                    25,  8, 25, 20, 25, 34, 25, 52,
                                    40, 13, 40, 28, 40, 41, 40, 56,
                                    44, 20, 44, 28, 44, 52, 44, 56,
                                    59, 34, 59, 41, 59, 52, 59, 56},
                                   { 5,  8,  5, 11,  5, 23,  5, 30,
                                    19,  8, 19, 11, 19, 36, 19, 38,
                                    24,  8, 24, 23, 24, 36, 24, 50,
                                    39, 11, 39, 30, 39, 38, 39, 56,
                                    48, 23, 48, 30, 48, 50, 48, 56,
                                    61, 36, 61, 38, 61, 50, 61, 56},
                                   {  1, 11,  1, 13,  1, 16,  1, 18,
                                     26, 18, 26, 11, 26, 45, 26, 52,
                                     29, 11, 29, 13, 29, 47, 29, 45,
                                     32, 13, 32, 16, 32, 50, 32, 47,
                                     35, 16, 35, 18, 35, 52, 35, 50,
                                     62, 45, 62, 47, 62, 50, 62, 52}
                                 };
int[][] RhDodecahedronFaceMap  = {
                                   { 7,  4, 18, 10,  4, 18,
                                     7, 18, 34, 43, 18, 34,
                                     7, 34, 23, 49, 34, 23,
                                     7,  4, 23, 22,  4, 23,

                                    22,  4, 28, 10,  4, 28,
                                    18, 10, 43, 38, 10, 43,
                                    34, 49, 43, 60, 49, 43,
                                    23, 49, 22, 47, 49, 22,

                                    55, 38, 60, 43, 38, 60,
                                    55, 60, 47, 49, 60, 47,
                                    55, 47, 28, 22, 47, 28,
                                    55, 28, 38, 10, 28, 38
                                   },

                                   { 9,  4, 16, 14,  4, 16,
                                     9, 16, 36, 42, 16, 36,
                                     9, 36, 20, 53, 36, 20,
                                     9, 20,  4, 21, 20,  4,

                                    14,  4, 30, 21,  4, 30,
                                    16, 42, 14, 41, 42, 14,
                                    36, 42, 53, 60, 42, 53,
                                    20, 21, 53, 45, 21, 53,

                                    42, 41, 60, 57, 41, 60,
                                    53, 45, 60, 57, 45, 60,
                                    21, 30, 45, 57, 30, 45,
                                    14, 41, 30, 57, 41, 30
                                   },

                                   { 3,  8, 13, 15,  8, 13,
                                     3, 13, 28, 40, 13, 28,
                                     3, 28, 20, 44, 28, 20,
                                     3, 20,  8, 25, 20,  8,
```


rwgray@rwgrayprojects.com

```
                                    8, 25, 15, 34, 25, 15,
                                   13, 15, 40, 41, 15, 40,
                                   28, 44, 40, 56, 44, 40,
                                   20, 44, 25, 52, 44, 25,

                                   15, 41, 34, 59, 41, 34,
                                   40, 56, 41, 59, 56, 41,
                                   44, 56, 52, 59, 56, 52,
                                   25, 34, 52, 59, 34, 52
                                  },

                                  {
                                    5,  8, 11, 19,  8, 11,
                                    5, 11, 30, 39, 11, 30,
                                    5, 30, 23, 48, 30, 23,
                                    5, 23,  8, 24, 23,  8,

                                    8, 19, 24, 36, 19, 24,
                                   11, 19, 39, 38, 19, 39,
                                   30, 39, 48, 56, 39, 48,
                                   23, 48, 24, 50, 48, 24,

                                   19, 38, 36, 61, 38, 36,
                                   39, 38, 56, 61, 38, 56,
                                   48, 50, 56, 61, 50, 56,
                                   24, 36, 50, 61, 36, 50
                                  },

                                  {  1, 11, 29, 29, 13,  1,
                                     1, 13, 32, 32, 16,  1,
                                     1, 16, 35, 35, 18,  1,
                                     1, 18, 26, 26, 11,  1,

                                    26, 45, 29, 29, 11, 26,
                                    29, 47, 32, 32, 13, 29,
                                    32, 50, 35, 35, 16, 32,
                                    35, 52, 26, 26, 18, 35,

                                    62, 45, 26, 26, 52, 62,
                                    62, 47, 29, 29, 45, 62,
                                    62, 50, 32, 32, 47, 62,
                                    62, 52, 35, 35, 50, 62}
                                 };

// 1 icosahedron
int[] IcosaVertices = { 2,  6,
                        12, 17,
                        27, 31, 33, 37,
                        46, 51,
                        54, 58
                       };

int[] IcosaEdgeMap  = { 2,  6,  2, 12,  2, 17,  2, 37,  2, 27,
                          6, 12,  6, 17,  6, 31,  6, 33,
                        12, 27, 12, 46, 12, 31,
                        17, 33, 17, 51, 17, 37,
```


rwgray@rwgrayprojects.com

```
                        27, 37, 27, 54, 27, 46,
                        31, 46, 31, 58, 31, 33,
                        33, 58, 33, 51,
                        37, 51, 37, 54,
                        46, 54, 46, 58,
                        51, 54, 51, 58,
                        54, 58
                       };

int[] IcosaFaceMap  = { 2,  6, 17,
                         2, 12,  6,
                         2, 17, 37,
                         2, 37, 27,
                         2, 27, 12,

                        37, 54, 27,
                        27, 54, 46,
                        27, 46, 12,
                        12, 46, 31,
                        12, 31, 6,
                         6, 31, 33,
                         6, 33, 17,
                        17, 33, 51,
                        17, 51, 37,
                        37, 51, 54,

                        58, 54, 51,
                        58, 46, 54,
                        58, 31, 46,
                        58, 33, 31,
                        58, 51, 33
                       };



// 1 rhombic triacontahedron
int[] RhTriaVertices = { 2,  4,  6,  8, 11,
                         12, 13, 16, 17, 18, 20, 23,
                         27, 28, 30, 31, 33, 34, 36, 37,
                         38, 41, 45, 46, 47, 50, 51, 52,
                         54, 56, 58, 60
                        };

int[] RhTriaEdgeMap  = { 2,  4,  4,  6,  6,  8,  8,  2,
                          2, 11, 11, 12,  4, 12,
                         12, 13, 13,  6,  6, 23,
                          6, 16, 16, 17, 17,  8,
                         17, 18,  2, 18,
                          2, 20, 20, 27, 27, 28,
                         12, 28, 12, 30, 13, 31,
                         23, 31, 23, 33, 33, 16,
                         18, 37, 37, 20, 11, 27,

                         54, 56, 56, 58, 58, 60, 60, 54,
                         54, 45, 45, 46, 46, 56,
                         58, 47, 58, 41, 58, 50,
                         60, 51, 52, 54, 54, 38,
```


rwgray@rwgrayprojects.com

```
                        38, 27, 27, 45, 46, 47,
                        47, 31, 31, 41, 41, 33,
                        33, 50, 50, 51, 51, 52, 52, 37, 37, 38,

                        28, 46, 30, 46, 30, 31,

                        17, 36, 36, 51, 51, 34, 34, 17,
                        36, 37, 33, 34
                       };

int[] RhTriaFaceMap  = { 2,  4,  6,  6,  8,  2,
                         2, 11,  4,  4, 11, 12,
                         4, 12, 13,  4, 13,  6,
                         6, 16,  8,  8, 16, 17,
                         8, 17, 18,  8, 18,  2,
                         2, 18, 37,  2, 37, 20,
                         2, 20, 27,  2, 27, 11,
                        11, 27, 28, 11, 28, 12,
                         6, 13, 31,  6, 31, 23,
                         6, 23, 33,  6, 33, 16,

                        54, 60, 58, 58, 56, 54,
                        54, 56, 45, 45, 56, 46,
                        56, 58, 47, 47, 46, 56,
                        47, 58, 41, 41, 31, 47,
                        58, 50, 33, 33, 41, 58,
                        58, 60, 51, 51, 50, 58,
                        60, 54, 52, 52, 51, 60,
                        54, 38, 37, 37, 52, 54,
                        45, 27, 38, 38, 54, 45,

                        20, 37, 38, 38, 27, 20,
                        23, 31, 41, 41, 33, 23,

                        12, 28, 46, 46, 30, 12,

                        12, 30, 31, 31, 13, 12,
                        31, 30, 46, 46, 47, 31,
                        28, 27, 45, 45, 46, 28,

                        17, 34, 51, 51, 36, 17,
                        18, 17, 36, 36, 37, 18,
                        37, 36, 51, 51, 52, 37,
                        17, 16, 33, 33, 34, 17,
                        34, 33, 50, 50, 51, 34
                       };


// 1 120 Polyhedron
int[] P120Vertices = { 1,  2,  3,  4,  5,  6,  7,  8,  9, 10,
                      11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
                      21, 22, 23, 24, 25, 26, 27, 28, 29, 30,
                      31, 32, 33, 34, 35, 36, 37, 38, 39, 40,
                      41, 42, 43, 44, 45, 46, 47, 48, 49, 50,
                      51, 52, 53, 54, 55, 56, 57, 58, 59, 60,
                      61, 62
                     };
```


rwgray@rwgrayprojects.com

```
int[] P120EdgeMap  = {
                      1, 2, 1, 4, 1, 6, 1, 8,
                      2,  3, 2,  4,
                      2,  8, 2,  9, 2, 10, 2, 11,
                      2, 18, 2, 19, 2, 20,
                      3, 4, 3, 11, 3, 12,
                      4, 5, 4, 6, 4, 12,
                      5, 6, 5, 12, 5, 13,
                      6,  7, 6,  8, 6, 13, 6, 14,
                      6, 15, 6, 16, 6, 23,
                      7, 8,
                      7, 16, 7, 17,
                      8, 9, 8, 17,
                      9, 17, 9, 18,
                     10, 11, 10, 20, 10, 27,
                     11, 12, 11, 21, 11, 27,
                     12, 13, 12, 21,
                     12, 28, 12, 29, 12, 22,
                     12, 30,
                     13, 14, 13, 22, 13, 31,
                     14, 23, 14, 31,
                     15, 16, 15, 23, 15, 33,
                     16, 17, 16, 24, 16, 33,
                     17, 18, 17, 24, 17, 25, 17, 34,
                     17, 35, 17, 36,
                     18, 19, 18, 25, 18, 37,
                     19, 20, 19, 37,
                     20, 26, 20, 27, 20, 37,
                     21, 27, 21, 28,
                     22, 30, 22, 31,
                     23, 31, 23, 32, 23, 33,
                     24, 33, 24, 34,
                     25, 36, 25, 37,
                     26, 27, 26, 37, 26, 38,
                     27, 28, 27, 38, 27, 39, 27, 44,
                     27, 45,
                     28, 29, 28, 39, 28, 46,
                     29, 30, 29, 46,
                     30, 31, 30, 40, 30, 46,
                     31, 32,
                     31, 40, 31, 41, 31, 47, 31, 48,
                     32, 33, 32, 41,
                     33, 34,
                     33, 41, 33, 42, 33, 49,
                     33, 50,
                     34, 35, 34, 42, 34, 51,
                     35, 36, 35, 51,
                     36, 37, 36, 43, 36, 51,
                     37, 38, 37, 43, 37, 52, 37, 53,
                     38, 44, 38, 53, 38, 54,
                     39, 45, 39, 46,
                     40, 46, 40, 47,
                     41, 48, 41, 49, 41, 58,
                     42, 50, 42, 51,
                     43, 51, 43, 52,
                     44, 45, 44, 54,
```


rwgray@rwgrayprojects.com

```
                        45, 55, 45, 46, 45, 54,
                        46, 47,
                        46, 55, 46, 56, 46, 57,
                        47, 48, 47, 57, 47, 58,
                        48, 58,
                        49, 50, 49, 58,
                        50, 51, 50, 58, 50, 59,
                        51, 52, 51, 59, 51, 60,
                        51, 61,
                        52, 53, 52, 61, 52, 54,
                        53, 54,
                        54, 55, 54, 56,
                        54, 60, 54, 61, 54, 62,
                        55, 56,
                        56, 57, 56, 58, 56, 62,
                        57, 58,
                        58, 59, 58, 60, 58, 62,
                        59, 60,
                        60, 61, 60, 62,
                       };

int[] P120FaceMap  = {
                          1,  2,  4,
                          2,  3,  4,
                          2, 20, 10,
                          2, 10, 11,
                          2, 11,  3,
                          3, 11, 12,
                          3, 12,  4,
                         20, 26, 27,
                         20, 27, 10,
                         10, 27, 11,
                         11, 27, 21,
                         11, 21, 12,
                         21, 27, 28,
                         12, 21, 28,
                         12, 28, 29,

                          1,  4,  6,
                          4, 12,  5,
                          4,  5,  6,
                          5, 12, 13,
                          5, 13,  6,
                          6, 13, 14,
                          6, 14, 23,
                         12, 29, 30,
                         12, 30, 22,
                         12, 22, 13,
                         13, 22, 31,
                         22, 30, 31,
                         13, 31, 14,
                         14, 31, 23,
                         23, 31, 32,

                          1,  6,  8,
                          6, 23, 15,
                          6, 15, 16,
```


rwgray@rwgrayprojects.com

```
 6, 16,  7,
 6,  7,  8,
 8,  7, 17,
 7, 16, 17,
23, 32, 33,
15, 23, 33,
16, 15, 33,
24, 16, 33,
34, 24, 33,
17, 16, 24,
17, 24, 34,
17, 34, 35,

 1,  8,  2,
 8, 17,  9,
 8,  9,  2,
 9, 17, 18,
 9, 18,  2,
 2, 18, 19,
 2, 19, 20,
17, 35, 36,
17, 36, 25,
17, 25, 18,
18, 25, 37,
25, 36, 37,
19, 18, 37,
20, 19, 37,
20, 37, 26,

27, 26, 38,
27, 38, 44,
27, 44, 45,
27, 45, 39,
27, 39, 28,
28, 39, 46,
28, 46, 29,
39, 45, 46,
38, 54, 44,

55, 45, 54,
45, 44, 54,

45, 55, 46,
46, 55, 56,
55, 54, 56,
56, 54, 62,

30, 29, 46,
30, 46, 40,
31, 30, 40,
40, 46, 47,
31, 40, 47,
31, 47, 48,
31, 48, 41,
31, 41, 32,
46, 56, 57,
47, 46, 57,
```


rwgray@rwgrayprojects.com

```
  47, 57, 58,
  48, 47, 58,
  41, 48, 58,
  57, 56, 58,
  58, 56, 62,

  33, 32, 41,
  33, 41, 49,
  33, 49, 50,
  33, 50, 42,
  33, 42, 34,
  34, 42, 51,
  42, 50, 51,
  35, 34, 51,
  49, 41, 58,
  50, 49, 58,
  50, 58, 59,
  51, 50, 59,
  51, 59, 60,
  59, 58, 60,
  60, 58, 62,

  36, 35, 51,
  36, 51, 43,
  37, 36, 43,
  43, 51, 52,
  37, 43, 52,
  37, 52, 53,
  37, 53, 38,
  37, 38, 26,
  51, 60, 61,
  52, 51, 61,
  52, 61, 54,
  53, 52, 54,
  38, 53, 54,
  54, 61, 60,
  54, 60, 62

};
```


rwgray@rwgrayprojects.com