Fracture-10

Version 0.4

Abstract

This paper presents a solution derivation and difficulty quantification for the Fracture-10 puzzle, originally design by David Pitcher.

Contents

Table of Figures

Table of Equations

List of Tables

Introduction

The Fracture-10 puzzle is a "Twisty Puzzle", a subset of permutation puzzles, originally design and built by David Pitcher in 2011.

Designer's Entry	https://www.pitcherpuzzles.com/fracture-series
Designer's Video	https://youtu.be/JWf-sdnpWA0
Museum Entry	http://twistypuzzles.com/cgi-bin/puzzle.cgi?pkey=1667

Defining the Solved State

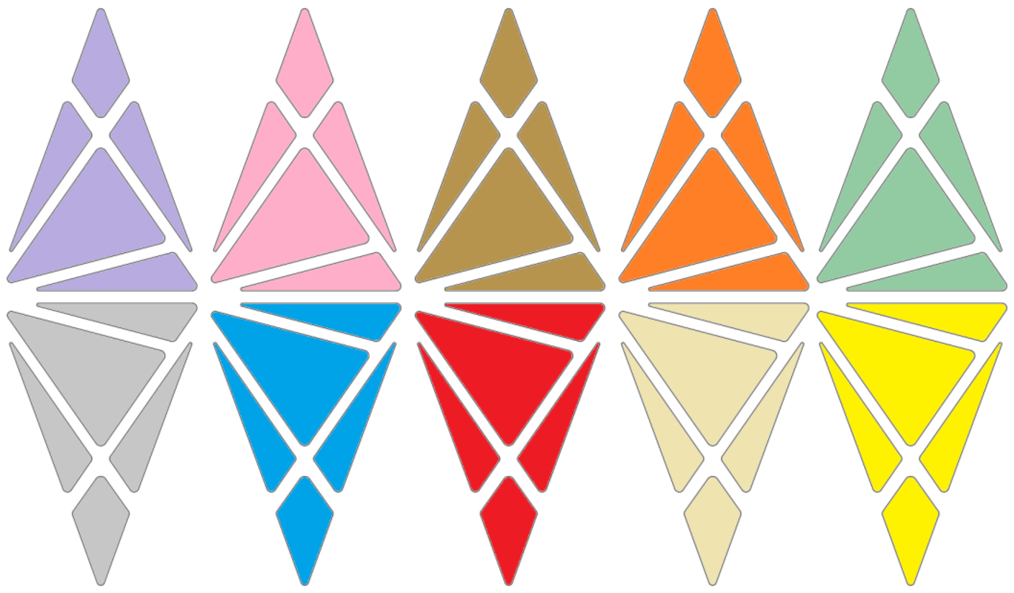
For the purposes of this paper, I'd like to define what the goal state of the puzzle is. I propose the following assumptions.

Ten-Color Assumption

The solved state is defined by each of the ten (10) faces consisting of a single and mutually exclusive color.

The purpose of this assumption is to prevent a situation in which two or more pieces cannot be uniquely identified at the piece level (i.e. without context to the relative position of the other pieces). A puzzle which doesn't meet this assumption can give rise to complications typically termed "Parity Error". There are alternative means of enforcing disambiguation at the piece level, but uniquely coloring each face is by far the most common method.

Figure 1 Ten-Color Assumption

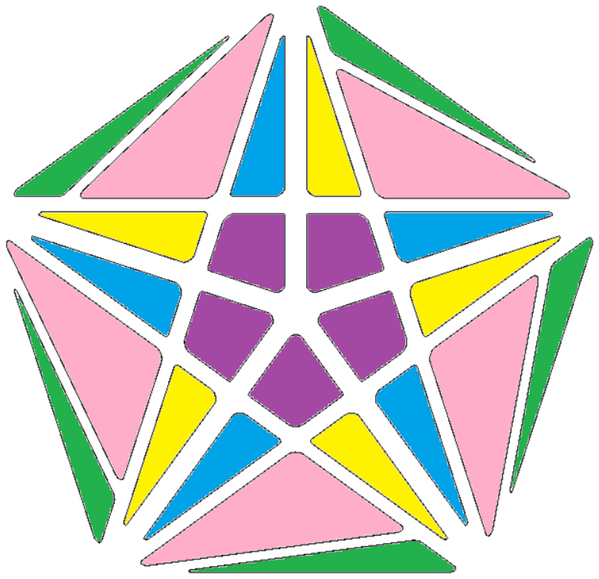


Rotational Symmetry Assumption

The orientations of the pieces in the solved position have 5-fold rotational symmetry about the polar axis of the bipyramid.

The purpose of this assumption is to simplify the notation used to convey the algorithms of this paper. If this assumption does not hold, then this paper may be used to solve to an intermediate position that is within three axis rotations of the solved solution. See Equatorial Orientation.

Figure 2 Rotational Symmetry Assumption



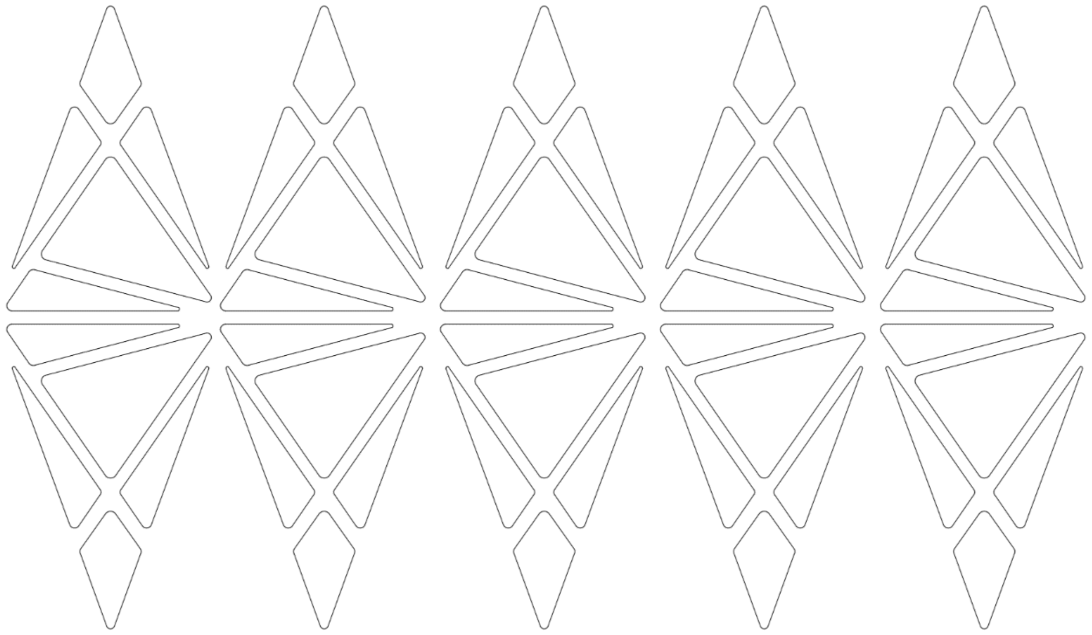
Interior Pieces Assumption

All unstickered pieces (interior pieces) are neither given an orientation nor a permutation and can always be considered to be in their solved states.

Global Orientation

The first of the three phases in solving the Fracture-10 puzzle is to orient the pieces of the scrambled puzzle into a state congruent to the orientation of the pieces of the solved puzzle. This would be analogous to a "stickerless solve". Global Orientation can always be achieved in five or fewer twists. The process of solving Global Orientation is two-fold: first solving Polar Orientation followed by solving for Equatorial Orientation while maintaining Polar Orientation.

Figure 3 Orientation of the pieces of the solved puzzle.

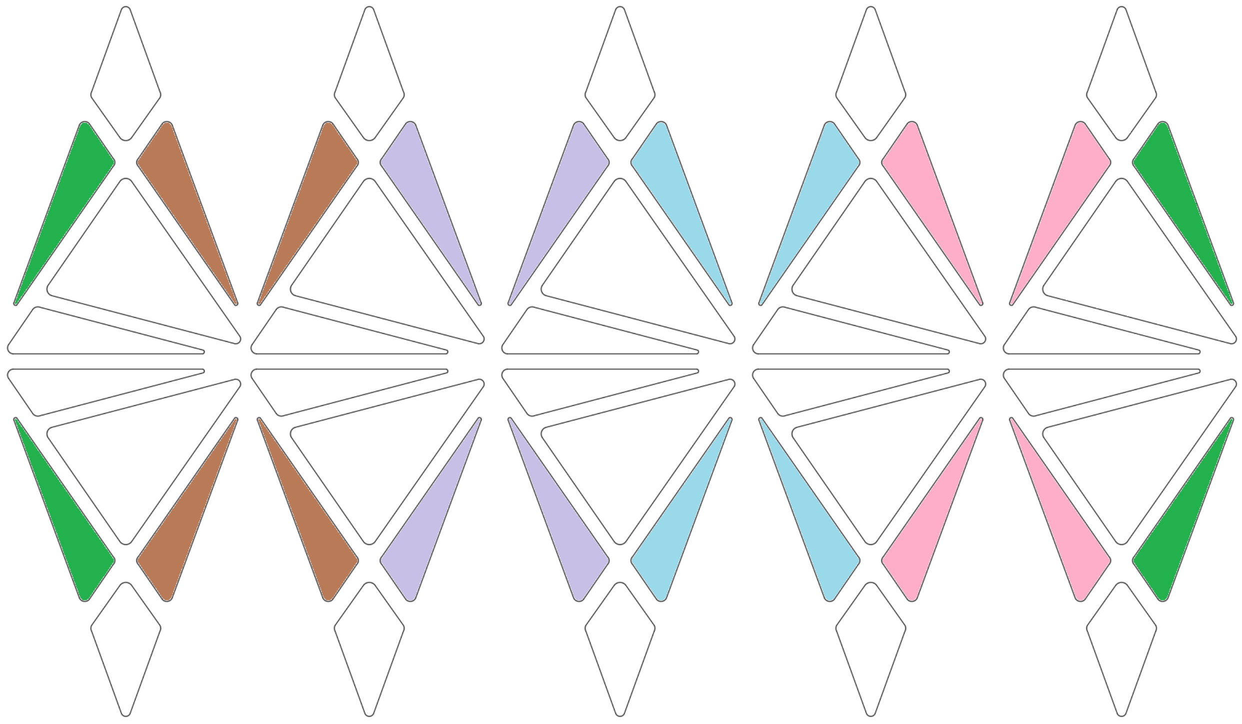


Polar Orientation

Defining Polar Orientation

I define the Polar Orientation to be the state of orientation of all the edge pieces that touch one of the two poles.

Figure 4 Example of Correct Polar Orientation

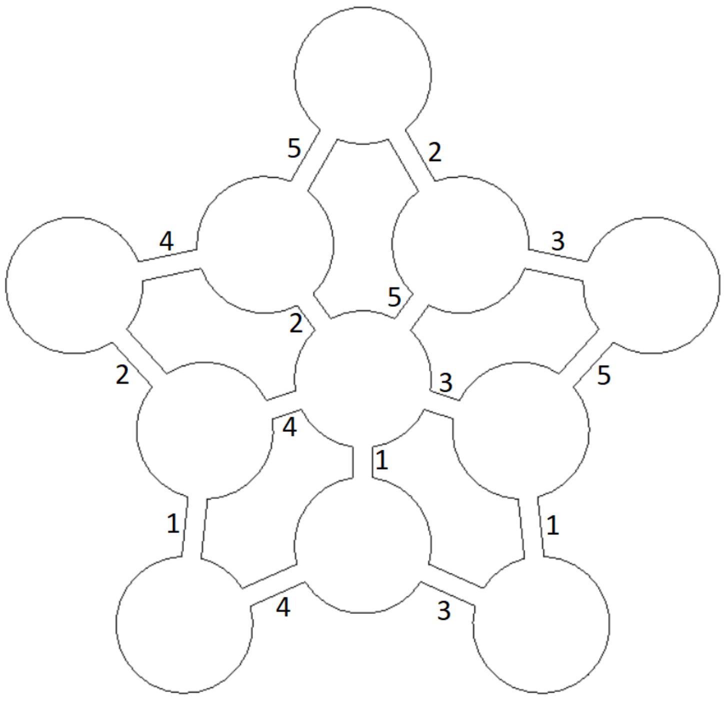


In any given scrambled state, there can exist either two (2), three (3), or five (5) of the equatorial axes that can be spun. The number of available ("spinnable") axes is defined exclusively by the Polar Orientation. The goal of this sub-section is to manipulate the puzzle into a state where any of the five axes can be spun.

The Size of the Axis Availability Space

In order to derive a method to achieve this, we will first explore how axis blocking occurs. If we were to start with a Polar Orientation in which all five axes are spinnable, and we rotate any one of the axes by 90-degrees (either clockwise or counterclockwise), we would *likely* be blocking both the axes immediately adjacent to the spun axis. With two axes now blocked, we have reduced our spinnable axis count from five to three. If we were to further spin the selected axis by another 90-degrees we would necessarily have unblocked any previous blocking and returned to five spinnable axes. Instead of having further spun the selected axis, if one of the other two available unspun axes were instead rotated by-90 degrees (either direction), the other previously available axis will *likely* now be blocked. This further reduces our spinnable axes count down to just the two axes that are each currently spun 90-degrees; the remaining three axes are blocked. From there the only two axes that can spin are the axes we have already spun. If we spin either one of those two axes by an additional 90-degrees (either direction), one of the previously blocked axes will now become unblocked. If we instead were to spin both of the spinnable axes by 90-degrees (either direction) then we have returned to the center vertex of the Axis Availability graph (see below); a state where all five axes are once again spinnable. The entire space of possible Axis Availability (ignoring loops) is covered by applying rotational symmetry to the previously described generic case. Furthermore, the radius of the Axis Availability graph is 2; a desired Polar Orientation which allows all five axes to spin (the center vertex of the Axis Availability Graph) is at most 2 spins away from any possible starting Polar Orientation.

Figure 5 Graph of Axis Availability (Excluding Loops)

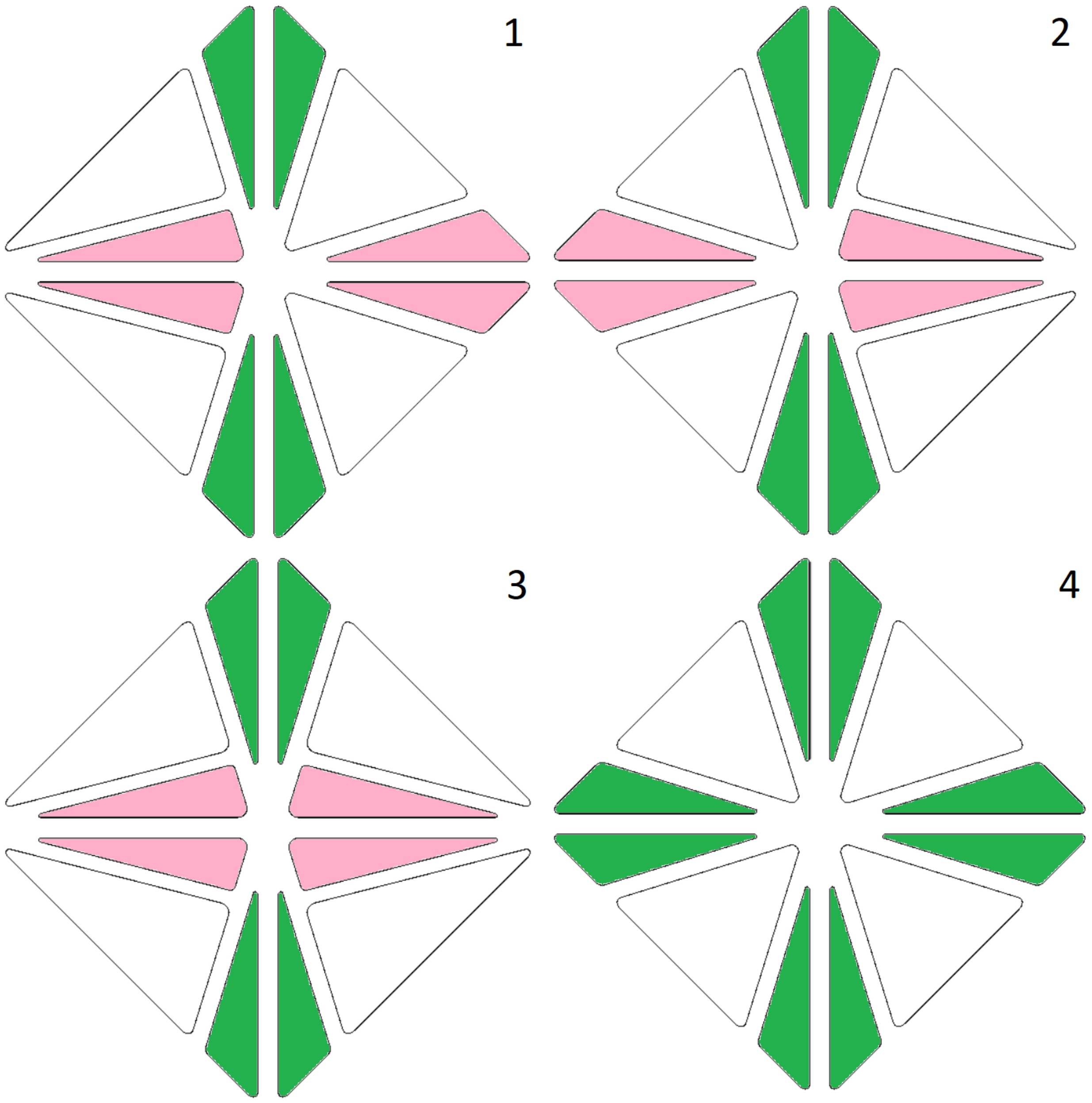


Assuming the axes of the puzzle are numbered sequentially around the equator (see Figure 18 or Figure 9), Figure 5 represents the graph of axis availability excluding loops. Traversing along any edge is analogous to rotating the edge's labeled axis by 90-degrees (in either direction). The degree of each vertex, when disregarding loops, corresponds to the number of available axes.

Determination of Correct Polar Orientation

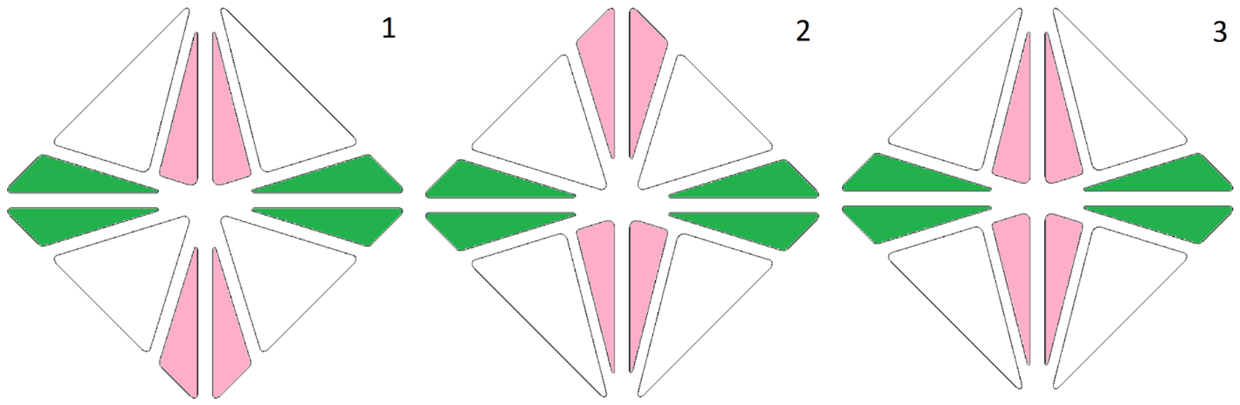
Polar Orientation was earlier defined as the state of orientation of all the edge pieces that touch one of the two poles. Our desired Polar Orientation is achieved when all five pairs of polarly opposed edges converge to a point at the equator.

Figure 6 Correct Polar Orientation



These are the four possible configurations which express correctly oriented pairs of polarly opposed edge pieces (green). Rotating any of these configurations by 180 degrees will maintain desired orientation. Notice the configuration consisting of four green edges: This configuration maintains desired orientation when rotated by 90-degrees (either direction). This results in loops in the Graph of Axis Availability, specifically five loops at the central vertex and two loops at each of the five 3rd-degree vertices. This is also the source of the two qualified "*likely*" blockages in "The Size of the Axis Availability Space".

Figure 7 Incorrect Polar Orientation



These are the three possible incorrect orientations of pairs of polarly opposed edge pieces (pink). Each of these can be converted to a correct orientation by rotating 90-degrees (in either direction).

Solving Polar Orientation

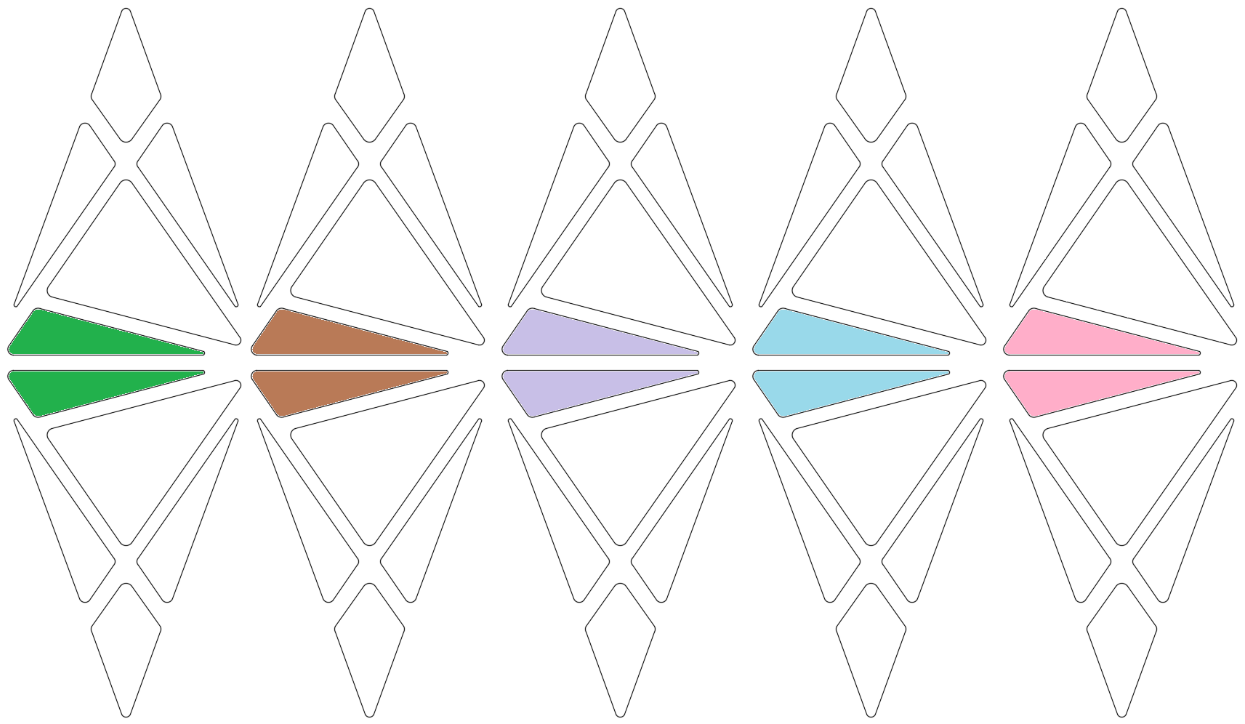
All unavailable (blocked) axes will necessarily have the correct Polar Orientation. Available (unblocked) axes with incorrect orientation may be corrected by a 90-degree rotation (in either direction). To maintain correct Polar Orientation when solving for Equatorial Orientation, only 180-degree turns should be used.

Equatorial Orientation

Defining Equatorial Orientation

I define Equatorial Orientation to be the orientation of all of the edge pieces located about the equator of the puzzle.

Figure 8 Example of Correct Equatorial Orientation



The goal of this sub-section is to manipulate the puzzle into a state where the puzzle achieves five-fold rotational symmetry about the polar axis, as seen in Figure 2 or Figure 8.

The Size of the Equatorial Orientation Space

Each equatorial edge can exist in one of two possible orientations, which I will term "left" or "right" when viewed from the "top pole". A rigorous definition of what makes an edge piece "left" or "right" with respect to the vantage that is arbitrarily determined to be the "top pole" isn't required, but consistency within your own naming system is required.

If we assume that the orientation of each equatorial edge is independent from the orientation of the other equatorial edges, and that each equatorial edge can exist in one of two binary states ("left" or "right"), then there are at most 2^5 (or 32) possible configurations. The breakdown of the configurations is given below.

Equation 1 Combination Equation

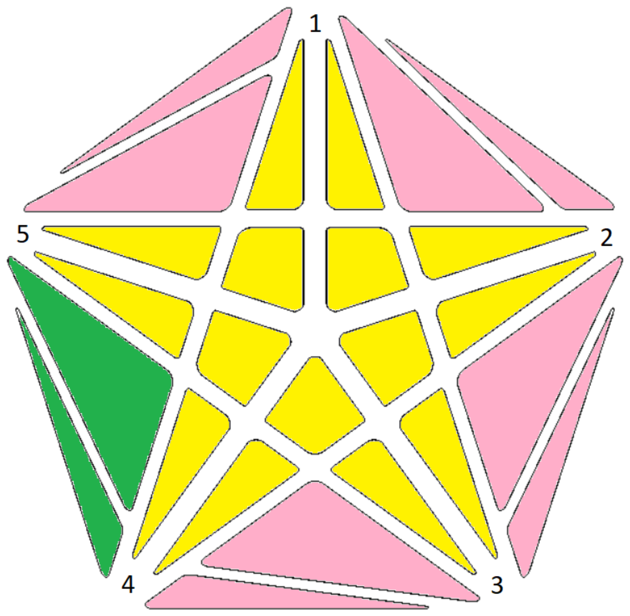
$$_nC_r = \frac{n!}{r!\,(n-r)!}$$

Table 1 Edge Orientation Configurations

Left	Right	Formula	Total
5	0	$_5C_5$	1
4	1	$_5C_4$	5
3	2	$_5C_3$	10
2	3	$_5C_2$	10
1	4	$_5C_1$	5
0	5	$_5C_0$	1

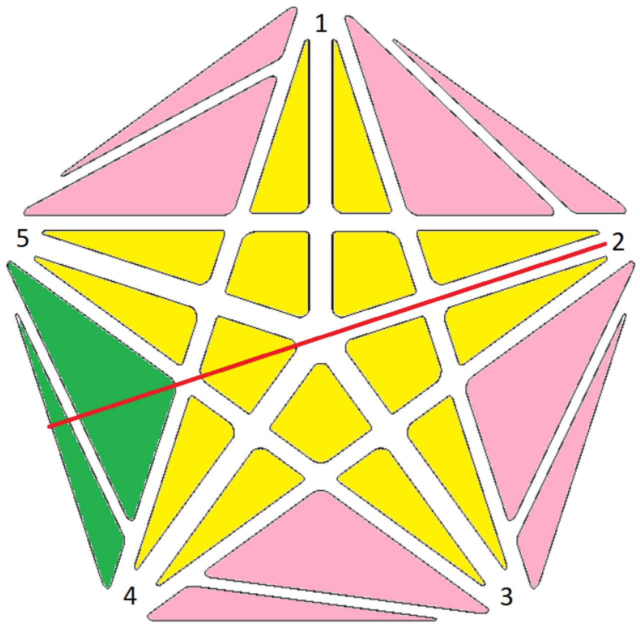
Take for example the following configuration:

Figure 9 Sample Equatorial Edges Configuration



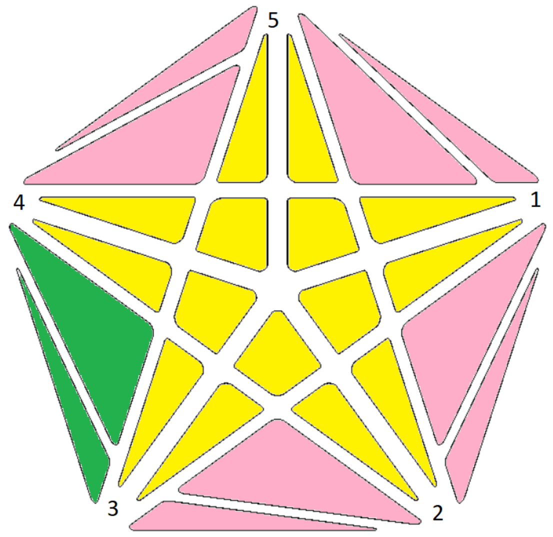
Let “Green” edges be defined as “Right” oriented edges and conversely let “Pink” edges be defined as “Left” oriented edges. Also, let each edge be uniquely identified by the axis which passes through it.

Figure 10 Sample Equatorial Edges Configuration, Axis 2



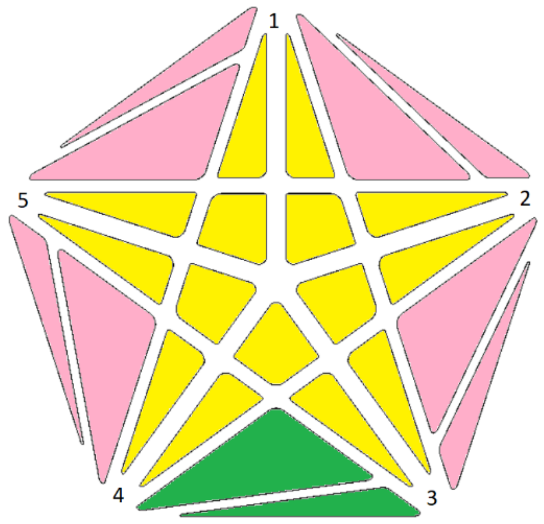
In the above Figure 9, edge #2 is “Right” and all four other edges are “Left”. This would be an example of the 4-Left-1-Right row of Table 1. Now, without moving the puzzle, renumber the axes as follows (this is further covered in Coordinate Frame Transformations):

Figure 11 Sample Equatorial Edges Configuration, Renumbered



Next, perform a rigid body rotation of the whole puzzle about the polar axis in order to position axis 1 at the top of the image.

Figure 12 Sample Equatorial Edges Configuration, Renumbered and Reoriented



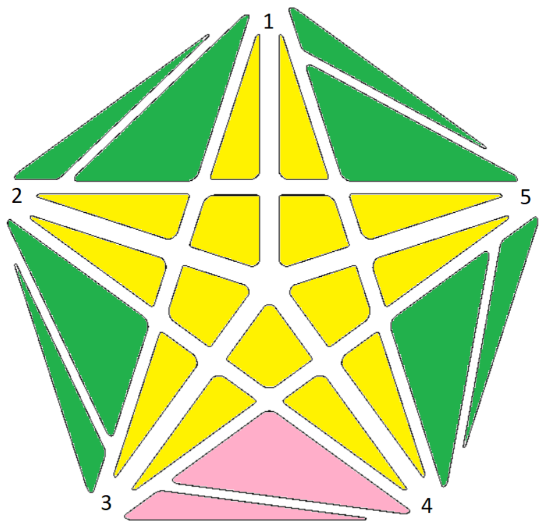
In the above "new" configuration, edge #1 is "Right" and all other edges are "Left". Therefore, using rotational symmetry, the table of possible edge orientation configurations is reduced as follows:

Table 2 Edge Orientation Configurations, Rotational Simplification

Left	Right	Formula	Total
5	0	${}_5C_5$	1
4	1	${}_5C_4 / 5$	1
3	2	${}_5C_3 / 5$	2
2	3	${}_5C_2 / 5$	2
1	4	${}_5C_1 / 5$	1
0	5	${}_5C_0$	1

Furthermore, if we subsequently perform another rigid body rotation to view the puzzle configuration in Figure 12 from the vantage of the other pole:

Figure 13 Sample Equatorial Edges Configuration, Reoriented - Renumbered - Reoriented

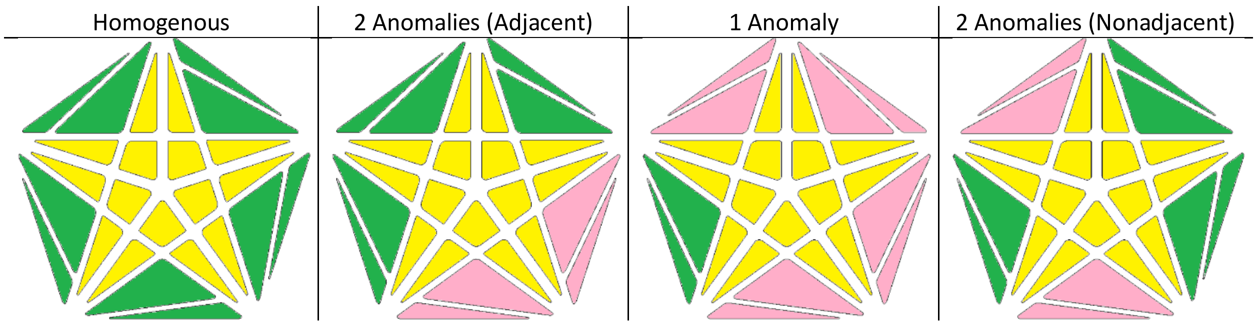


We now have another "new" configuration in which the quantity of "Right" and "Left" edges are swapped: edge #1 is now "Left" and all other edges are "Right". Therefore, mirror symmetry about the plane of the equator further reduces the possible edge orientations. Mirror symmetry also muddies the "Right" vs "Left" edge orientation nomenclature as one turns into the other when mirrored. As there are an odd number of equatorial edges, either "Right" or "Left" edges will have a majority when viewed from a given polar vantage. Let the minority orientation now be referred to as the anomalous orientation. Anomalous edges are preserved under mirror symmetry about the equatorial plane.

Table 3 Edge Orientation Configurations, Rotational and Mirror Simplifications

Anomalies	Formula	Total
0	${}_5C_5 = {}_5C_0$	1
1	${}_5C_4 / 5 = {}_5C_1 / 5$	1
2	${}_5C_3 / 5 = {}_5C_2 / 5$	2

Figure 14 The Four Unique Equatorial Orientations



The above Figure 14 demonstrates the four unique Equatorial Orientations when accounting for symmetries. Any puzzle with the correct Polar Orientation can be expressed as one of the above four Equatorial Orientations through only rigid body rotations.

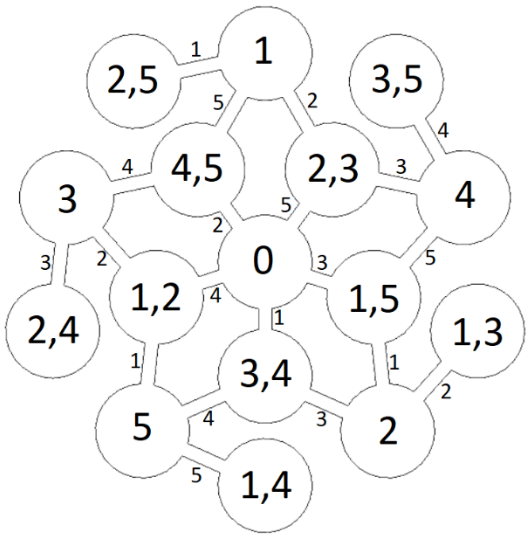
Solving Equatorial Orientation

Correct Equatorial Orientation is achieved when there are no anomalous edges. To guarantee preservation of the Polar Orientation, only 180-degree turns should be used. Using the axes notation from Figure 9 (axis 1 is at the top of the image and the axes increase clockwise), the “2 Anomalies (Nonadjacent)” state of Figure 14 can be transformed to the “Homogenous” state by performing sequential 180-degree turns on the following axes:

1. Rotation about axis 2 will transform “2 Anomalies (Nonadjacent)” into “1 Anomaly”.
2. Rotation about axis 1 will transform “1 Anomaly” into “2 Anomalies (Adjacent)”.
3. Rotation about axis 3 will convert “2 Anomalies (Adjacent)” into “Homogenous”.

A less general graph is given below:

Figure 15 Graph of Anomalous Edges (Excluding Loops)

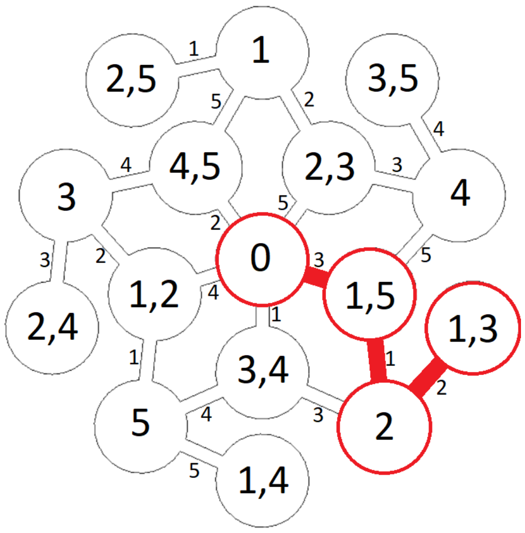


The graph’s vertices are labeled with the anomalous equatorial edges (“0” represents null or the Homogenous state) and the graph’s edges are labeled by the axis which must be rotated 180-degrees to traverse said edge. This graph has a radius of 3, which means correct Equatorial Orientation can be established in at most three moves. From inside to outside:

1. The center of the graph is the “Homogenous” state
2. The inner band of 3rd-degree vertices corresponds to the “2 Anomalies (Adjacent)” state
3. The outer band of 3rd-degree vertices corresponds to the “1 Anomaly” state
4. The 1st-degree vertices correspond to the “2 anomalies (Nonadjacent)” state.

Since the puzzle currently expresses correct Polar Orientation, all axes are always available for 180-degree turns, as evidenced by the 5th-degree center vertex. All other vertices of the graph are represented as less than 5th-degree due to loops not being expressed. All missing axes at any given vertex of this graph are necessarily loops.

Figure 16 Example path

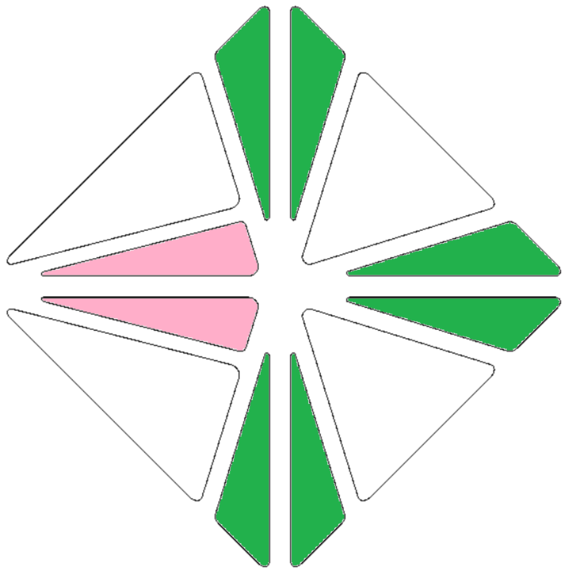


The above example path from Figure 16 corresponds to the generalized example from Figure 14.

Standard Orientation

A Standard Orientation frame is needed for consistency in the following sections. With the poles and equator oriented, the puzzle now exhibits one of two possible Global Orientations: The orientation seen in Figure 1 or the orientation seen in Figure 3. I have (arbitrarily) chosen to use the orientation of Figure 1 which could be defined as follows: The divergent edge at each equatorial vertex shall be on the left (or West).

Figure 17 The divergent edge is on the left



I will refer to this orientation as the Standard Orientation. Each chunk of incremental progress will necessarily begin and end in Standard Orientation and all instructions will be given with respect to a puzzle globally oriented to Standard Orientation.

Global Permutation

The goal of this section is to relocate the pieces about the puzzle while maintaining (starting and finishing each bit of incremental progress in) Standard Orientation.

Notation

The algorithms will utilize a common notation further described in this section. An example of one of these algorithms is as follows:

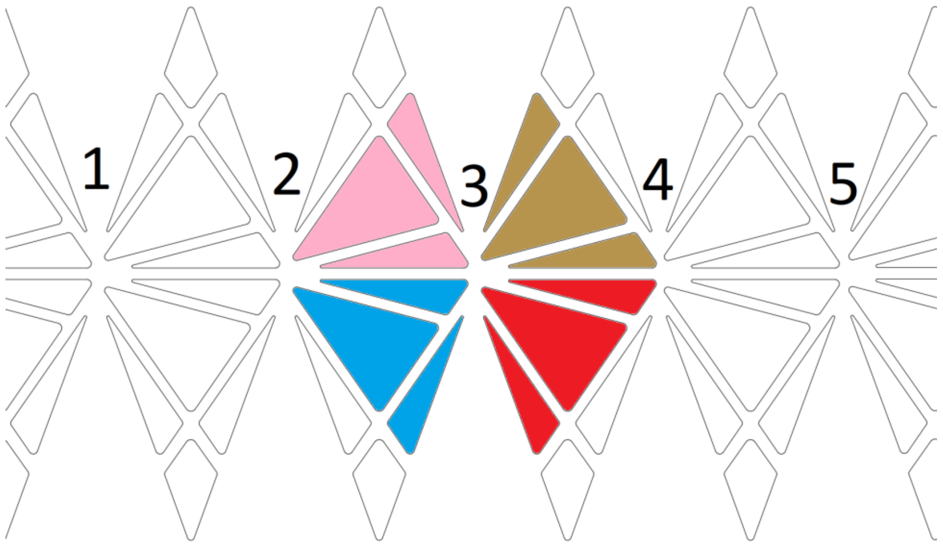
Equation 7 Impure Generic 4-Cycles, n=3, Reduced

$$G4_3^p = \{2''\}\,(3)^p\,\{2''\};\quad p \in \{-1,1,2\}$$

Axes Numbering

The numbers which have the superscripts of zero, one, or two apostrophes correspond to the axes of rotation. In the above Impure Generic 4-Cycle, the only axes used are axis 2 and axis 3. The numbering scheme is as follows:

Figure 18 Axes Numbering

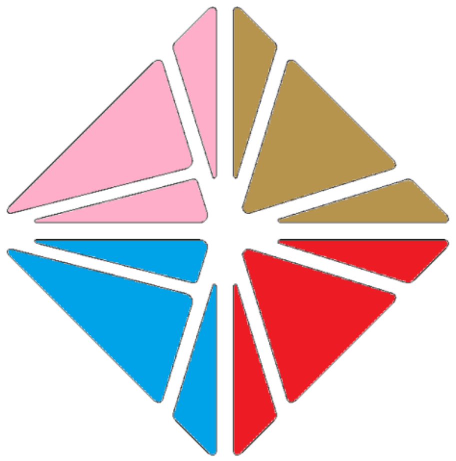


Above is a flattened sticker diagram with Axis 3 colored. The axes increase from left to right in Standard Orientation with axis 5 arithmetically overflowing back to axis 1.

Axes Superscripts

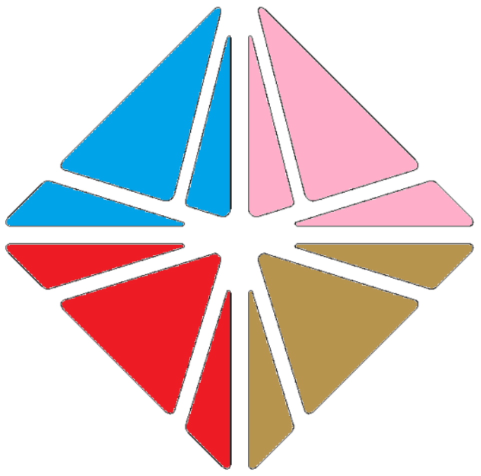
“Stops” are the useful angular positions a given axis may be rotated to; positions which might not impede the rotation of adjacent axes. All five equatorial axes have four stops, each located 90-degrees apart from one another. Axis 3 is isolated below:

Figure 19 Axis 3 Isolated



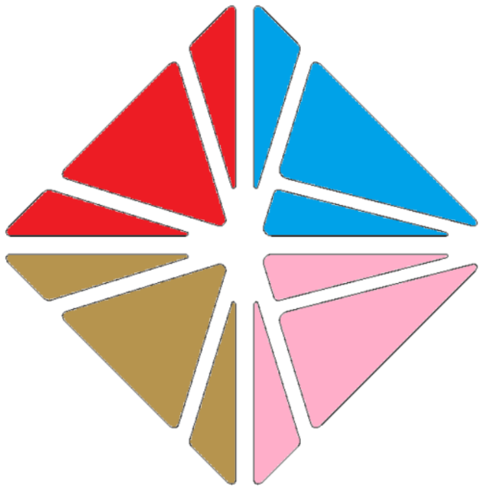
Rotating axis 3 by 90-degrees clockwise results in the following:

Figure 20 Axis 3 Rotated Clockwise 90-Degrees



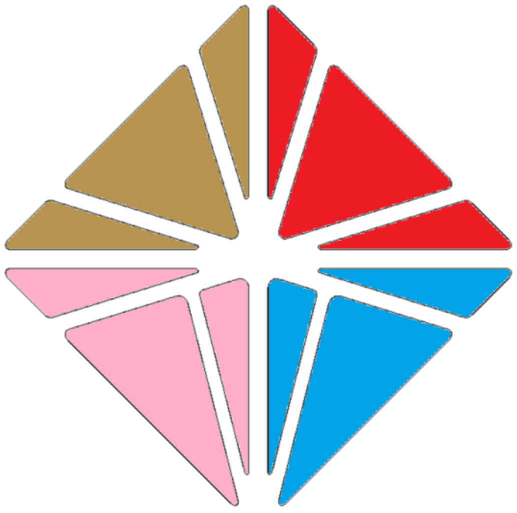
The turn can be progressed to a second stop by performing another 90-degree clockwise rotation.

Figure 21 Axis 3 Rotated 180-degrees



Lastly, the turn can be progressed to the final unique stop with a third 90-degree clockwise turn.

Figure 22 Axis 3 Rotated 90-degrees Counterclockwise



Rotating a further 90-degrees clockwise would return the axis to its starting state: the original stop. The axes of this puzzle do not have a means of recording their absolute position. Applied relative rotational displacement is congruent to the applied absolute rotational displacement modulo 4. This is to say that three clockwise 90-degree turns are congruent to negative one clockwise 90-degree turns (a counterclockwise 90-degree turn). Similarly, a half turn may be expressed as two 90-degree turns either clockwise or counterclockwise.

Table 4 Superscript Notation

Superscript	(none)	‘ (apostrophe)	“ (double apostrophe)
Interpretation	90-Degrees Clockwise	90-Degrees Counterclockwise	180-Degree Turn (Either Direction)

Grouping Characters

Parentheses (), braces {}, and brackets [] are used for both grouping and sub-cycle iteration.

Grouping

A parenthetical, bracketed, or braced portion conveys grouping. Conjugates (which take the form $X\ Y\ X^{-1}$) are denoted with braces: $\{X\}\ Y\ \{X^{-1}\}$ or its equivalent $\{X, Y\}$. Commutators (which take the form of either $X\ Y\ X^{-1}\ Y^{-1}$) are denoted by brackets: $[X][Y][X^{-1}][Y^{-1}]$ or its equivalent $[X, Y]$.

Sub-cycle Iteration

Sub-cycle iteration is denoted by a grouping raised to a superscript. The grouping (the instructions contained within the parentheses, braces, or brackets) should be completed in its entirety before being iteratively repeated. The total number of iterations is denoted by the grouping’s superscript. A grouping without a superscript has a default superscript of one and does not get repeated.

Coordinate Frame Transformations

Many of the following "generic" cycles are nominally expressed about an axis, frequently axis 3 which tends to yield visually concise Movement Charts. The onus is on the solver to index the axes to produce the desired piece-cycles. Said differently, the choice of which axis is labeled "axis 3" is a critical choice made by the solver at the start of each algorithm and the chosen "axis 3" (coordinate frame) will likely be different between algorithms.

Construction of larger algorithms by chaining base algorithms together, such as constructing a Setup-BaseAlgorithm-Setup^{-1} conjugate, will likely require coordinate frame transformations of some or all of the constituent algorithms (setup algorithm, base algorithm, and inverse of the setup algorithm) in order to maintain a constant coordinate frame in the constructed algorithm. The generic cycles can have their coordinate frames transformed by arithmetically shifting all axis numbering by a chosen constant (where 1 arithmetically underflows to 5 and 5 arithmetically overflows to 1). An example is given below where algorithm $\mathrm{G4}_3^{\mathrm{p}}$ has its axes shifted by +3 to be centered about axis 1.

Equation 7 Impure Generic 4-Cycles, n=3, Reduced

$$G4_3^p = \{2''\}\,(3)^p\,\{2''\};\quad p \in \{-1,1,2\}$$

Equation 2 Impure Generic 4-Cycles, Shifted +3

$$G4_1^p = \{5''\}\,(1)^p\,\{5''\};\quad p|p(mod)4\ \neq 0$$

Cycle Inverses

The inverse of any one move is to rotate the same axis to the same magnitude but in the opposite direction. For example, the inverse to turning axis 2 90-degrees counterclockwise ($2'$) would be to turn axis 2 90-degrees clockwise (2). The inverse of a sequence of moves is performed by reversing the sequence order and inverting the direction of each individual move. For example, the inverse of turning axis 2 90-degrees clockwise followed by turning axis 3 90-degrees counterclockwise (23') would be to turn axis 3 90-degrees clockwise followed by axis 2 by 90-degrees counterclockwise (32'). The full set of moves could be written as ($23'$) ($32'$).

Equation 3 Reduction of Inverses to Null

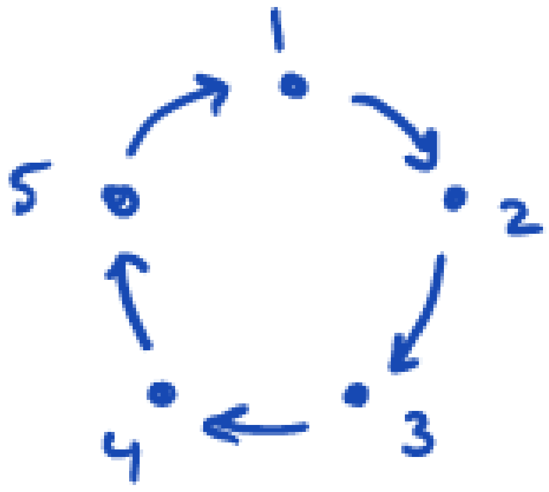
$$(23')\,(32') = 23'32' = 23^0 2' = 22' = 2^0 = Null$$

Note, a 180-degree rotation is its own inverse as half turns are not direction specific.

Cycle Notation

Cycle notation expresses the remapping of pieces under a permutation. Put differently, cycle notation conveys the result of performing a given algorithm. While denoted similar to Grouping parentheses, cycle notation is distinguishable by commas partitioning each member and (in this paper) only containing letters. An example is as follows:

Figure 23 Cycle Arrow Form



The mapping of Figure 23 could be expressed with $(1, 2, 3, 4, 5)$; each element maps to the element to its right with the last element mapping to the first element. Cycle form isn't unique. The above cycle could equivalently be expressed as (2, 3, 4, 5, 1) among others. While any element may appear first in the cycle, the order in which the elements appear must be preserved.

Constructing Permutation Algorithms

Conjugates of the form $\{X\}\, Y\, \{X^{-1}\}$ or $\{X, Y\}$ and Commutators of the form $[X][Y][X^{-1}][Y^{-1}]$ or $[X, Y]$ are the foundation of most useful algorithms.

Simplest Conjugate

The simplest conjugate sets X and Y to a single axis turn. Allow both X and Y to be any of the 5 axes and to be rotated to any multiple of 90-degrees. These conjugates are generically expressed by the 9 fields below:

Table 5 Generic Simplest Conjugates

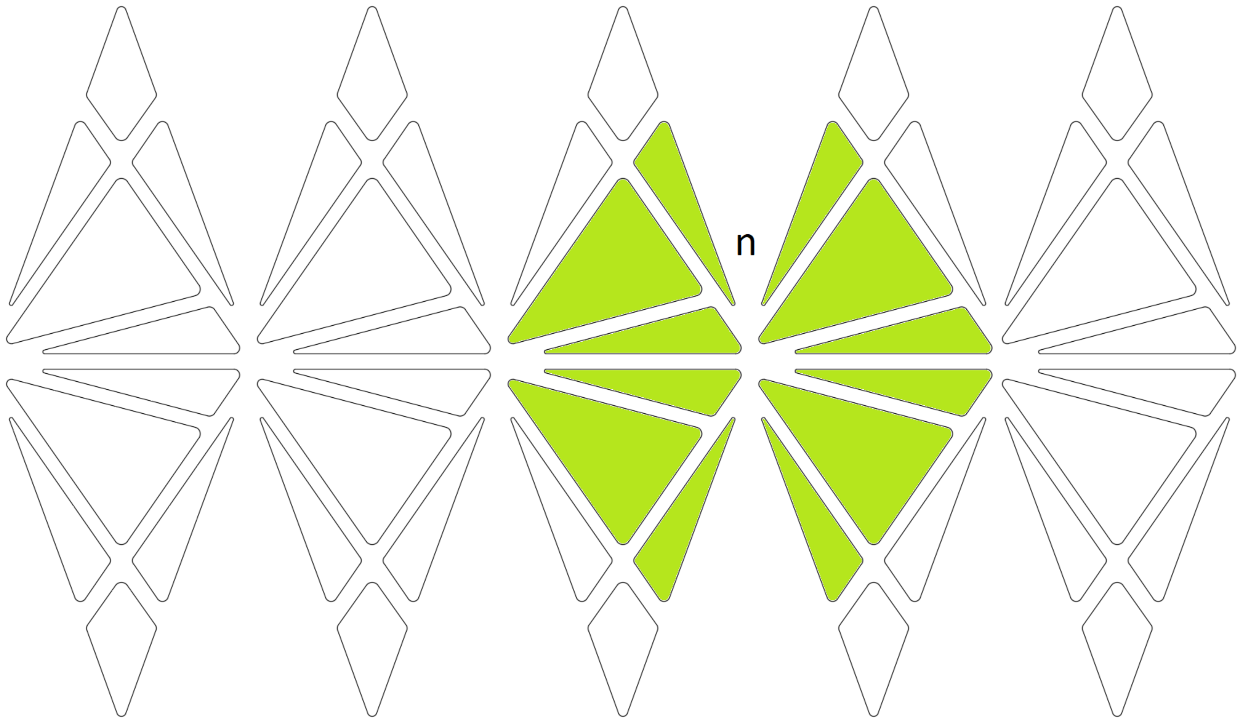
	X	Y	Result of $\{X\}\, Y\, \{X^{-1}\}$
1	$(n)^m; m \in \mathbb{Z}$	$(n)^p;\ p \in \mathbb{Z}$	$(n)^p \equiv Null$
2	$(n)^m; m \in \mathbb{Z}$	$(n \pm 2)^p;\ p \in \mathbb{Z}$	$(n \pm 2)^p \equiv Null$
3	$(n)^m; m \| m(mod)4 = 0$	$(n \pm 1)^p;\ p \in \mathbb{Z}$	$(n \pm 1)^p \equiv Null$
4	$(n)^m; m \in \{2\mathbb{Z} + 1\}$	$(n \pm 1)^p;\ p \mid p(mod)4 = 0$	Null
5	$(n)^m; m \in \{2\mathbb{Z} + 1\}$	$(n \pm 1)^p;\ p \mid p(mod)4 \neq 0$	Error
6	$(n)^m;\ m \| m(mod)4 = \pm 2$	$(n - 1)^p;\ p \mid p(mod)4 = \pm 2$	Equation 8
7	$(n)^m;\ m \| m(mod)4 = \pm 2$	$(n - 1)^p;\ p \in \{2\mathbb{Z} + 1\}$	Error
8	$(n)^m;\ m \| m(mod)4 = \pm 2$	$(n + 1)^p;\ p \mid p(mod)4 = 0$	Null
9	$(n)^m;\ m \| m(mod)4 = \pm 2$	$(n + 1)^p;\ p \mid p(mod)4 \neq 0$	Equation 7

Below are a set of Venn style diagrams conveying the overlap (green) between the pieces affected by X (blue) and Y (yellow) for each of the 9 generic fields.

Field 1

$(n)^m; m \in \mathbb{Z}$	$(n)^p;\ p \in \mathbb{Z}$	$(n)^p \equiv Null$

Figure 24 Field 1 Venn Diagram



Axis "n" is rotated clockwise by $(m + p - m)$ stops, which reduces to p stops, and is congruent to $p(mod)4$ stops. Each permutation algorithm is required to begin and end in Standard Orientation. The only exponents p that end in Standard Orientation are exponents $p \mid p(mod)4 = 0$. $(n)^0$ reduces to Null.

Field 2

$(n)^m; m \in \mathbb{Z}$	$(n \pm 2)^p;\ p \in \mathbb{Z}$	$(n \pm 2)^p \equiv Null$

Figure 25 Field 2 Venn Diagram, n+2

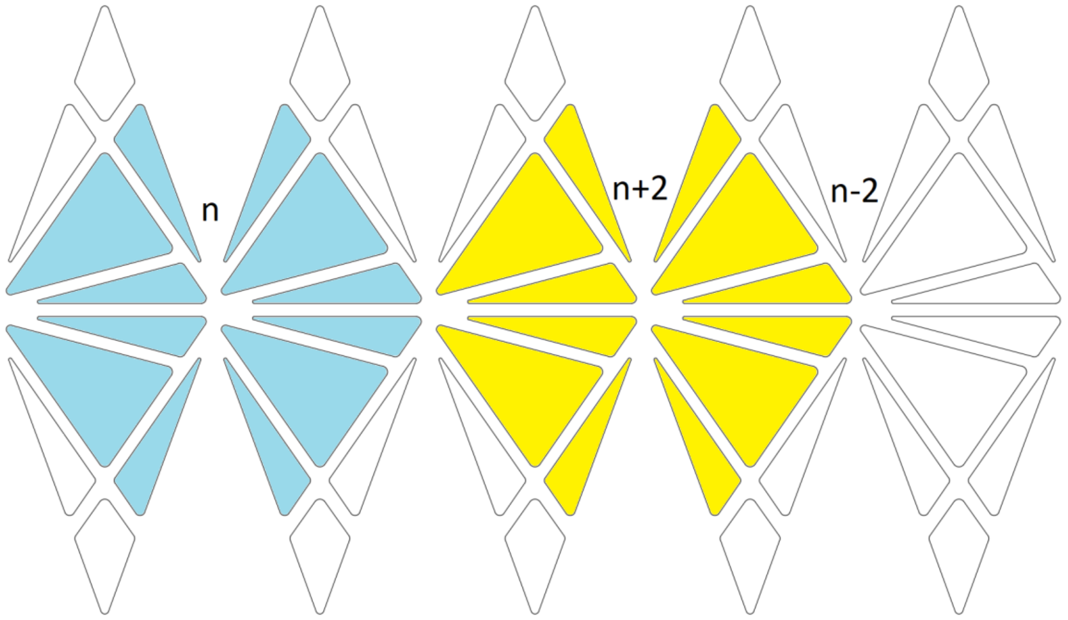
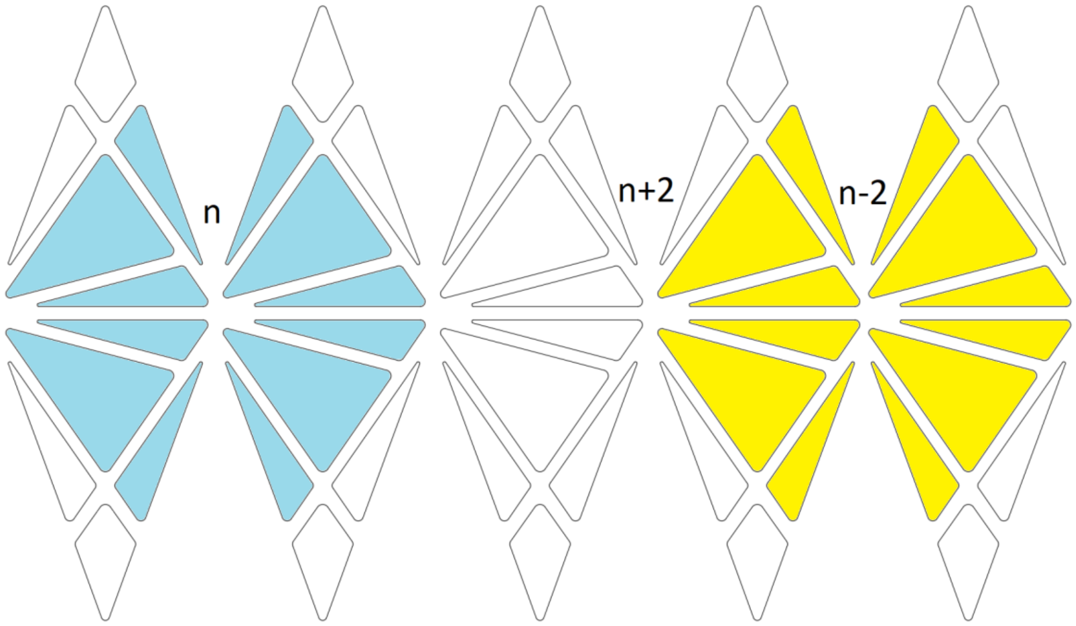


Figure 26 Field 2 Venn Diagram, n-2

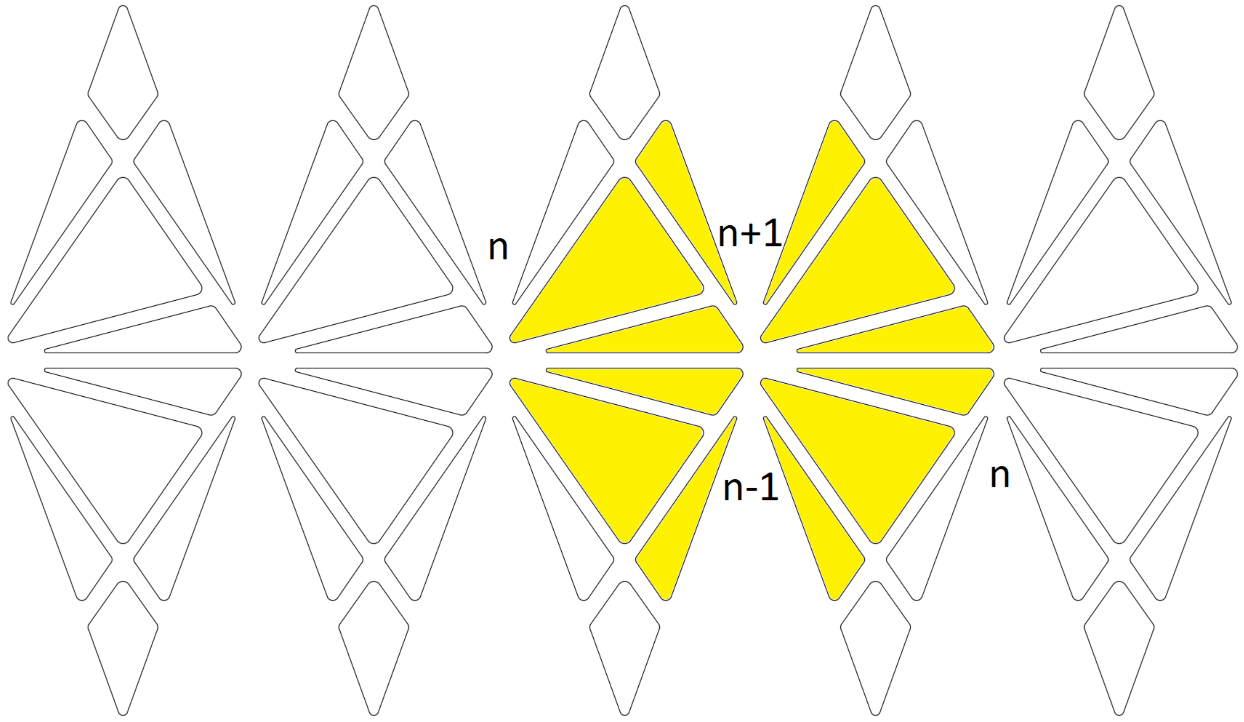


Axis "n" is rotated clockwise by $(m - m)$ stops, or $(n)^0$, which reduces to Null. Axis "$n \pm 2$" is rotated clockwise by p stops, and is congruent to a rotation of $p(mod)4$ stops. Each permutation algorithm is required to begin and end in Standard Orientation. The only exponents p that end in Standard Orientation are exponents $p \mid p(mod)4 = 0$ which is congruent to $(n \pm 2)^0$ which reduces to Null.

Field 3

$(n)^m; m\|m(mod)4 = 0$	$(n \pm 1)^p;\ p \in \mathbb{Z}$	$(n \pm 1)^p \equiv Null$

Figure 27 Field 3 Venn Diagram

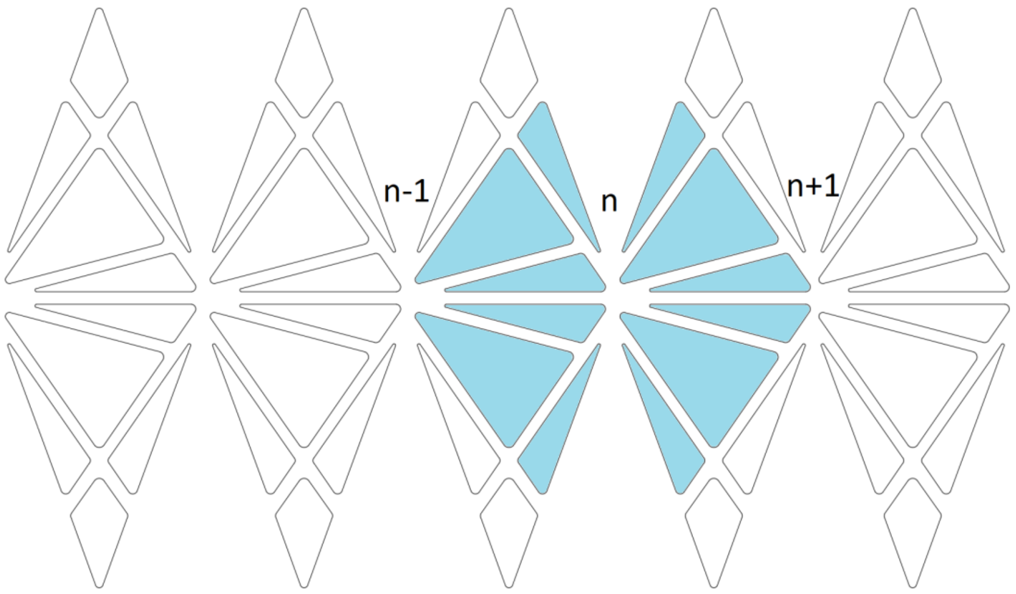


Axis “n” is rotated clockwise by an amount congruent to 0 stops, or $(n)^0$, which reduces to Null. Axis "$n \pm 1$" is rotated clockwise by p stops, and is congruent to a rotation of $p(mod)4$ stops. Each permutation algorithm is required to begin and end in Standard Orientation. The only exponents p that end in Standard Orientation are exponents $p \mid p(mod)4 = 0$ which is congruent to $(n \pm 1)^0$ which reduces to Null.

Field 4

$(n)^m; m \in \{2\mathbb{Z} + 1\}$	$(n \pm 1)^p;\ p \mid p(mod)4 = 0$	Null

Figure 28 Field 4 Venn Diagram

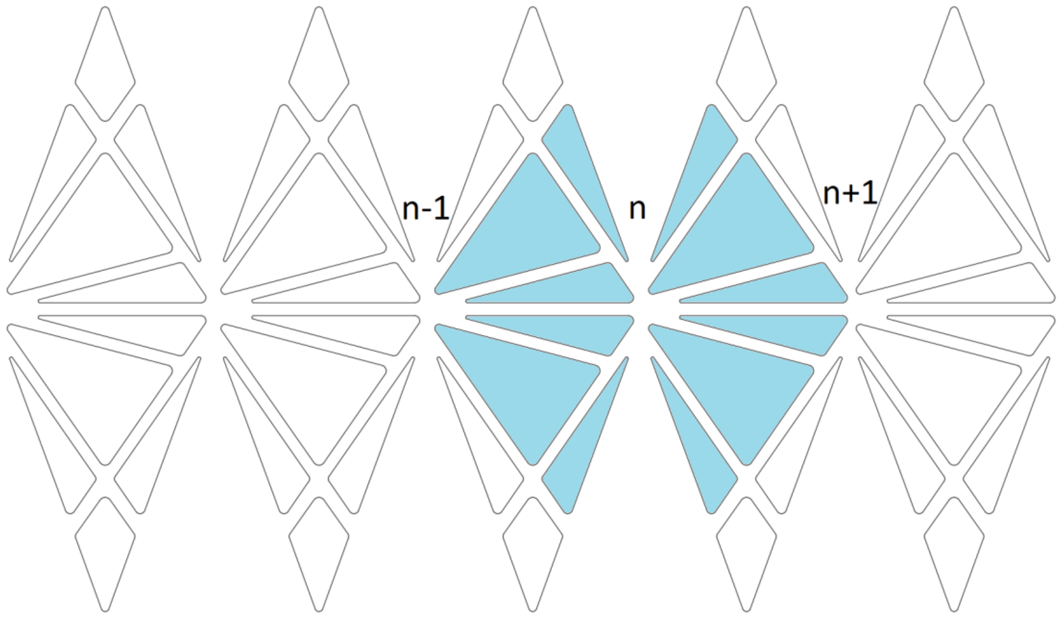


Axis "$n \pm 1$" is rotated clockwise by 0 stops, or $(n \pm 1)^0$, which reduces to Null. Axis "n" is rotated clockwise by $(m - m)$ stops, or $(n)^0$, which reduces to Null.

Field 8

$(n)^m; m\|m(mod)4 = \pm 2$	$(n+1)^p;\ p \mid p(mod)4 = 0$	Null

Figure 29 Field 8 Venn Diagram



Axis "$n \pm 1$" is rotated clockwise by 0 stops, or $(n \pm 1)^0$, and reduces to Null. Axis "n" is rotated clockwise by $(m - m)$ stops, or $(n)^0$, and reduces to Null.

Field 5

$(n)^m; m \in \{2\mathbb{Z}+1\}$	$(n \pm 1)^p;\ p \mid p(mod)4 \neq 0$	Error

Each permutation algorithm is required to begin and end in Standard Orientation. When starting in Standard Orientation, Polar Orientation is only maintained under an initial multiple of 180-degree turn applied to any axis. X of this {X,Y} conjugate breaks Polar Orientation and would not allow Y to be performed.

Field 7

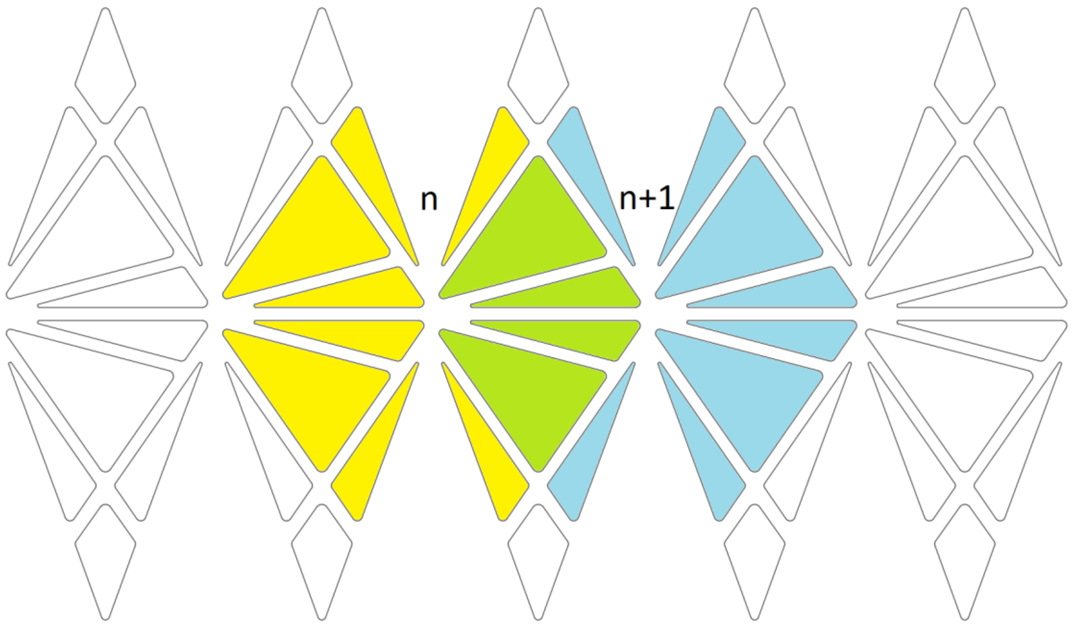
$(n)^m; m \mid m(mod)4 = \pm 2$	$(n-1)^p;\ p \in \{2\mathbb{Z}+1\}$	Error

Each permutation algorithm is required to begin and end in Standard Orientation. When starting in Standard Orientation, performing a move congruent to a 180-degree turn applied to axis "n" would leave axis "n-1" in the 3rd state of Figure 6. Applying a 90-degree turn to axis "n-1" would result in the 3rd state of Figure 7 and not preserve the Polar Orientation required to perform the X' portion of this {X,Y} conjugate.

Field 9

$(n)^m; m \mid m(mod)4 = \pm 2$	$(n+1)^p;\ p \mid p(mod)4 \neq 0$	Equation 7

Figure 30 Field 9 Venn Diagram



This conjugate results in an edge 4-cycle and a face 4-cycle. As the algorithm affects more than one piece-type (edges and faces), the algorithm is considered “impure”.

Equation 4 Impure Generic 4-Cycle

$$\mathrm{G4}_{\mathrm{n}}^{\mathrm{p}} = \{(\mathrm{n}-1)'', (\mathrm{n})^{\mathrm{p}}\}; \quad \mathrm{p}|\mathrm{p}(\mathrm{mod})4 \neq 0$$

While “n” may be any axis, it will arbitrarily be set to “3” to generate a specific example Movement Chart. “n” may be offset using Coordinate Frame Transformations.

Equation 5 Impure Generic 4-Cycles, reformatted

$$G4_n^p = \{(n-1)''\}\,(n)^p\,\{(n-1)''\}; \quad p|p(mod)4 \neq 0$$

Equation 6 Impure Generic 4-Cycle, Reformatted, n=3

$$G4_3^p = \{2''\}\,(3)^p\,\{2''\}; \quad p|p(mod)4 \neq 0$$

Which, for practical purposes, can be reduced to:

Equation 7 Impure Generic 4-Cycles, n=3, Reduced

$$G4_3^p = \{2''\}\,(3)^p\,\{2''\}; \quad p \in \{-1,1,2\}$$

Figure 31 Impure Generic 4-Cycles Movement Chart

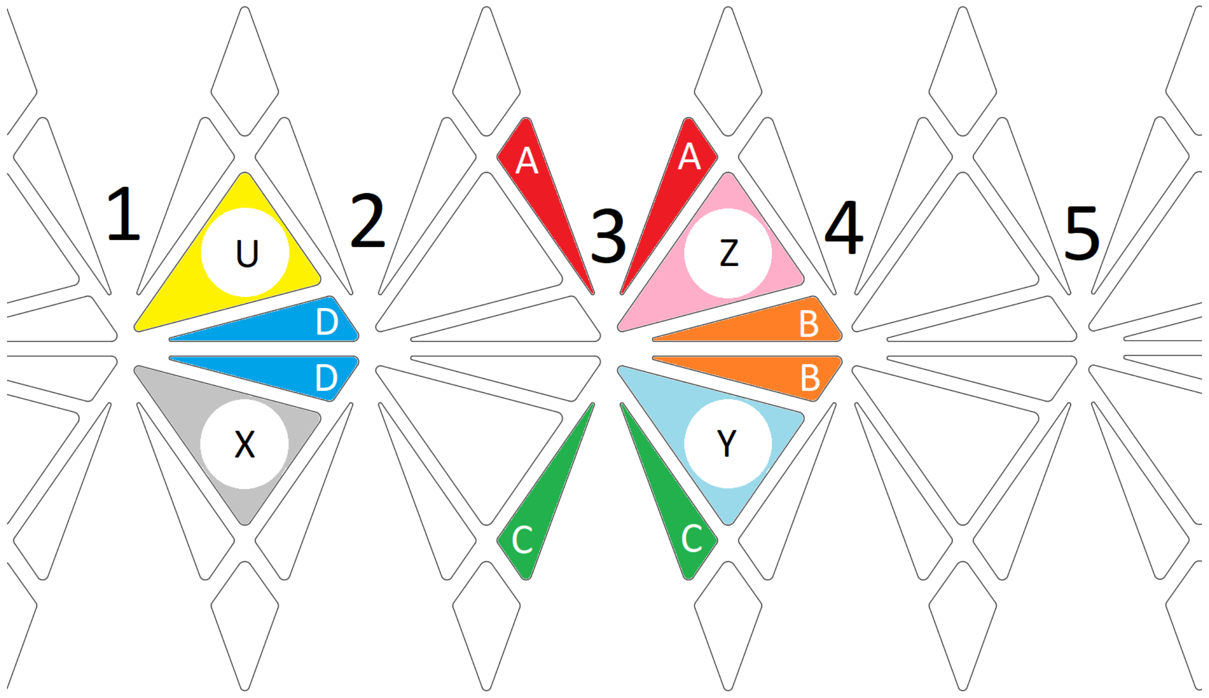


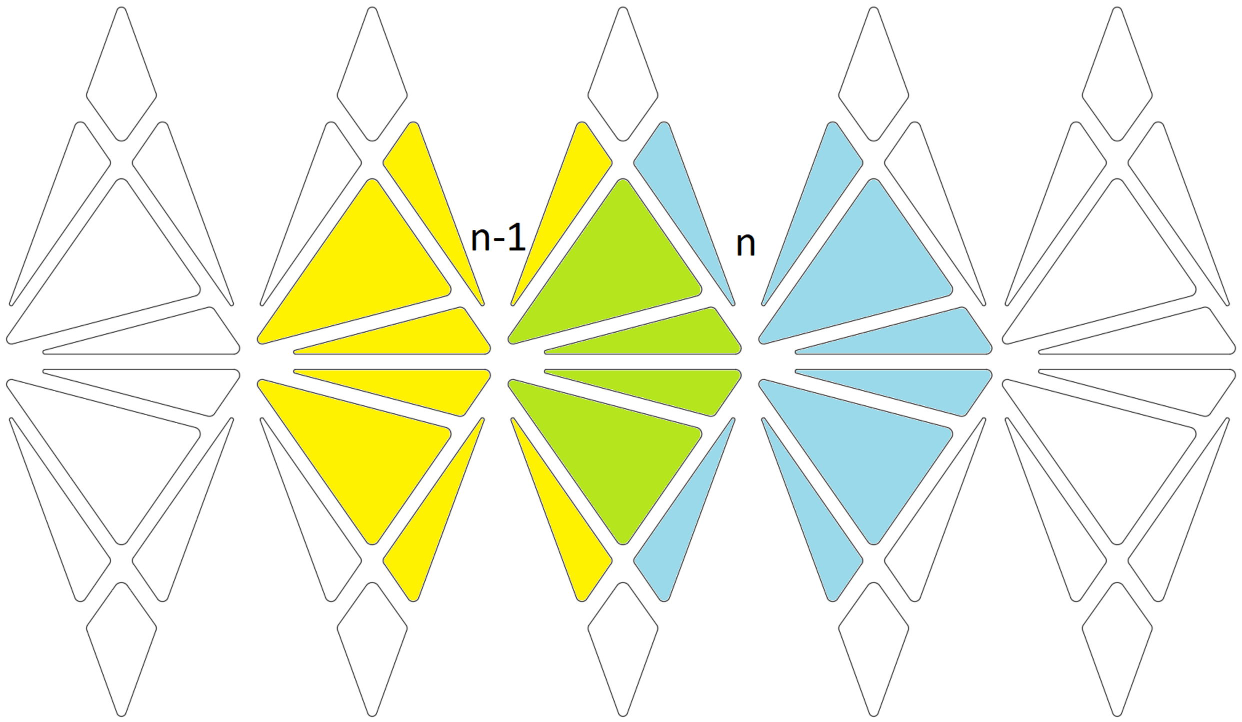
Table 6 Impure Specific 4-Cycles

N	Algorithm	Result
-1	{2″} 3′ {2″}	(A,D,C,B) (U,Y,Z,X)
1	{2″} 3{2″}	(A,B,C,D) (U,X,Z,Y)
2	{2″} 3″ {2″}	(A,C)(B,D) (U,Z)(X,Y)

Field 6

$(n)^m; m\|m(mod)4 = \pm 2$	$(n-1)^p;\ p \mid p(mod)4 = \pm 2$	Equation 8

Figure 32 Field 6 Venn Diagram



This conjugate results in a pair of edge 2-cycles and a pair of face 2-cycles. As the algorithm affects more than one piece-type, the algorithm is considered "impure". Additionally, the $G4_n^2$ variant of Field 9 will be included as it results in a substantially similar permutation of pieces.

Equation 8 Generic Double 2-Cycles

$$G2_m^n = \{n'', m''\}; \quad n = m \pm 1$$

While "n" may be any axis, "n" and "m" will disjointly be set to "3" and "4" to generate a specific example Movement Chart. "n" may be offset using Coordinate Frame Transformations.

Figure 33 Generic Double 2-Cycles Movement Chart

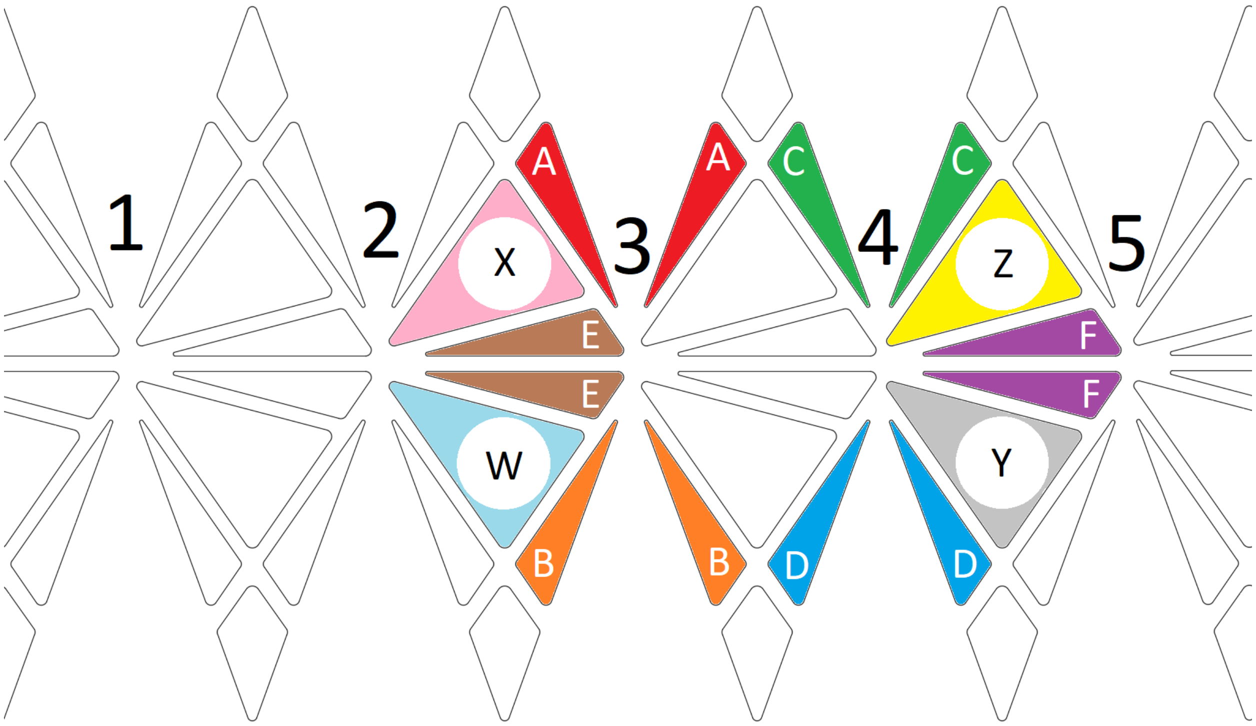


Table 7 Impure Specific Double 2-Cycles

N	M	Algorithm	Effect
3	4	$\{3''\}4''\{3''\}$	(E,F)(C,D) (W,Y)(X,Z)
4	3	$\{4''\}3''\{4''\}$	(E,F)(A,B) (W,Y)(X,Z)

Both variants of the impure specific double 2-cycles effect (E,F), (W,Y), and (X,Z). The choice of whether to swap the (C,D) or the (A,B) edge pairs will be situation dependent and at the discretion of the solver.

Other Uses for Conjugates

Conjugates are also frequently used to temporarily relocate pieces into "setup" positions. If algorithm "Y" *could* perform a desirable cycle but there are no Coordinate Frame Transformations that would result in "Y" acting on the desired pieces, a set of chained setup algorithms can be employed prior to performing "Y" to reposition the desired pieces to the locations which "Y" acts upon.

Simplest Commutator

Commutators take the form of $[X][Y][X^{-1}][Y^{-1}]$ or $[X,Y]$. The simplest possible commutator sets X and Y to a single axis turn. Let both X and Y be any of the 5 axes and allow both X and Y to be rotated to any multiple of 90-degrees. These cases are generically expressed by the 7 fields below:

	X	Y	Result of $[X][Y][X^{-1}][Y^{-1}]$
1	$(n)^m; m \in \mathbb{Z}$	$(n)^p;\ p \in \mathbb{Z}$	Null, See Field 1
2	$(n)^m; m \in \mathbb{Z}$	$(n \pm 2)^p;\ p \in \mathbb{Z}$	Null, See Field 2
3	$(n)^m; m \vert m(mod)4 = 0$	$(n \pm 1)^p;\ p \in \mathbb{Z}$	Null, See Field 3
4	$(n)^m; m \in \{2\mathbb{Z}+1\}$	$(n \pm 1)^p;\ p \mid p(mod)4 = 0$	Error, See Field 4
5	$(n)^m; m \in \{2\mathbb{Z}+1\}$	$(n \pm 1)^p;\ p \mid p(mod)4 \neq 0$	Error, See Field 5
6	$(n)^m; m \vert m(mod)4 = \pm 2$	$(n \pm 1)^p;\ p \in \{2\mathbb{Z}+1\}$	Error, See Field 7
7	$(n)^m; m \vert m(mod)4 = \pm 2$	$(n \pm 1)^p;\ p \mid p(mod)4 = \pm 2$	Error, See ≈ Field 5

There does not exist a commutator which uses only a single axis turn for X and Y. The above fields 5-7 had potential but were invalidated due to orientation problems. It turns out that the shortest commutator algorithm is a commutator nested within a conjugate where the conjugate handles the orientation issues.

Equation 9 Impure Generic Edge 3-Cycles, Face 5-Cycles

$$E3_q^{n,m} = \{(q-1)''(q+2)'', [(q+1)'', (q)^n]^m\}; \quad n,m \in \{2\mathbb{Z}+1\}$$

Equation 10 Impure Generic Edge 3-Cycles, Face 5-Cycles, Reformatted

$$E3_q^{n,m} = \{(q-1)''(q+2)''\}(\,[(q+1''][q^n][(q+1)''][q^{-n}]\,)^m\{(q+2)''(q-1)''\}; \quad n,m \in \{2\mathbb{Z}+1\}$$

While “q” may be any axis, it will arbitrarily be set to “3” to generate a specific example Movement Chart. “q” may be offset using Coordinate Frame Transformations.

Equation 11 Impure Generic Edge 3-Cycles, Face 5-Cycles, q=3

$$E3_3^{n,m} = \{2''5''\}(\,[4''][3^n][4''][3^{-n}]\,)^m\{5''2''\}; \quad n,m \in \{2\mathbb{Z}+1\}$$

Which, for useful purposes, can be reduced to:

Equation 12 Impure Generic Edge 3-Cycles, Face 5-Cycles, q=3, Reduced

$$E3_3^{n,m} = \{2''5''\}(\,[4''][3^n][4''][3^{-n}]\,)^m\{5''2''\}; \quad n,m = \{-1{,}1\}$$

An argument could be made for “useful values” of m to contain {-5,5} as these result in pure edge 3-Cycles, or {-6,-3,3,6} which result in pure face 5-cycles. I’ve excluded {-5,5} as the solution presented in this paper does not require a pure edge 3-cycle and more efficient edge 3-cycles exist. I’ve excluded {-6,-3,3,6} as there are face 3-cycles which are easier to utilize than 5-cycles and whose algorithms are shorter.

Below is the Venn Diagram of pieces exchanged between X (blue) and Y (yellow) of the commutator. The overlap (green) region provides a mechanism for a useful exchange of pieces between blue and yellow.

Figure 34 Shortest Commutator Venn Diagram

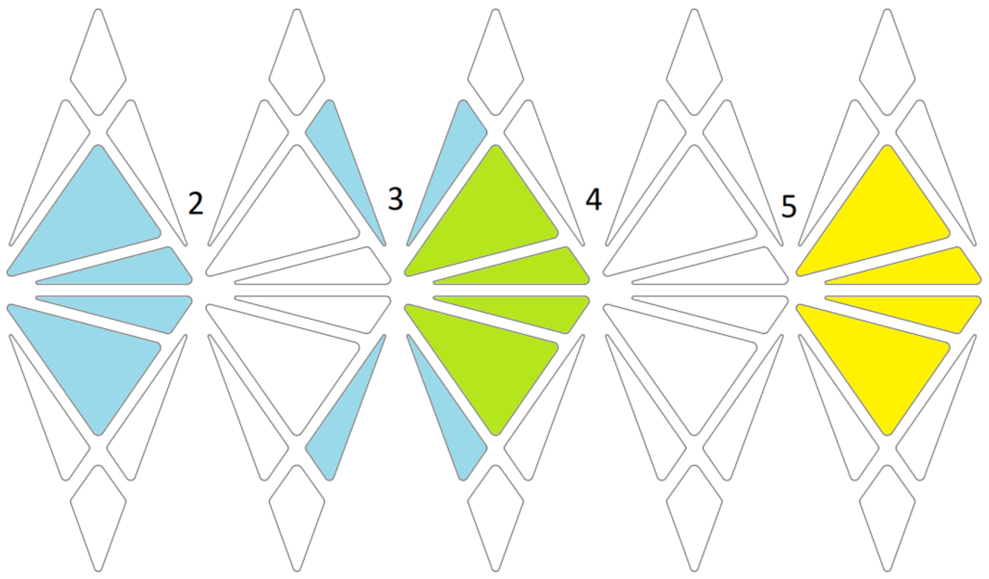


Figure 35 Impure Generic Edge 3-Cycle, Face 5-Cycles Movement Chart, q=3

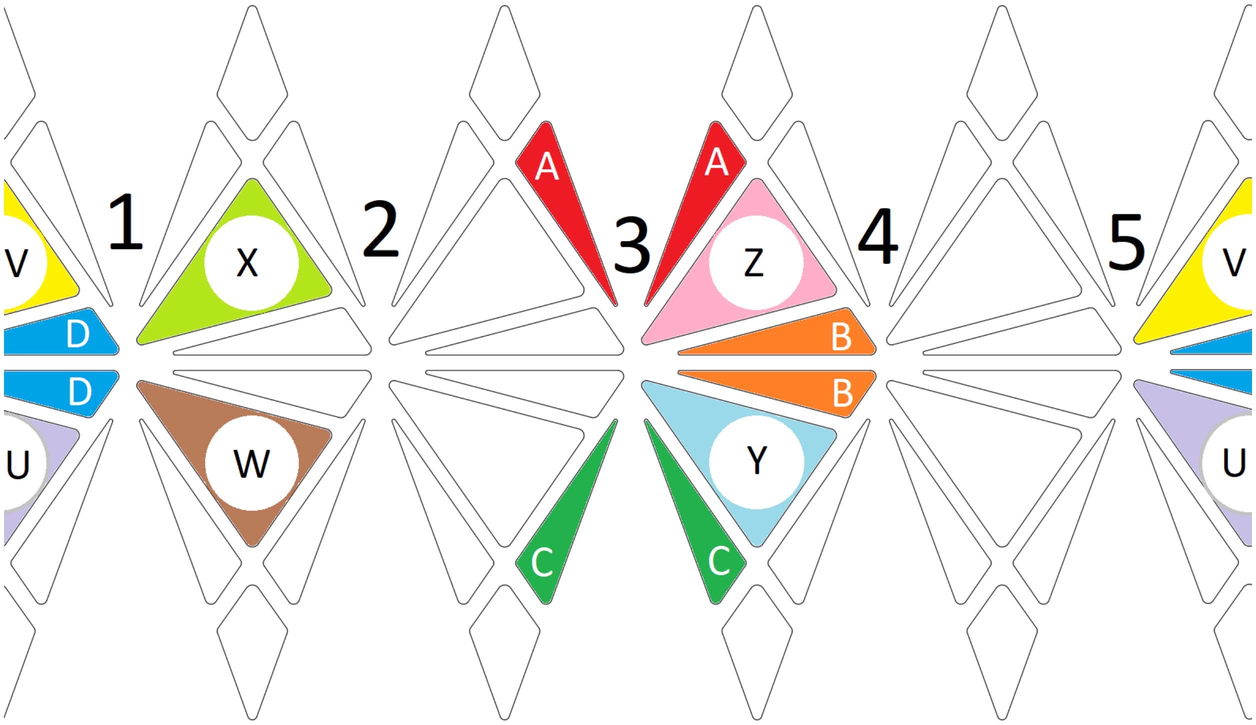


Table 8 Specific Edge 3-Cycles, q=3

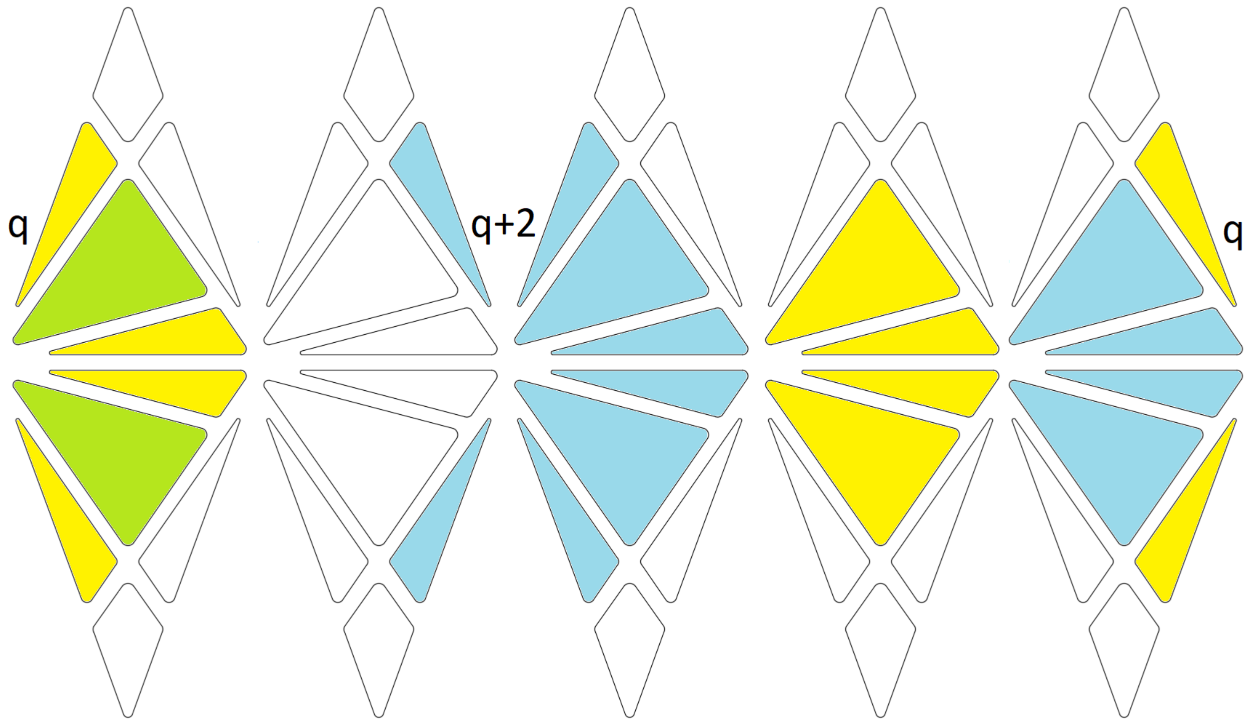
N	M	Algorithm	Results
-1	-1	$\{2''5''\}[3][4''][3'][4'']\{5''2''\}$	(A,B,D) (U,V,W,Z,Y)
-1	1	$\{2''5''\}[4''][3][4''][3']\{5''2''\}$	(A,D,B) (U,Y,Z,W,V)
1	-1	$\{2''5''\}[3'][4''][3][4'']\{5''2''\}$	(B,D,C) (U,X,Y,Z,V)
1	1	$\{2''5''\}[4''][3'][4''][3]\{5''2''\}$	(B,C,D) (U,V,Z,Y,X)

Constructing Pure Cycles

All three generic algorithms derived above are impure; they all cycle both edges and faces. While it is technically possible to solve the puzzle with only the presented impure algorithms, it would be exceptionally challenging to simultaneously permute both piece-types. Ideally, a set of pure cycles should be derived for at least one of the piece types. Due to the way the above algorithms were derived, they are necessarily the shortest possible conjugate and commutator algorithms of each of their respective cycle sizes. This means a pure cycle will have an equal or longer algorithm. As there are 10 faces and 15 edges, it will likely be faster to solve the edges with impure algorithms and the faces with pure algorithms compared to the other way around. Correct Parity is established when at least either all of the edges or all of the faces are correctly permuted. As such, only a pure 3-cycle or pure double 2-cycle will be required to permute the faces after the edges are permuted.

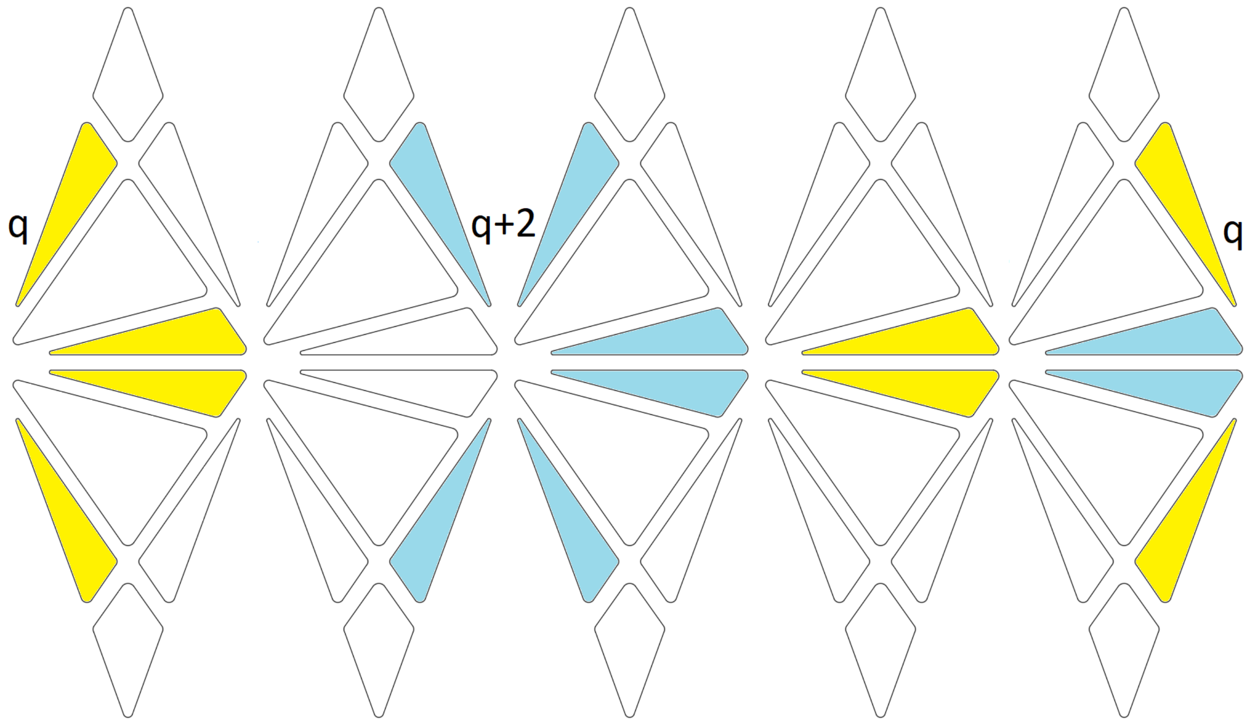
A set of generic pure face cycles can be constructed by sequencing $E3^{n,m}$, $G4^{p}$, and Coordinate Frame Transformations. If $G4^{p}$ is centered about q, and $E3^{n,m}$ is centered about $q+2$, then the algorithm's Venn diagram is as follows:

Figure 36 Generic Pure Face 3-Cycle Venn Diagram



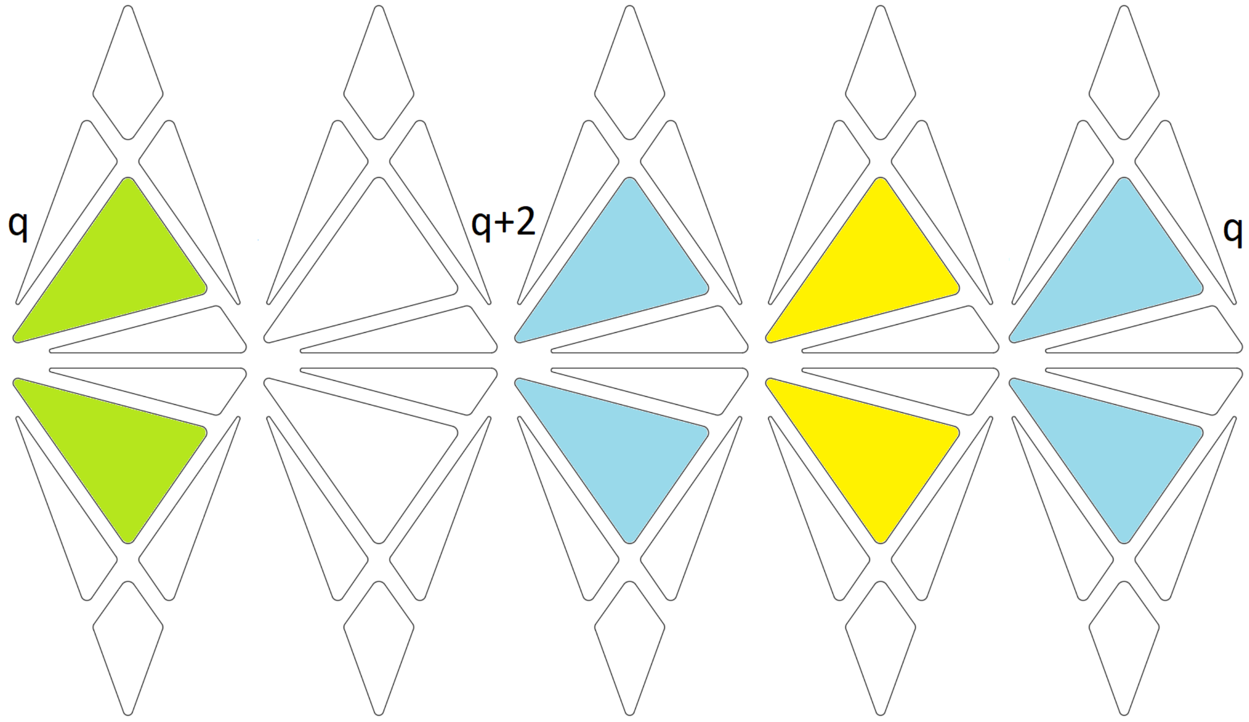
While both $E3_{q+2}^{n,m}$ and $G4_q^p$ are impure, there is no edge piece which is acted upon by both algorithms.

Figure 37 Generic Pure Face 3-Cycle Venn Diagram, Edges



However, there are faces which are acted upon by both algorithms.

Figure 38 Generic Pure Face 3-Cycle Venn Diagram, Faces



Therefore, there could exist a commutator constructed from $E3_{q+2}^{n,m}$ and $G4_{q}^{p}$ which results in a pure face #-cycle. Allowing n and m to be ±1, p to be {-1, 1, 2}, X of the commutator to be either $G4_{q}^{p}$, or $E3_{q+2}^{n,m}$, results in 24 different combinations for a generic pure faces n-cycle. The "#" in "#-cycle" happens to be 3 for all 24 combinations and these are the simplest pure face 3-cycles.

Figure 39 Pure Faces 3-cycle Movement Diagram

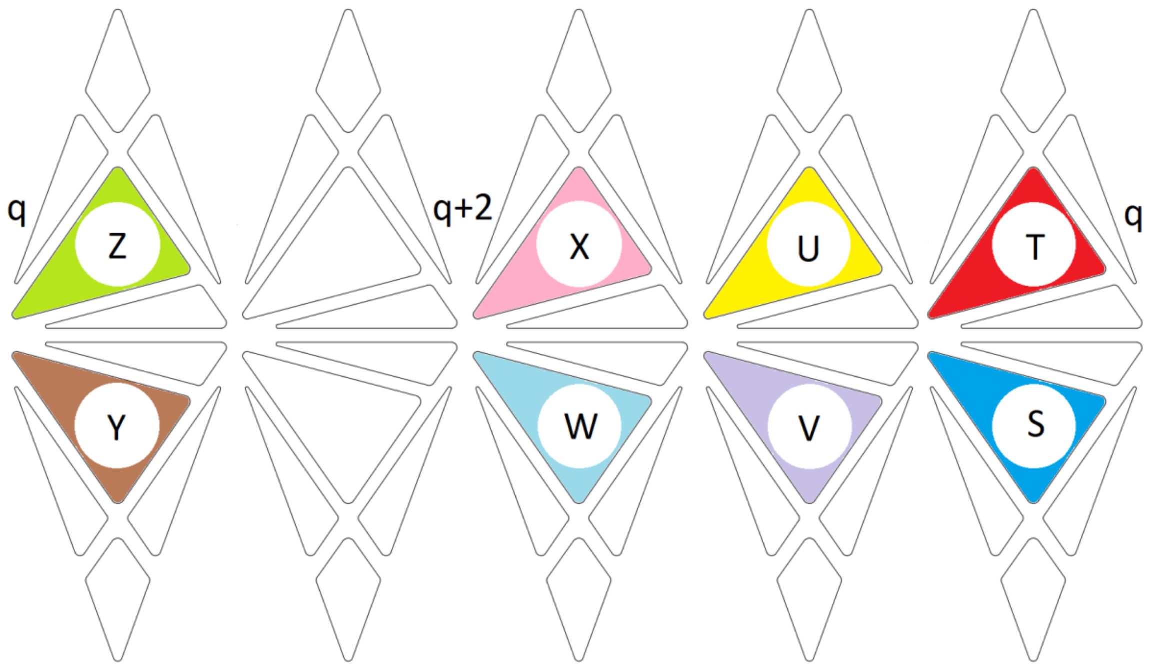


Table 9 Simplest Face 3-Cycles

	n	m	p	$G4^p$	Algorithm	Result
1	-1	-1	-1	X	$[G4_q^{-1}][E3_{q+2}^{-1,-1}][G4_q^1][E3_{q+2}^{-1,1}]$	(Y,T,U)
2	-1	-1	1	X	$[G4_q^1][E3_{q+2}^{-1,-1}][G4_q^{-1}][E3_{q+2}^{-1,1}]$	(Z,Y,T)
3	-1	-1	2	X	$[G4_q^2][E3_{q+2}^{-1,-1}][G4_q^2][E3_{q+2}^{-1,1}]$	(Y,T,V)
4	-1	1	-1	X	$[G4_q^{-1}][E3_{q+2}^{-1,1}][G4_q^1][E3_{q+2}^{-1,-1}]$	(Y,X,U)
5	-1	1	1	X	$[G4_q^1][E3_{q+2}^{-1,1}][G4_q^{-1}][E3_{q+2}^{-1,-1}]$	(Z,Y,X)
6	-1	1	2	X	$[G4_q^2][E3_{q+2}^{-1,1}][G4_q^2][E3_{q+2}^{-1,-1}]$	(Y,X,V)
7	1	-1	-1	X	$[G4_q^{-1}][E3_{q+2}^{1,-1}][G4_q^1][E3_{q+2}^{1,1}]$	(Z,S,Y)
8	1	-1	1	X	$[G4_q^1][E3_{q+2}^{1,-1}][G4_q^{-1}][E3_{q+2}^{1,1}]$	(Z,S,V)
9	1	-1	2	X	$[G4_q^2][E3_{q+2}^{1,-1}][G4_q^2][E3_{q+2}^{1,1}]$	(Z,S,U)
10	1	1	-1	X	$[G4_q^{-1}][E3_{q+2}^{1,1}][G4_q^1][E3_{q+2}^{1,-1}]$	(Z,W,Y)
11	1	1	1	X	$[G4_q^1][E3_{q+2}^{1,1}][G4_q^{-1}][E3_{q+2}^{1,-1}]$	(Z,W,V)
12	1	1	2	X	$[G4_q^2][E3_{q+2}^{1,1}][G4_q^2][E3_{q+2}^{1,-1}]$	(Z,W,U)
13	-1	-1	-1	Y	$[E3_{q+2}^{-1,-1}][G4_q^1][E3_{q+2}^{-1,1}][G4_q^{-1}]$	(Z,T,Y)
14	-1	-1	1	Y	$[E3_{q+2}^{-1,-1}][G4_q^{-1}][E3_{q+2}^{-1,1}][G4_q^1]$	(Y,U,T)
15	-1	-1	2	Y	$[E3_{q+2}^{-1,-1}][G4_q^2][E3_{q+2}^{-1,1}][G4_q^2]$	(Y,V,T)
16	-1	1	-1	Y	$[E3_{q+2}^{-1,1}][G4_q^1][E3_{q+2}^{-1,-1}][G4_q^{-1}]$	(Z,X,Y)
17	-1	1	1	Y	$[E3_{q+2}^{-1,1}][G4_q^{-1}][E3_{q+2}^{-1,-1}][G4_q^1]$	(Y,U,X)
18	-1	1	2	Y	$[E3_{q+2}^{-1,1}][G4_q^2][E3_{q+2}^{-1,-1}][G4_q^2]$	(Y,V,X)
19	1	-1	-1	Y	$[E3_{q+2}^{1,-1}][G4_q^1][E3_{q+2}^{1,1}][G4_q^{-1}]$	(Z,V,S)
20	1	-1	1	Y	$[E3_{q+2}^{1,-1}][G4_q^{-1}][E3_{q+2}^{1,1}][G4_q^1]$	(Z,Y,S)
21	1	-1	2	Y	$[E3_{q+2}^{1,-1}][G4_q^2][E3_{q+2}^{1,1}][G4_q^2]$	(Z,U,S)
22	1	1	-1	Y	$[E3_{q+2}^{1,1}][G4_q^1][E3_{q+2}^{1,-1}][G4_q^{-1}]$	(Z,V,W)
23	1	1	1	Y	$[E3_{q+2}^{1,1}][G4_q^{-1}][E3_{q+2}^{1,-1}][G4_q^1]$	(Z,Y,W)
24	1	1	2	Y	$[E3_{q+2}^{1,1}][G4_q^2][E3_{q+2}^{1,-1}][G4_q^2]$	(Z,U,W)

All 24 combinations nominally accomplish the same task: purely cycle faces. With that said, it is my opinion that the combinations at lines 2 and 7, along with their inverses at lines 13 and 20 respectively, are the easiest to apply as the 3 faces all share a single axis: the q axis.

Equation 13 Pure Face 3-cycle, Combination 2

$$F2_q = \left[G4_q^1\right]\left[E3_{q+2}^{-1,-1}\right]\left[G4_q^{-1}\right]\left[E3_{q+2}^{-1,1}\right]$$

Let q=3, then Combination 2 would be:

Equation 14 Pure Face 3-cycle, Combination 2, q=3

$$F2_3 = \left[G4_3^1\right]\left[E3_5^{-1,-1}\right]\left[G4_3^{-1}\right]\left[E3_5^{-1,1}\right]$$

Expanding to axis notation yields:

Equation 15 Pure Face 3-cycle, Combination 2, q=3, Expanded

$$F2_3 = \left[\left\{2''\right\} 3 \left\{2''\right\}\right]\left[\left\{4''2''\right\}[5][1''][5'][1'']\left\{2''4''\right\}\right]\left[\left\{2''\right\} 3' \left\{2''\right\}\right]\left[\left\{4''2''\right\}[1''][5][1''][5']\left\{2''4''\right\}\right]$$

Adjacent disjoint cycles (cycles that do not share any pieces) may be performed in any order. The set of 8 pieces that are cycled under the move 4” and the set of 8 pieces that are cycled under move 2” are respectively disjoint; there are no pieces acted upon by both axes 2 and 4. As such, they can be optionally swapped.

Equation 16 Pure Face 3-cycle, Combination 2, q=3, Expanded, Reordered

$$F2_3 = \left[\left\{2''\right\} 3 \left\{2''\right\}\right]\left[\left\{2''4''\right\}[5][1''][5'][1'']\left\{4''2''\right\}\right]\left[\left\{2''\right\} 3' \left\{2''\right\}\right]\left[\left\{2''4''\right\}[1''][5][1''][5']\left\{2''4''\right\}\right]$$

A simplification may be performed when the same axis is sequentially utilized.

Equation 17 Constant Axis Sequential Simplification

$$2'' + 2'' \equiv 2^0 = Null$$

The pieces cycled in the “1” axis are mutually exclusive from the pieces cycled in both the “3” and “4” axes. As such, a further simplification may be performed.

Equation 18 Pure Face 3-cycle, Combination 2 About Axis 3, Expanded, Reduced

$$F2_3 = \left[\left\{2''\right\} 3\right]\left[\left\{4''\right\}[5][1''][5'][1'']\left\{4''\right\}\right]\left[3'\right]\left[\left\{4''\right\}[1''][5][1''][5']\left\{2''4''\right\}\right]$$

Equation 19 Pure Face 3-cycle, Combination 2 About Axis 3, Clean

$$F2_3 = 2''34''51''5'4''3'4''51''5'2''4''$$

Figure 40 Select Pure Face 3-Cycles Movement Chart

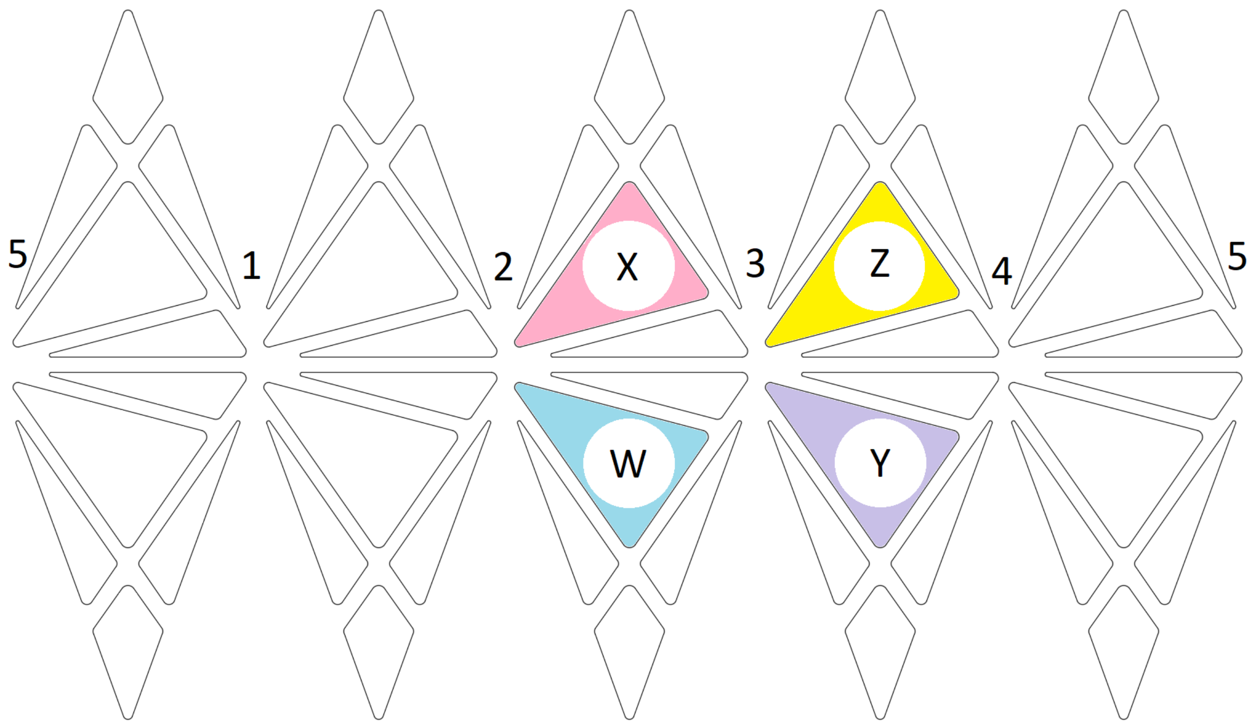


Table 10 Select Pure Face 3-Cycles

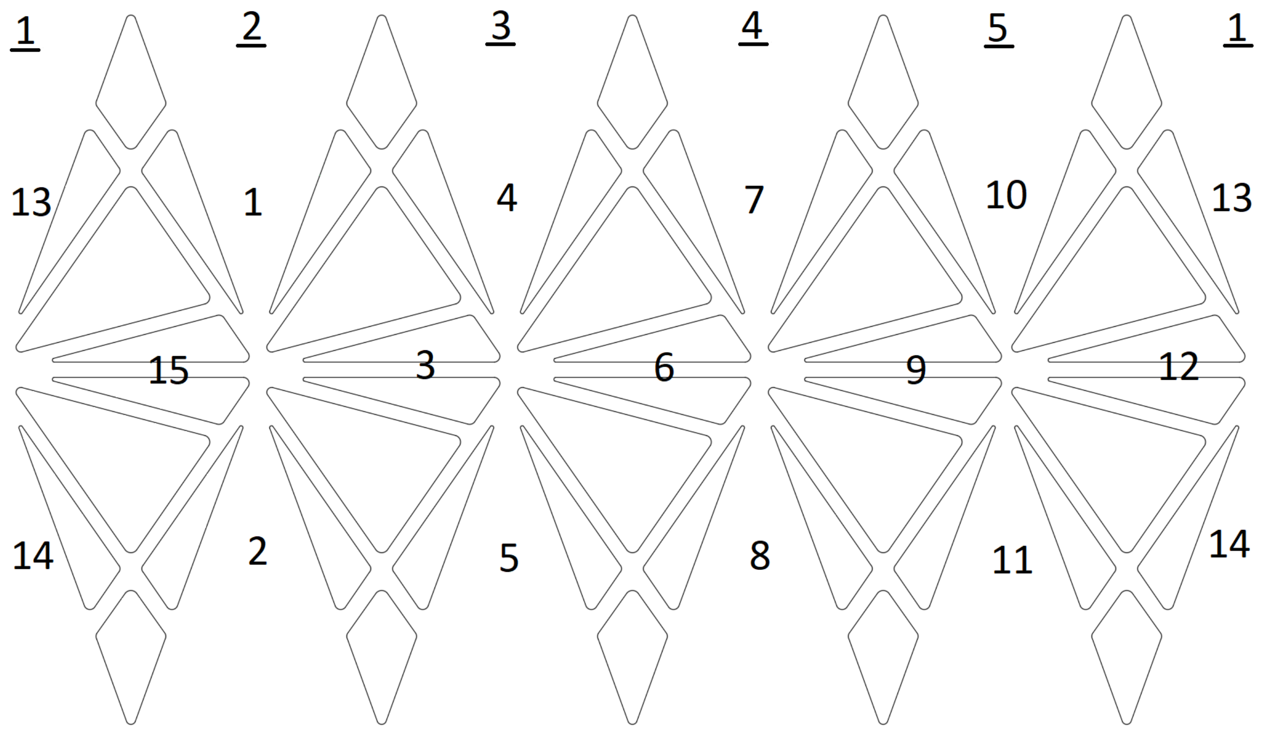
F2	2″34″51″5′4″3′4″51″5′2″4″	(X,Z,Y)
F2’	4″2″51″5′4″34″51″5′4″3′2″	(X,Y,Z)
F7	2″3′4″5′1″54″34″5′1″52″4″	(W,Y,Z)
F7’	4″2″5′1″54″3′4″5′1″54″32″	(W,Z,Y)

Permuting the Edges

A possible procedure for permuting the edges is to alternate solving polar pairs followed by the adjacent non-divergent edge (see Figure 17) in a systematic Easterly fashion. This procedure is very unlikely to be the most efficient method for any given scramble, but it is algorithmic. To trim down an already large set of possibilities, I will further constrain this edge permuting method to require the Northern polar edge to be solved before the Southern polar edge. This is to say that the edges will be solved sequentially with the below numbering. Intuition and heuristics may allow for a more efficient permutation of the edges.

Permuting the First Eleven Edges

Figure 41 Location and Axes Labeling for Edge Permutation

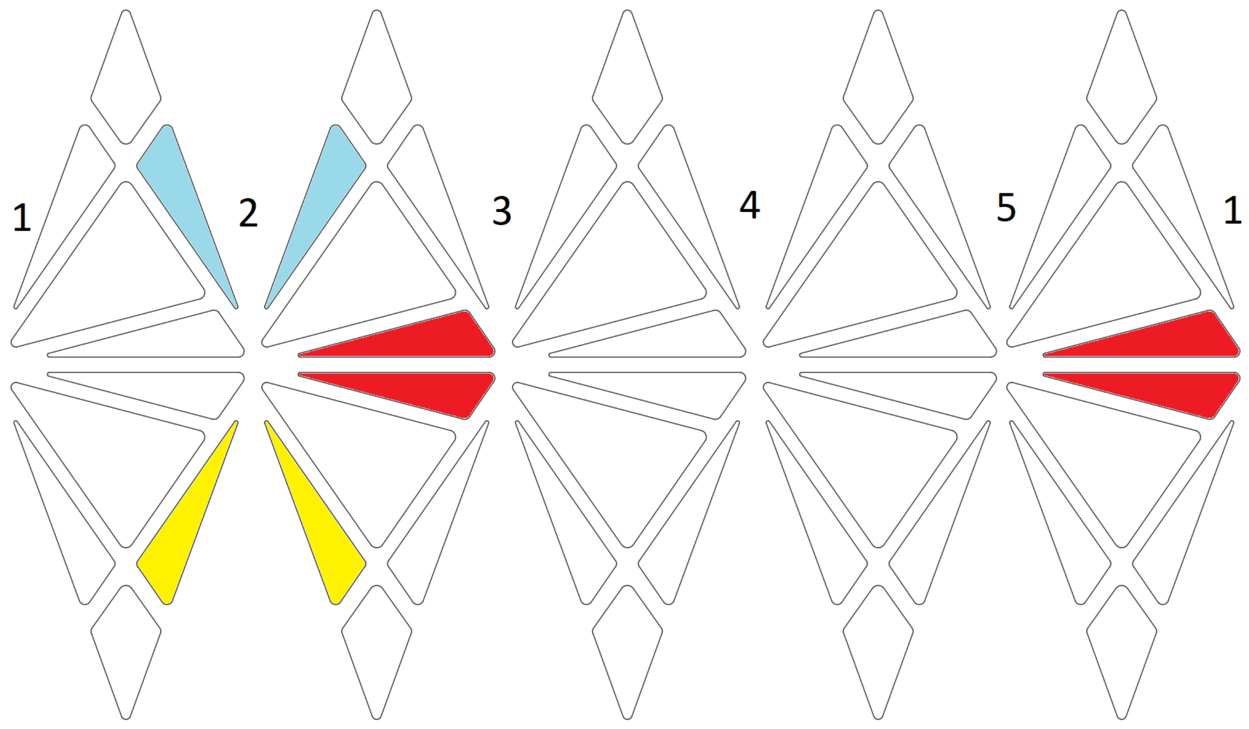


The underlined numbers represent axes. The non-underlined numbers uniquely identify edge locations, called "holes". The non-underlined "1" represents hole number 1. Let the contents of hole number 2 be the edge piece that should be in hole 1 when the puzzle is solved. The following line of Table 11 will permute the target edge piece from hole 2 to hole 1:

Hole	Piece	Algorithm	Reduced	# of Algs	# of Moves
1	2	$G4_2^2$	$1''2''1''$	1	3

This line from Table 11 could be interpreted as follows: To permute the contents of hole number 2 ("Piece") to the location of hole number 1 ("Hole"), perform the $G4_2^2$ composite algorithm ("Algorithm") which is equivalent to $1''2''1''$ ("Reduced") and consists of 1 constituent algorithm ("# of Algs") which requires a total of 3 axis turns when simplified ("# of Moves").

Permuting the First Edge, Location 2



The blue location is the target hole (hole 1). The yellow edge is the piece which occupies hole 1 in the solved position. The yellow edge is currently in hole 2. The red edges are the ancillary contents which will get relocated under the prescribed algorithm. The green edges are edges which have been previously permuted to their respective holes (yellow relocated to blue becomes green) and should maintain their permutation at the end of the prescribed algorithm. There are no green edges in this example. See Appendix 1: Edge Permutations for more information.

Table 11 Algorithms to Permute the First 11 Edges

Hole	Piece	Algorithm	Reduced	# of Algs	# of Moves
1	2	$G4_2^2$	$1''2''1''$	1	3
1	3	$G4_2^{-1}$	$1''2'1''$	1	3
1	4	$G4_3^1G2_4^5G4_2^1$	$2''32''5''4''5''1''21''$	3	9
1	5	$G4_3^{-1}G2_4^5G4_2^1$	$2''3'2''5''4''5''1''21''$	3	9
1	6	$G2_4^5G4_2^1$	$5''4''5''1''21''$	2	6
1	7	$G4_4^{-1}G4_2^{-1}$	$3''4'3''1''2'1''$	2	6
1	8	$G4_4^1G4_2^{-1}$	$3''43''1''2'1''$	2	6
1	9	$E3_2^{-1,-1}$	$1''4''23''2'3''4''1''$	1	8
1	10	$G4_5^1G4_2^1$	$4''54''1''21''$	2	6

1	11	$G4_5^{-1}G4_2^1$	$4''5'4''1''21''$	2	6
1	12	$G4_2^1$	$1''21''$	1	3
1	13	$G4_1^{-1}E3_2^{-1,-1}$	$5''1'5''1''4''23''2'3''4''1''$	2	11
1	14	$G4_1^1E3_2^{-1,-1}$	$5''15''1''4''23''2'3''4''1''$	2	11
1	15	$G4_1^2E3_2^{-1,-1}$	$5''1''5''1''4''23''2'3''4''1''$	2	11
2	3	$E3_2^{1,1}$	$1''4''3''23''2'4''1''$	1	8
2	4	$G4_3^{-1}G2_5^1E3_2^{1,-1}$	$2''3'2''1''5''4''2'3''23''4''1''$	3	12
2	5	$G4_3^1G2_5^1E3_2^{1,-1}$	$2''32''1''5''4''2'3''23''4''1''$	3	12
2	6	$G4_2^1G2_5^4G4_2^{-1}$	$1''21''4''5''4''1''2'1''$	3	9
2	7	$G4_4^1E3_2^{1,-1}$	$3''43''1''4''2'3''23''4''1''$	2	11
2	8	$G4_4^{-1}E3_2^{1,-1}$	$3''4'3''1''4''2'3''23''4''1''$	2	11
2	9	$E3_2^{1,-1}$	$1''4''2'3''23''4''1''$	1	8
2	10	$G4_2^1G4_5^1G4_2^{-1}$	$1''21''4''54''1''2'1''$	3	9
2	11	$G4_2^1G4_5^{-1}G4_2^{-1}$	$1''21''4''5'4''1''2'1''$	3	9
2	12	$E3_2^{-1,-1}G4_2^{-1}$	$1''4''23''2'3''4''2'1''$	2	9
2	13	$G4_1^{-1}E3_2^{1,-1}$	$5''1'5''1''4''2'3''23''4''1''$	2	11
2	14	$G4_1^1E3_2^{1,-1}$	$5''15''1''4''2'3''23''4''1''$	2	11
2	15	$G4_1^2E3_2^{1,-1}$	$5''1''5''1''4''2'3''23''4''1''$	2	11
3	4	$G4_3^1G2_4^5G2_1^2$	$2''32''5''4''5''2''1''2''$	3	9
3	5	$G4_3^{-1}G2_4^5G2_1^2$	$2''3'2''5''4''5''2''1''2''$	3	9
3	6	$G2_4^5G2_1^2$	$5''4''5''2''1''2''$	2	6
3	7	$G4_4^{-1}$	$3''4'3''$	1	3
3	8	$G4_4^1$	$3''43''$	1	3
3	9	$G4_4^2$	$3''4''3''$	2	3
3	10	$G4_5^1G2_1^2$	$4''54''2''1''2''$	2	6
3	11	$G4_5^{-1}G2_1^2$	$4''5'4''2''1''2''$	1	6
3	12	$G2_1^2$	$2''1''2''$	2	3
3	13	$G4_1^{-1}G2_3^4$	$5''1'5''4''3''4''$	2	6

3	14	$G4_1^1G2_3^4$	$5''15''4''3''4''$	2	6
3	15	$G4_1^2G2_3^4$	$5''1''5''4''3''4''$	1	6
4	5	$G4_3^2$	$2''3''2''$	1	3
4	6	$G4_3^{-1}$	$2''3'2''$	1	3
4	7	$E3_4^{-1,1}G4_3^1$	$3''1''5''45''4'1''3''2''32''$	2	11
4	8	$E3_4^{1,1}G4_3^1$	$3''1''5''4'5''41''3''2''32''$	2	11
4	9	$G2_1^5G4_3^1$	$5''1''5''2''32''$	2	6
4	10	$G4_5^1E3_3^{-1,-1}$	$4''54''2''5''34''3'4''5''2''$	2	11
4	11	$G4_5^{-1}E3_3^{-1,-1}$	$4''5'4''2''5''34''3'4''5''2''$	2	11
4	12	$E3_3^{-1,-1}$	$2''5''34''3'4''5''2''$	1	8
4	13	$G4_1^1G4_3^1$	$5''15''2''32''$	2	6
4	14	$G4_1^{-1}G4_3^1$	$5''1'5''2''32''$	2	6
4	15	$G4_3^1$	$2''32''$	1	3
5	6	$E3_3^{1,1}$	$2''5''4''3'4''35''2''$	1	8
5	7	$G4_3^1E3_4^{-1,1}G4_3^{-1}$	$2''32''3''1''5''45''4'1''3''2''3'2''$	3	14
5	8	$G4_3^1E3_4^{1,1}G4_3^{-1}$	$2''32''3''1''5''4'5''41''3''2''3'2''$	3	14
5	9	$G4_3^1E3_4^{1,-1}G4_3^{-1}$	$2''32''3''1''4'5''45''1''3''2''3'2''$	3	14
5	10	$G4_5^1E3_3^{1,-1}$	$4''54''2''5''3'4''34''5''2''$	2	11
5	11	$G4_5^{-1}E3_3^{1,-1}$	$4''5'4''2''5''3'4''34''5''2''$	2	11
5	12	$E3_3^{1,-1}$	$2''5''3'4''34''5''2''$	1	8
5	13	$G4_3^1G4_1^1G4_3^{-1}$	$2''32''5''15''2''3'2''$	3	9
5	14	$G4_3^1G4_1^{-1}G4_3^{-1}$	$2''32''5''1'5''2''3'2''$	3	9
5	15	$G4_1^{-1}G4_3^1G4_1^1G4_3^{-1}$	$5''1'5''2''32''5''15''2''3'2''$	4	12
6	7	$G2_2^3E3_4^{-1,1}G2_2^3$	$3''2''1''5''45''4'1''2''3''$	3	10
6	8	$G2_2^3E3_4^{1,1}G2_2^3$	$3''2''1''5''4'5''41''2''3''$	3	10
6	9	$G2_2^3E3_4^{-1,-1}G2_2^3$	$3''2''1''45''4'5''1''2''3''$	3	10
6	10	$G4_5^{-1}$	$4''5'4''$	1	3
6	11	$G4_5^1$	$4''54''$	1	3

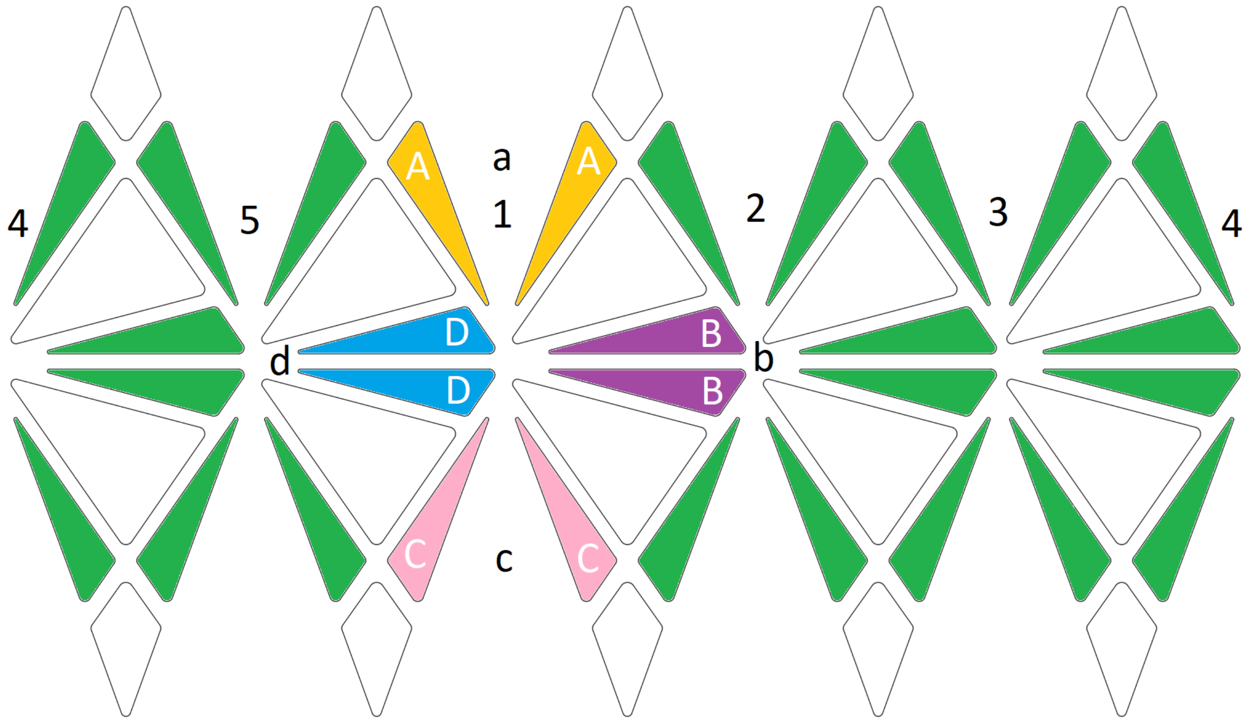
6	12	$G4_5^2$	$4''5''4''$	1	3
6	13	$E3_1^{-1,1}$	$5''3''2''12''1'3''5''$	1	8
6	14	$E3_1^{1,1}$	$5''3''2''1'2''13''5''$	1	8
6	15	$E3_1^{1,1}$	$5''3''1'2''12''3''5''$	1	8
7	8	$G2_4^5G2_5^4$	$5''4''5''4''5''4''$	2	6
7	9	$E3_4^{-1,1}$	$3''1''5''45''4'1''3''$	1	8
7	10	$G4_4^{-1}E3_5^{-1,1}G4_4^1$	$3''4'3''4''2''1''51''5'2''4''3''43''$	3	14
7	11	$G4_4^{-1}E3_5^{1,1}G4_4^1$	$3''4'3''4''2''1''5'1''52''4''3''43''$	3	14
7	12	$G4_4^{-1}G2_1^2G4_4^1$	$3''4'3''2''1''2''3''43''$	3	9
7	13	$G4_4^1G4_1^{-1}G4_4^{-1}$	$3''43''5''1'5''3''4'3''$	3	9
7	14	$G4_4^1G4_1^1G4_4^{-1}$	$3''43''5''15''3''4'3''$	3	9
7	15	$E3_4^{-1,-1}$	$3''1''45''4'5''1''3''$	1	8
8	9	$E3_4^{1,1}$	$3''1''5''4'5''41''3''$	1	8
8	10	$G4_4^1E3_5^{-1,1}G4_4^{-1}$	$3''43''4''2''1''51''5'2''4''3''4'3''$	3	14
8	11	$G4_4^1E3_5^{1,1}G4_4^{-1}$	$3''43''4''2''1''5'1''52''4''3''4'3''$	3	14
8	12	$G4_4^1E3_5^{-1,-1}G4_4^{-1}$	$3''43''4''2''51''5'1''2''4''3''4'3''$	3	14
8	13	$G4_1^{-1}E3_4^{1,1}$	$5''1'5''3''1''5''4'5''41''3''$	2	11
8	14	$G4_1^1E3_4^{1,1}$	$5''15''3''1''5''4'5''41''3''$	2	11
8	15	$E3_4^{1,-1}$	$3''1''4'5''45''1''3''$	1	8
9	10	$G2_3^4E3_5^{-1,1}G2_3^4$	$4''3''2''1''51''5'2''3''4''$	3	10
9	11	$G2_3^4E3_5^{1,1}G2_3^4$	$4''3''2''1''5'1''52''3''4''$	3	10
9	12	$G2_3^4E3_5^{-1,-1}G2_3^4$	$4''3''2''51''5'1''2''3''4''$	3	10
9	13	$G4_1^{-1}$	$5''1'5''$	1	3
9	14	$G4_1^1$	$5''15''$	1	3
9	15	$G4_1^2$	$5''1''5''$	1	3
10	11	$G2_1^5G2_5^1$	$5''1''5''1''5''1''$	2	6
10	12	$G4_3^2G4_5^{-1}G4_3^2$	$2''3''2''4''5'4''2''3''2''$	3	9
10	13	$G4_5^{-1}E3_1^{-1,1}G4_5^1$	$4''5'4''5''3''2''12''1'3''5''4''54''$	3	14

10	14	$G4_5^{-1}E3_1^{1,1}G4_5^1$	$4''5'4''5''3''2''1'2''13''5''4''54''$	3	14
10	15	$G4_5^{-1}E3_1^{1,-1}G4_5^1$	$4''5'4''5''3''1'2''12''3''5''4''54''$	3	14
11	12	$G2_1^5G2_3^4E3_5^{1,1}G2_3^4G2_1^5$	$5''1''5''4''3''2''1''5'1''52''3''4''5''1''5''$	5	16
11	13	$G4_5^1E3_1^{-1,1}G4_5^{-1}$	$4''54''5''3''2''12''1'3''5''4''5'4''$	3	14
11	14	$G4_5^1E3_1^{1,1}G4_5^{-1}$	$4''54''5''3''2''1'2''13''5''4''5'4''$	3	14
11	15	$G4_5^1E3_1^{1,-1}G4_5^{-1}$	$4''54''5''3''1'2''12''3''5''4''5'4''$	3	14

Permuting the Last Four Edges

Assume a goal state of the following

Figure 42 Goal State of the Last Four Edges



Let the locations (holes) be denoted with lowercase letters (a, b, c, and d), and edge pieces be denoted with uppercase letters (A, B, C, and D).

Table 12 Possible Permutations of the Last Four Edges

	a	b	c	d	Composite Algorithms	Number of Algorithms
1	A	B	D	C	$(\{G2_2^1G2_3^4\}G4_1^{-1}\{G2_3^4G2_2^1\})(\{G2_4^5\}E3_1^{-1-,1}\{G2_4^5\})$	8
2	A	C	B	D	$(\{G2_2^1G2_3^4\}G4_1^1\{G2_3^4G2_2^1\})(\{G2_4^5\}E3_1^{-1,1}\{G2_4^5\})$	8
3	A	C	D	B	$\{G2_4^5\}E3_1^{1,1}\{G2_4^5\}$	3
4	A	D	B	C	$\{G2_4^5\}E3_1^{1,-1}\{G2_4^5\}$	3
5	A	D	C	B	$(\{G2_4^5\}E3_1^{1,1}\{G2_4^5\})(\{G2_2^1G2_3^4\}G4_1^{-1}\{G2_3^4G2_2^1\})(\{G2_4^5\}E3_1^{-1-,1}\{G2_4^5\})$	11

6	B	A	C	D	$(\{G2_2^1G2_3^4\}G4_1^{-1}\{G2_3^4G2_2^1\})(\{G2_4^5\}E3_1^{1,1}\{G2_4^5\})$	8
7	B	A	D	C	$(\{G2_4^5\}E3_1^{1,-1}E3_1^{-1,-1}\{G2_4^5\})$	4
8	B	C	A	D	$E3_1^{1,1}E3_1^{-1,-1}$	2
9	B	C	D	A	$\{G2_2^1G2_3^4\}G4_1^1\{G2_3^4G2_2^1\}$	5
10	B	D	A	C	$(\{G2_2^1G2_3^4\}G4_1^{-1}\{G2_3^4G2_2^1\})(\{G2_4^5\}E3_1^{-1,1}\{G2_4^5\})$	8
11	B	D	C	A	$\{G2_4^5\}E3_1^{-1-,1}\{G2_4^5\}$	3
12	C	A	B	D	$E3_1^{-1,1}E3_1^{1,-1}$	2
13	C	A	D	B	$\{G2_2^1G2_3^4\}G4_1^{-1}\{G2_3^4G2_2^1\}\{G2_4^5\}E3_1^{1-,1}\{G2_4^5\}$	8
14	C	B	A	D	$(\{G2_4^5\}E3_1^{1,1}\{G2_4^5\})(\{G2_2^1G2_3^4\}G4_1^1\{G2_3^4G2_2^1\})(\{G2_4^5\}E3_1^{1,1}\{G2_4^5\})$	11
15	C	B	D	A	$\{G2_4^5\}E3_1^{-1,-1}E3_1^{1,1}\{G2_4^5\}$	4
16	C	D	A	B	$\{G2_2^1G2_3^4\}G4_1^2\{G2_3^4G2_2^1\}$	5
17	C	D	B	A	$(\{G2_2^1G2_3^4\}G4_1^1\{G2_3^4G2_2^1\})(\{G2_4^5\}E3_1^{1,1}\{G2_4^5\})$	8
18	D	A	B	C	$\{G2_2^1G2_3^4\}G4_1^{-1}\{G2_3^4G2_2^1\}$	5
19	D	A	C	B	$\{G2_4^5\}E3_1^{-1,1}\{G2_4^5\}$	3
20	D	B	A	C	$\{G2_4^5\}E3_1^{1,-1}E3_1^{-1,1}\{G2_4^5\}$	4
21	D	B	C	A	$(\{G2_2^1G2_3^4\}G4_1^1\{G2_3^4G2_2^1\})(\{G2_4^5\}E3_1^{1,-1}\{G2_4^5\})$	8
22	D	C	A	B	$(\{G2_2^1G2_3^4\}G4_1^1\{G2_3^4G2_2^1\})(\{G2_4^5\}E3_1^{-1-,1}\{G_4^5\})$	8
23	D	C	B	A	$(\{G2_4^5\}E3_1^{1,1}E3_1^{-1,1}\{G2_4^5\})$	4

Table 13 Possible Permutations of the Last Four Edges, Reduced

	a	b	c	d	Reduced Composite Algorithms	Moves
1	A	B	D	C	1″2″1″4″3″4″5″1′5″4″3″4″1″2″1″5″4″3″12″1′2″3″4″5″	25
2	A	C	B	D	1″2″1″4″3″4″5″15″4″3″4″1″2″1″5″4″3″2″12″1′3″4″5″	25
3	A	C	D	B	5″4″3″2″1′2″13″4″5″	10
4	A	D	B	C	5″4″3″1′2″12″3″4″5″	10
5	A	D	C	B	5″4″3″2″1′2″13″4″5″1″2″1″4″3″4″5″1′5″4″3″4″1″2″1″5″4″3″12″1′2″3″4″5″	35
6	B	A	C	D	1″2″1″4″3″4″5″1′5″4″3″4″1″2″1″5″4″3″2″1′2″13″4″5″	25
7	B	A	D	C	5″4″3″1′2″12″12″1′2″3″4″5″	14
8	B	C	A	D	5″3″2″1′2″1″2″1′2″3″5″	11
9	B	C	D	A	1″2″1″4″3″4″5″15″4″3″4″1″2″1″	15
10	B	D	A	C	1″2″1″4″3″4″5″1′5″4″3″4″1″2″1″5″4″3″2″12″1′3″4″5″	24
11	B	D	C	A	5″4″3″12″1′2″3″4″5″	10
12	C	A	B	D	5″3″2″12″1″2″12″3″5″	11
13	C	A	D	B	1″2″1″4″3″4″5″1′5″4″3″4″1″2″1″5″4″3″1′2″12″3″4″5″	25
14	C	B	A	D	5″4″3″2″1′2″13″4″5″1″2″1″4″3″4″5″15″4″3″4″1″2″1″5″4″3″2″1′2″13″4″5″	35
15	C	B	D	A	5″4″3″12″1″2″13″4″5″	12
16	C	D	A	B	1″2″1″4″3″4″5″1″5″4″3″4″1″2″1″	15
17	C	D	B	A	1″2″1″4″3″4″5″15″4″3″4″1″2″1″5″4″3″2″1′2″13″4″5″	25
18	D	A	B	C	1″2″1″4″3″4″5″1′5″4″3″4″1″2″1″	15
19	D	A	C	B	5″4″3″2″12″1′3″4″5″	10
20	D	B	A	C	5″4″3″1′2″1″2″1′3″4″5″	11
21	D	B	C	A	1″2″1″4″3″4″5″15″4″3″4″1″2″1″5″4″3″1′2″12″3″4″5″	25
22	D	C	A	B	1″2″1″4″3″4″5″15″4″3″4″1″2″1″5″4″3″12″1′2″3″4″5″	25
23	D	C	B	A	5″4″3″2″1′2″12″12″1′3″4″5″	14

Permuting the Faces

The faces *could* be permuted using only setup moves and the four pure face 3-cycles previously derived:

Figure 40 Select Pure Face 3-Cycles Movement Chart

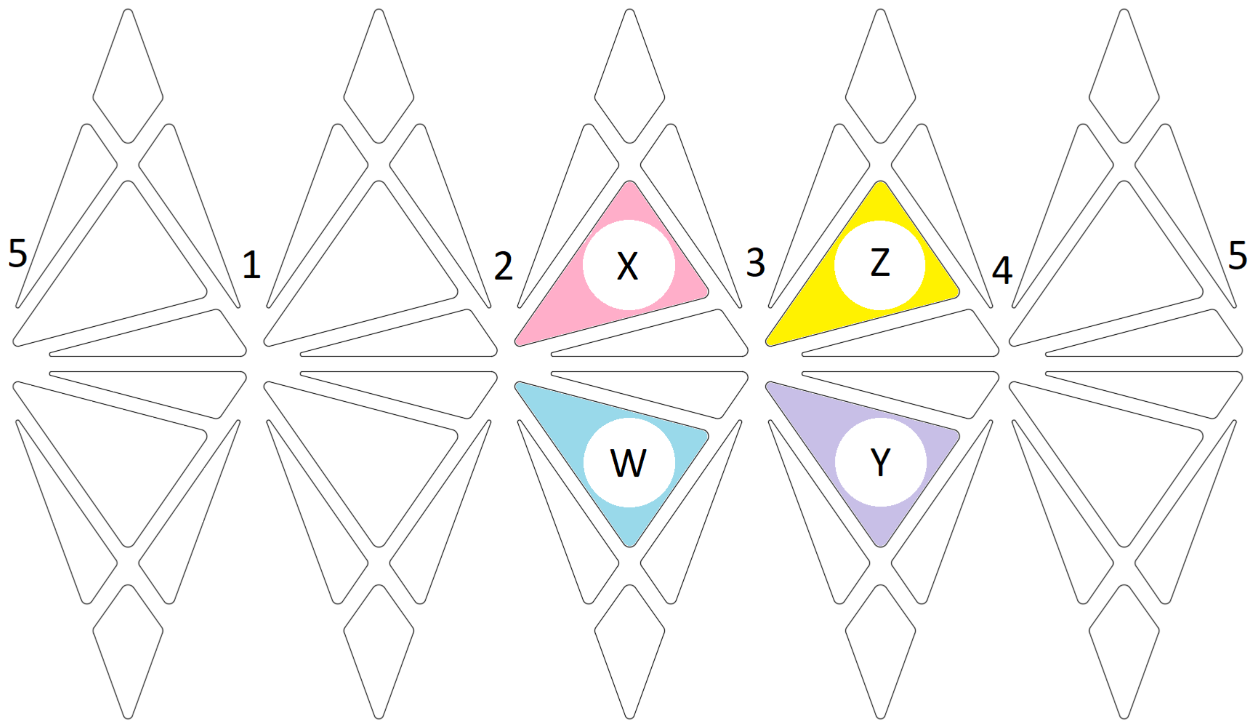


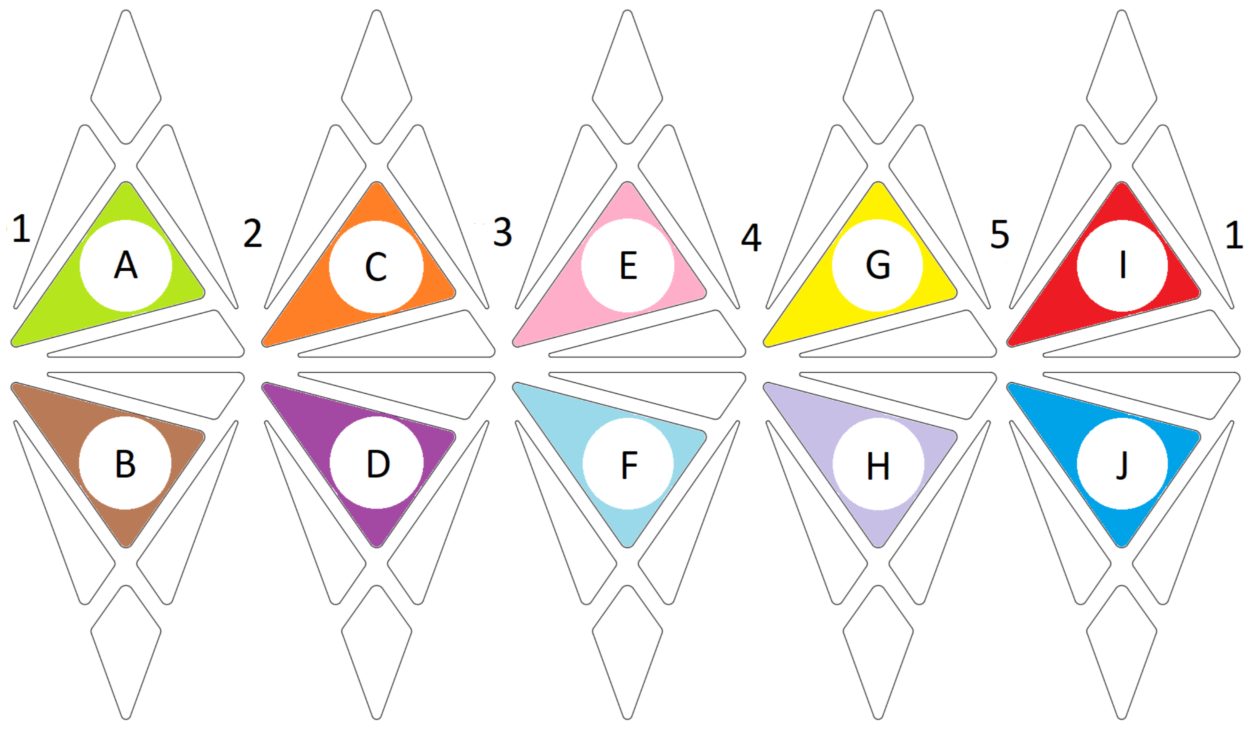
Table 10 Select Pure Face 3-Cycles

F2	2″34″51″5′4″3′4″51″5′2″4″	(X,Z,Y)
F2’	4″2″51″5′4″34″51″5′4″3′2″	(X,Y,Z)
F7	2″3′4″5′1″54″34″5′1″52″4″	(W,Y,Z)
F7’	4″2″5′1″54″3′4″5′1″54″32″	(W,Z,Y)

This set of four pure face 3-cycles always operates on Z and Y while selectively operating on either X or W. To permute with any of the 6 unlabeled face pieces using the algorithms above, the pure face 3-cycle algorithm must be nested within a conjugate. The conjugates will temporarily relocate the unlabeled desired face pair into the face locations between axis 2 and 3.

Selectively Permuting Any 3 Faces

Figure 43 Labeling of All Faces



There are 120 combinations of three faces when selecting from ten faces with no repeats ($_{10}C_3 = 120$, see Equation 1). Accounting for 5-fold rotational symmetry, the number is reduced to a fifth of the possible 120 combinations. For example, (A,C,E) is congruent to (C,E,G) by rigid body motion about the polar axis (which is the same as being congruent via Coordinate Frame Transformations).

Table 14 All Face 3-Cycles

#	3-Cycle			Composite Algorithm	Short $(F2_n, F7_n, F13_n, F20_n)$	Alternative	Short $(F1_n - F24_n)$
1	A	B	C	$\{G4_3^2, [G4_3^1, E3_5^{-1,-1}]\}$	$\{G4_3^2, F2_3\}$	-	-
2	A	B	D	$\{G2_1^2, [E3_3^{1,-1}, G4_1^{-1}]\}$	$\{G2_1^2, F20_1\}$	-	-
3	A	B	E	$\{G2_5^4, [G4_1^1, E3_3^{-1,-1}]\}$	$\{G2_5^4, F2_1\}$	$[G4_1^1, E3_3^{-1,1}]$	$F5_1$
4	A	B	F	$\{G2_4^5, [E3_3^{1,-1}, G4_1^{-1}]\}$	$\{G2_4^5, F20_1\}$	$[E3_3^{1,1}, G4_1^{-1}]$	$F23_1$
5	A	B	G	$\{(G2_3^2 G2_5^4), [G4_5^1, E3_2^{-1,-1}]\}$	$\{(G2_3^2 G2_5^4), F2_5\}$	$\{G4_3^{-1}, [E3_3^{1,1}, G4_1^2]\}$	$\{G4_3^{-1}, F24_1\}$
6	A	B	H	$\{(G2_3^2 G2_5^4), [E3_2^{1,-1}, G4_5^{-1}]\}$	$\{(G2_3^2 G2_5^4), F20_5\}$	$\{G4_4^1 G4_3^{-1}, [E3_5^{-1,-1}, G4_3^{-1}]\}$	$\{G4_4^1 G4_3^{-1}, F14_3\}$
7	A	B	I	$[G4_1^1, E3_3^{-1,-1}]$	$F2_1$	-	-
8	A	B	J	$[E3_3^{1,-1}, G4_1^{-1}]$	$F20_1$	-	-
9	A	C	E	$\{(G4_5^{-1} G4_2^1), [E3_4^{-1,-1}, G4_2^1]\}$	$\{(G4_5^{-1} G4_2^1), F13_2\}$	-	-
10	A	C	F	$\{(G2_5^4 G4_2^1), [E3_4^{-1,-1}, G4_2^1]\}$	$\{(G2_5^4 G4_2^1), F13_2\}$	$[E3_5^{-1,-1}, G4_3^{-1}]$	$F14_3$
11	A	C	G	$\{(G2_3^2 G4_1^{-1} G4_3^1), [G4_3^1, E3_5^{-1,-1}]\}$	$\{(G2_3^2 G4_1^{-1} G4_3^1), F2_3\}$	$\{G4_4^1, [G4_4^2, E3_1^{-1,1}]\}$	$\{G4_4^1, F6_4\}$
12	A	C	H	$\{G4_4^1, [E3_4^{-1,-1}, G4_2^1]\}$	$\{G4_4^1, F13_2\}$	$[G4_4^{-1}, E3_1^{-1,1}]$	$F4_2$
13	A	C	J	$\{G4_2^1, [E3_4^{-1,-1}, G4_2^1]\}$	$\{G4_2^1, F13_2\}$	-	-
14	A	D	E	$\{(G2_5^4 G4_2^{-1}), [E3_4^{-1,-1}, G4_2^1]\}$	$\{(G2_5^4 G4_2^{-1}), F13_2\}$	$[E3_5^{1,-1}, G4_3^2]$	$F21_3$
15	A	D	F	$\{G4_3^{-1}, [E3_5^{1,-1}, G4_3^{-1}]\}$	$\{G4_3^{-1}, F20_3\}$	-	-
16	A	D	G	$\{G4_4^{-1}, [G4_2^1, E3_4^{-1,-1}]\}$	$\{G4_4^{-1}, F2_2\}$	-	-

17	A	D	H	$\{(G4_{1}^{1}G2_{2}^{3}),[G4_{3}^{-1},E3_{5}^{1,-1}]\}$	$\{(G4_{1}^{1}G2_{2}^{3}),F7_{3}\}$	$[G4_{4}^{2},E3_{1}^{-1,1}]$	$F6_{4}$
18	A	D	J	$\{(G2_{5}^{4}G4_{3}^{-1}),[E3_{5}^{1,-1},G4_{3}^{-1}]\}$	$\{(G2_{5}^{4}G4_{3}^{-1}),F20_{3}\}$	$[E3_{4}^{-1,-1},G4_{2}^{2}]$	$F15_{2}$
19	A	E	J	$\{(G4_{5}^{-1}G4_{2}^{2}),[E3_{4}^{-1,-1},G4_{2}^{1}]\}$	$\{(G4_{5}^{-1}G4_{2}^{2}),F13_{2}\}$	$[G4_{3}^{2},E3_{5}^{1,1}]$	$F12_{3}$
20	A	F	H	$\{G4_{1}^{-1},[E3_{1}^{1,-1},G4_{4}^{-1}]\}$	$\{G4_{1}^{-1},F20_{4}\}$	$[G4_{1}^{1},E3_{3}^{1,1}]$	$F11_{1}$
21	A	F	J	$\{G4_{3}^{1},[G4_{1}^{-1},E3_{3}^{1,-1}]\}$	$\{G4_{3}^{1},F7_{1}\}$	-	-
22	A	H	J	$\{G4_{1}^{1},[G4_{1}^{-1},E3_{3}^{1,-1}]\}$	$\{G4_{1}^{1},F7_{1}\}$	$[E3_{3}^{1,-1},G4_{1}^{1}]$	$F19_{1}$
23	B	D	F	$\{(G4_{5}^{1}G4_{2}^{-1}),[E3_{4}^{1,-1},G4_{2}^{-1}]\}$	$\{(G4_{5}^{1}G4_{2}^{-1}),F20_{2}\}$	$\{G4_{1}^{1},[E3_{1}^{1,-1},G4_{4}^{1}]\}$	$\{G4_{1}^{1},F19_{4}\}$
24	B	D	H	$\{(G2_{3}^{2}G4_{1}^{-1}G4_{3}^{1}),[G4_{3}^{-1},E3_{5}^{1,-1}]\}$	$\{(G2_{3}^{2}G4_{1}^{-1}G4_{3}^{1}),F7_{3}\}$	$\{G4_{3}^{1},[G4_{4}^{2},E3_{1}^{-1,-1}]\}$	$\{G4_{3}^{1},F3_{4}\}$

Table 14 provides an example for each face 3-cycle using conjugates of $F2_n, F7_n, F13_n,$ *and* $F20_n$ as well as more efficient cycles using other face 3-cycles.

Table 15 All Face 3-Cycles, Efficient and with Inverses

#	3-Cycle			Composite Algorithm	Qty		Qty
1	A	B	C	$G4_{3}^{-1}E3_{5}^{-1,-1}G4_{3}^{-1}E3_{5}^{-1,1}G4_{3}^{2}$	5	2’’3’4’’51’’5’4’’3’4’’51’’5’4’’3’’2’’	15
	A	C	B			2’’3’’4’’51’’5’4’’34’’51’’5’4’’32’’	
2	A	B	D	$G2_{1}^{2}E3_{3}^{1,-1}G4_{1}^{-1}E3_{3}^{1,1}G4_{1}^{1}G2_{1}^{2}$	6	2’’1’’5’’3’4’’32’’1’2’’3’4’’32’’15’’2’’1’’2’’	18
	A	D	B			2’’1’’2’’5’’1’2’’3’4’’32’’12’’3’4’’35’’1’’2’’	
3	A	B	E	$G4_{1}^{1}E3_{3}^{-1,1}G4_{1}^{-1}E3_{3}^{-1,-1}$	4	5’’12’’4’’34’’3’2’’1’2’’34’’3’4’’2’’5’’	16
	A	E	B			5’’2’’4’’34’’3’2’’12’’34’’3’4’’2’’1’5’’	
4	A	B	F	$E3_{3}^{1,1}G4_{1}^{-1}E3_{3}^{1,-1}G4_{1}^{1}$	4	2’’5’’4’’3’4’’32’’1’2’’3’4’’34’’2’’15’’	16
	A	F	B			5’’1’2’’4’’3’4’’32’’12’’3’4’’34’’5’’2’’	
5	A	B	G	$G4_{3}^{-1}E3_{3}^{1,1}G4_{1}^{2}E3_{3}^{1,-1}G4_{1}^{2}G4_{3}^{1}$	6	2’’3’5’’4’’3’4’’32’’1’’2’’3’4’’34’’2’’1’’5’’2’’12’’	20
	A	G	B			2’’1’2’’5’’1’’2’’4’’3’4’’32’’1’’2’’3’4’’34’’5’’32’’	
6	A	B	H	$G4_{4}^{1}G4_{3}^{-1}E3_{5}^{-1,-1}G4_{3}^{-1}E3_{5}^{-1,1}G4_{3}^{2}G4_{4}^{-1}$	7	3’’43’’2’’3’4’’51’’5’4’’3’4’’51’’5’4’’3’’2’’3’’4’3’’	21
	A	H	B			3’’43’’2’’3’’4’’51’’5’4’’34’’51’’5’4’’32’’3’’4’3’’	
7	A	B	I	$G4_{1}^{1}E3_{3}^{-1,-1}G4_{1}^{-1}E3_{3}^{-1,1}$	4	5’’12’’34’’3’2’’1’2’’34’’3’2’’5’’	14
	A	I	B			5’’2’’34’’3’2’’12’’34’’3’2’’1’5’’	
8	A	B	J	$E3_{3}^{1,-1}G4_{1}^{-1}E3_{3}^{1,1}G4_{1}^{1}$	4	2’’5’’3’4’’32’’1’2’’3’4’’32’’15’’	14
	A	J	B			5’’1’2’’3’4’’32’’12’’3’4’’35’’2’’	
9	A	C	E	$G4_{5}^{-1}G4_{2}^{1}E3_{4}^{-1,-1}G4_{2}^{1}E3_{4}^{-1,1}G4_{2}^{2}G4_{5}^{1}$	7	4’’5’4’’1’’23’’45’’4’3’’23’’45’’4’3’’2’’1’’4’’54’’	21
	A	E	C			4’’5’4’’1’’2’’3’’45’’4’3’’2’3’’45’’4’3’’2’1’’4’’54’’	
10	A	C	F	$E3_{5}^{-1,-1}G4_{3}^{-1}E3_{5}^{-1,1}G4_{3}^{1}$	4	4’’2’’51’’5’4’’3’4’’51’’5’4’’32’’	14
	A	F	C			2’’3’4’’51’’5’4’’34’’51’’5’2’’4’’	
11	A	C	G	$G4_{4}^{-1}E3_{1}^{-1,1}G4_{4}^{2}E3_{1}^{-1,-1}G4_{4}^{-1}$	5	3’’4’5’’2’’12’’1’5’’4’’5’’12’’1’2’’5’’4’3’’	17
	A	G	C			3’’45’’2’’12’’1’5’’4’’5’’12’’1’2’’5’’43’’	
12	A	C	H	$G4_{4}^{-1}E3_{1}^{-1,1}G4_{4}^{1}E3_{1}^{-1,-1}$	4	3’’4’5’’2’’12’’1’5’’45’’12’’1’2’’5’’3’’	16
	A	H	C			3’’5’’2’’12’’1’5’’4’5’’12’’1’2’’5’’43’’	
13	A	C	J	$G4_{2}^{1}E3_{4}^{-1,-1}G4_{2}^{1}E3_{4}^{-1,1}G4_{2}^{2}$	5	1’’23’’45’’4’3’’23’’45’’4’3’’2’’1’’	15
	A	J	C			1’’2’’3’’45’’4’3’’2’3’’45’’4’3’’2’1’’	
14	A	D	E	$E3_{5}^{1,-1}G4_{3}^{2}E3_{5}^{1,1}G4_{3}^{2}$	4	4’’2’’5’1’’51’’4’’3’’4’’1’’5’1’’54’’3’’2’’	16
	A	E	D			2’’3’’4’’5’1’’51’’4’’3’’4’’1’’5’1’’52’’4’’	
15	A	D	F	$G4_{3}^{-1}E3_{5}^{1,-1}G4_{3}^{-1}E3_{5}^{1,1}G4_{3}^{2}$	5	2’’3’4’’5’1’’54’’3’4’’5’1’’54’’3’’2’’	15

	A	F	D			2’’3’’4’’5’1’’54’’34’’5’1’’54’’32’’	
16	A	D	G	$G4_4^{-1}G4_2^1E3_4^{-1,-1}G4_2^{-1}E3_4^{-1,1}G4_4^1$	6	3’’4’3’’1’’23’’45’’4’3’’2’3’’45’’4’1’’43’’	18
	A	G	D			3’’4’1’’45’’4’3’’23’’45’’4’3’’2’1’’3’’43’’	
17	A	D	H	$G4_4^2E3_1^{-1,1}G4_4^2E3_1^{-1,-1}$	4	3’’4’’5’’2’’12’’1’5’’4’’5’’12’’1’2’’5’’3’’	16
	A	H	D			3’’5’’2’’12’’1’5’’4’’5’’12’’1’2’’5’’4’’3’’	
18	A	D	J	$E3_4^{-1,-1}G4_2^2E3_4^{-1,1}G4_2^2$	4	3’’1’’45’’4’5’’3’’2’’3’’5’’45’’4’3’’2’’1’’	16
	A	J	D			1’’2’’3’’45’’4’5’’3’’2’’3’’5’’45’’4’1’’3’’	
19	A	E	J	$G4_3^2E3_5^{1,1}G4_3^2E3_5^{1,-1}$	4	2’’3’’4’’1’’5’1’’54’’3’’4’’5’1’’51’’4’’2’’	16
	A	J	E			2’’4’’1’’5’1’’54’’3’’4’’5’1’’51’’4’’3’’2’’	
20	A	F	H	$G4_1^1E3_3^{1,1}G4_1^{-1}E3_3^{1,-1}$	4	5’’12’’4’’3’4’’32’’1’2’’3’4’’34’’2’’5’’	16
	A	H	F			5’’2’’4’’3’4’’32’’12’’3’4’’34’’2’’1’5’’	
21	A	F	J	$G4_3^1G4_1^{-1}E3_3^{1,-1}G4_1^1E3_3^{1,1}G4_3^{-1}$	6	2’’32’’5’’1’2’’3’4’’34’’2’’12’’4’’3’4’’5’’2’’	18
	A	J	F			2’’5’’4’’34’’2’’1’2’’4’’3’4’’32’’15’’2’’3’2’’	
22	A	H	J	$E3_3^{1,-1}G4_1^1E3_3^{1,1}G4_1^{-1}$	4	5’’2’’3’4’’34’’2’’12’’4’’3’4’’32’’1’5’’	16
	A	J	H			5’’12’’3’4’’34’’2’’1’2’’4’’32’’5’’	
23	B	D	F	$G4_1^1E3_1^{1,-1}G4_4^1E3_1^{1,1}G4_4^{-1}G4_1^{-1}$	6	5’’3’’2’’12’’5’’45’’2’’1’2’’15’’4’3’’5’’1’5’’	18
	B	F	D			5’’15’’3’’45’’1’2’’12’’5’’4’5’’2’’1’2’’3’’5’’	
24	B	D	H	$G4_3^1G4_4^2E3_1^{-1,-1}G4_4^2E3_1^{-1,1}G4_3^{-1}$	6	2’’32’’3’’4’’5’’12’’1’5’’4’’5’’12’’1’5’’3’’2’’3’2’’	20
	B	H	D			2’’32’’3’’5’’12’’1’5’’4’’5’’12’’1’5’’4’’3’’2’’3’2’’	

Permuting Strategy

With the ability to selectively permute any three edges, the general permuting strategy is as follows:

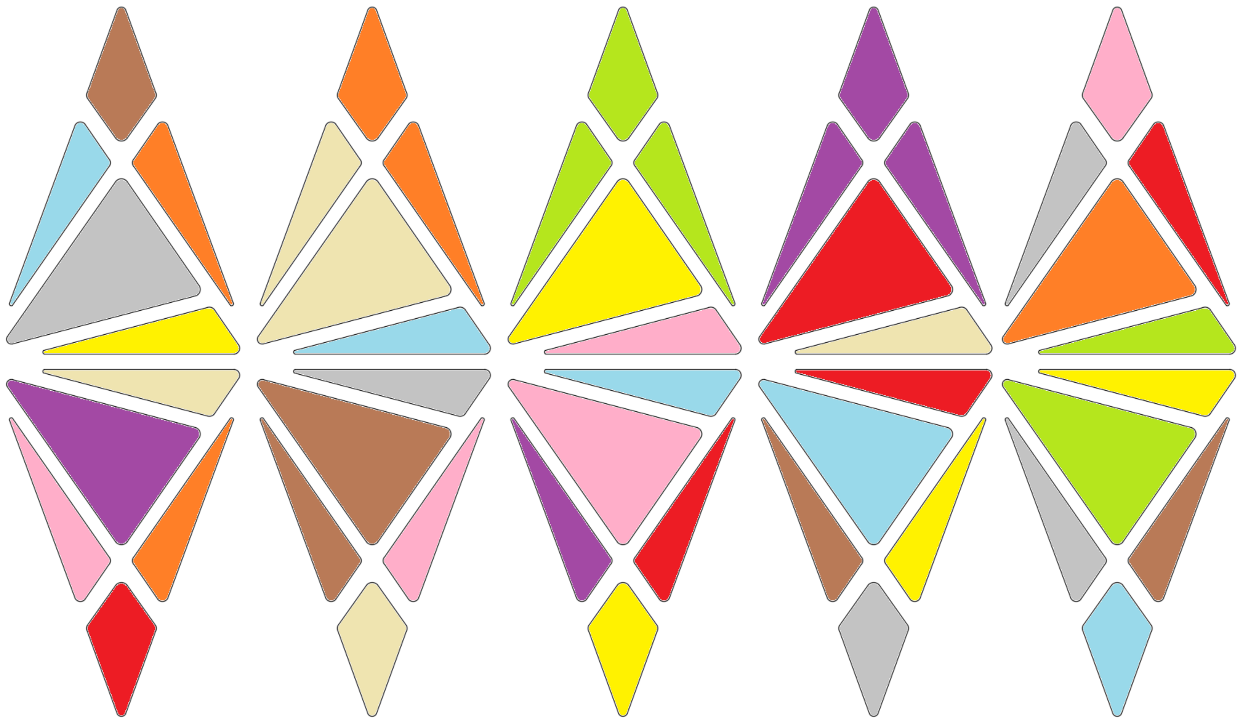
1. Identify one Face which is not correctly permuted. This will be one of the three Faces to be cycled.
2. Identify the target Location of the Face identified in step 1. The Face currently occupying this Location will also be one of the three Faces cycled.
3. Identify the target Location of the Face identified in step 2. The Face currently occupying this Location will be the third Face* in the 3-cycle.
4. Identify the congruent cycle through rigid body symmetry.
5. Perform the identified permutation.
6. Repeat steps 1-4 until all faces are permuted (up to 4 iterations may be required).

*In the event that the second Face piece (the piece identified step two) belongs in the location identified in step one (a two-cycle), any third incorrectly permuted face piece may be used to complete the 3-cycle.

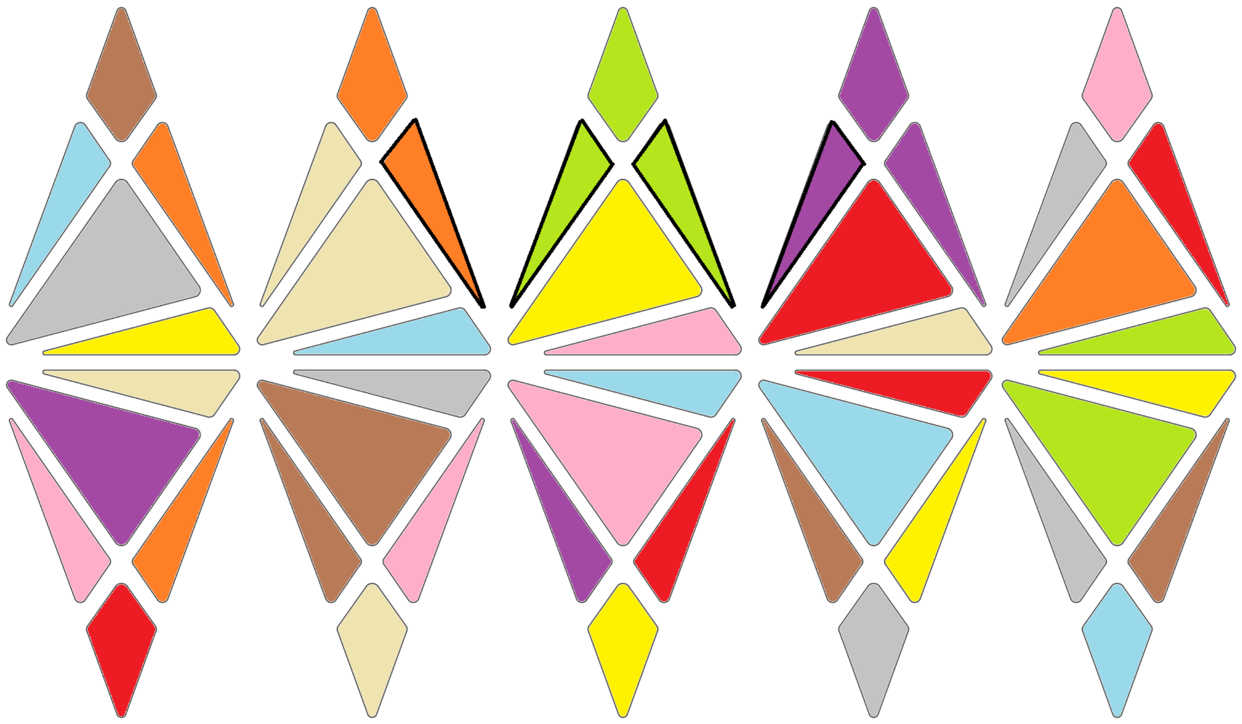
Exemplar Permutation Solve

We have now derived both a rigorous set of algorithms as well as a solving methodology and are ready to apply them to an example scrambled state.

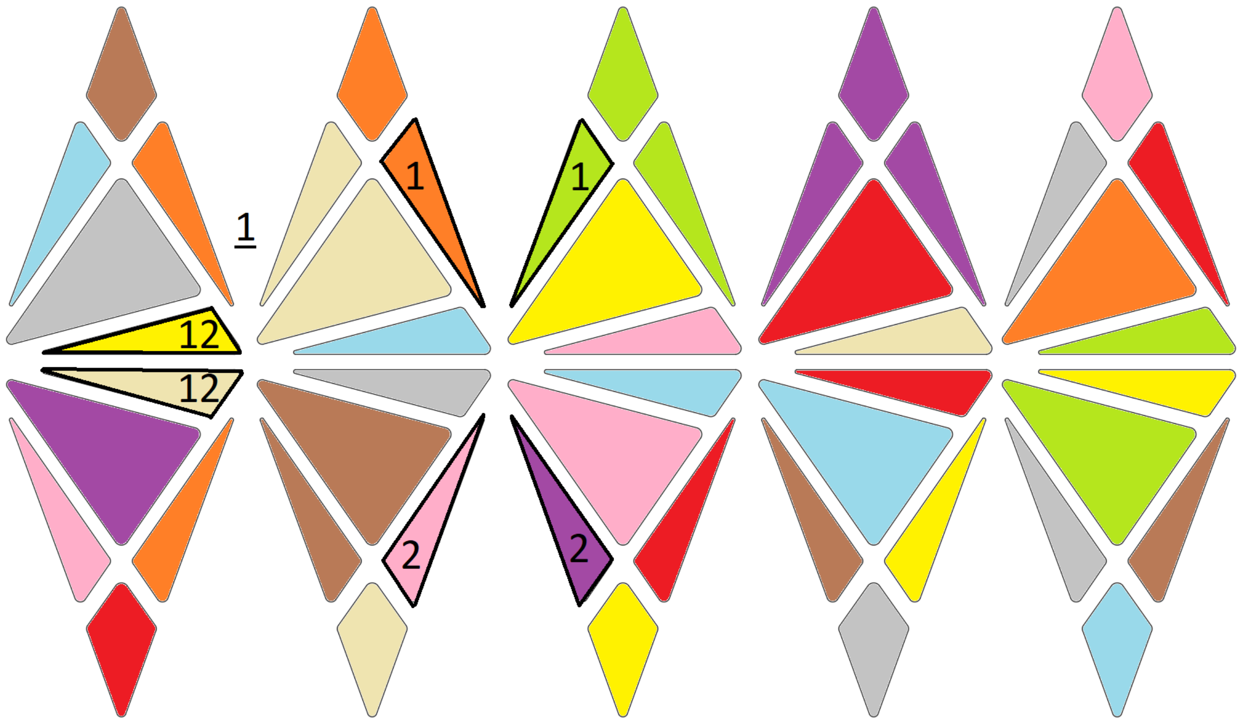
Figure 44 Starting Position



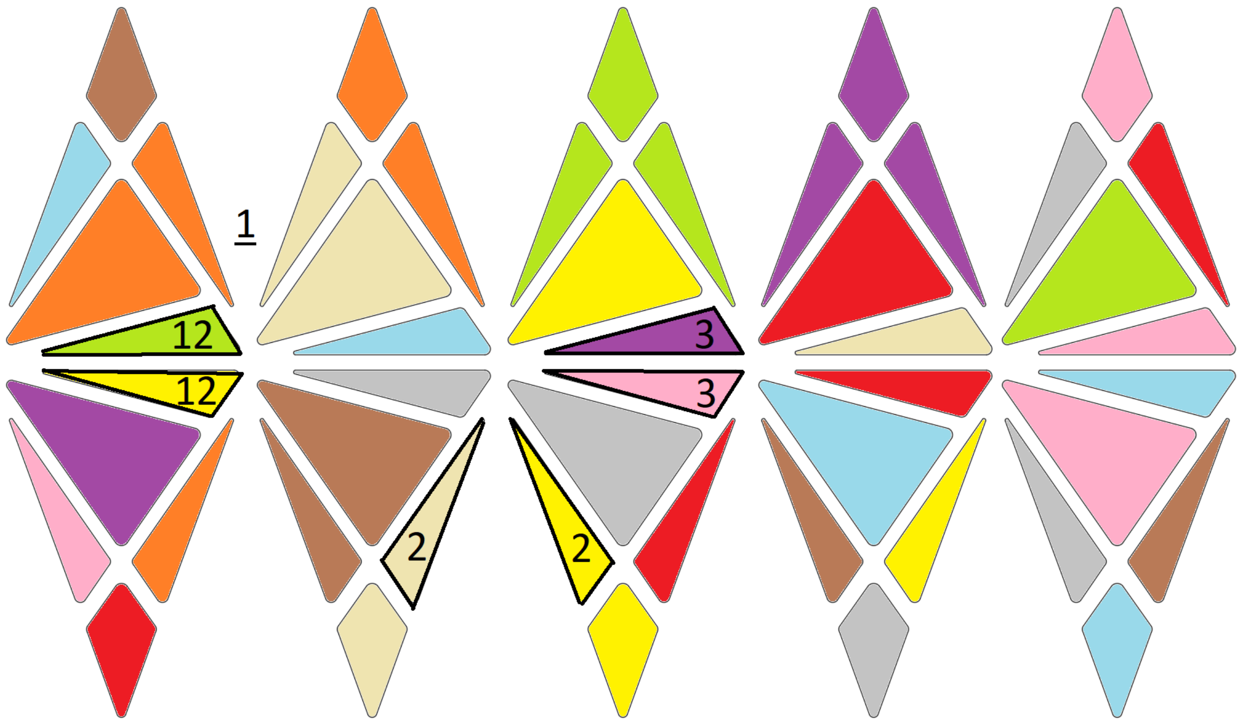
An initial analysis shows a total of two edges and zero faces are correctly permuted (Orange-Green and Green-Purple are the two correctly permuted edges). As the two edges are not polarly opposed, our solution method will only consider one edge as solved.



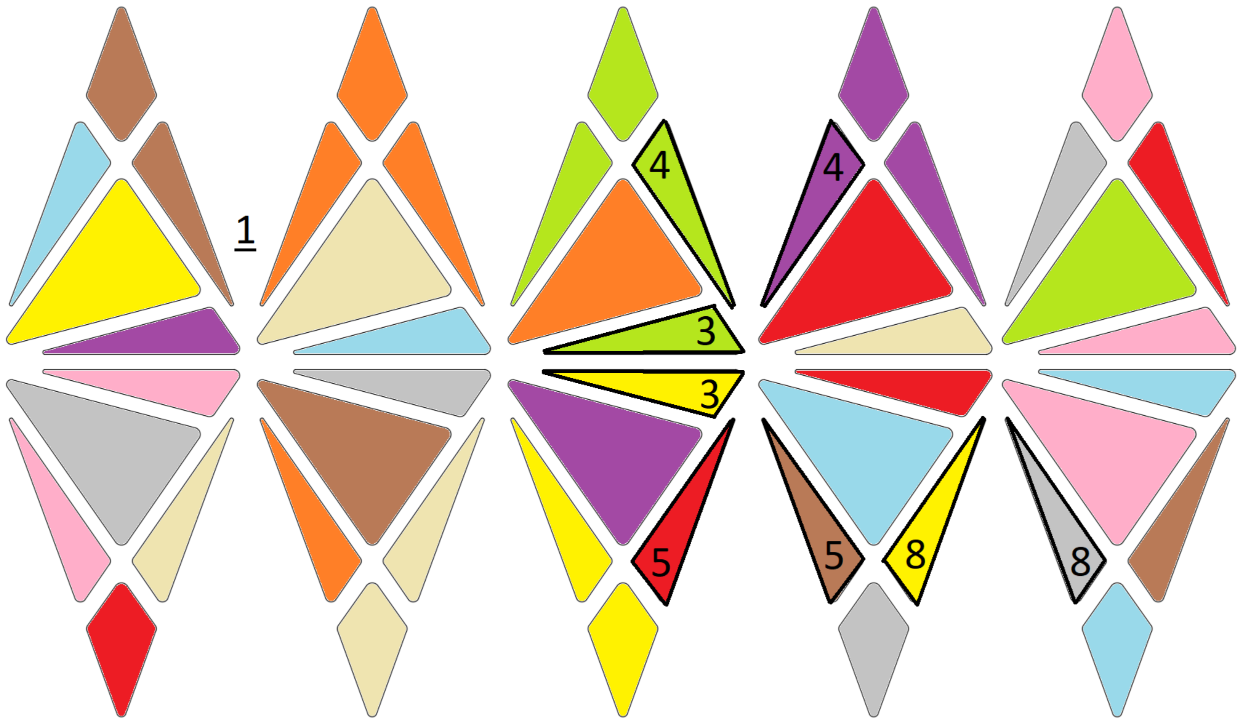
Either edge could be selected, but I have chosen to consider the Orange-Green edge as the one solved edge. That is to say that Edge 1 is located in Location 1. Edge 2, however, is located in Location 12. Edge 2 is to be permuted to Location 2 without disturbing the contents of Location 1 (Edge 1). Perform Permuting the Second Edge Location 12.



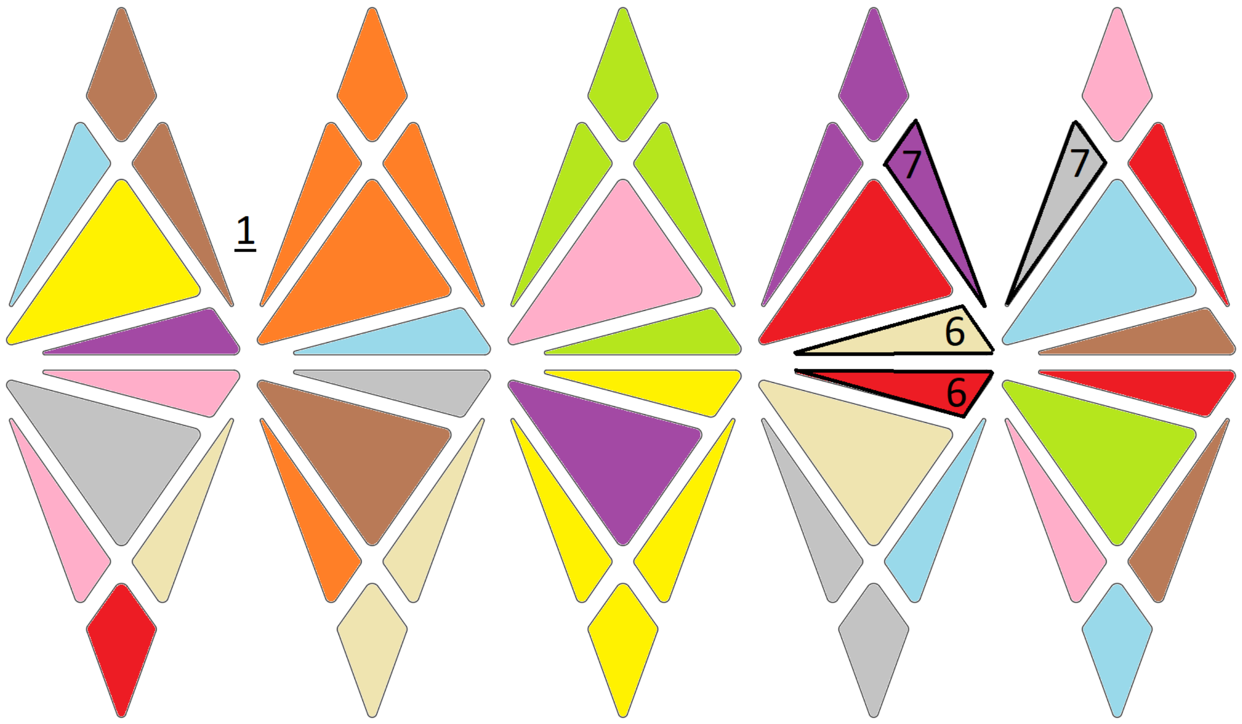
The results of Permuting the Second Edge Location 12 are shown below. Axis 1 is underlined for reference and will remain constant throughout the edge permutations.



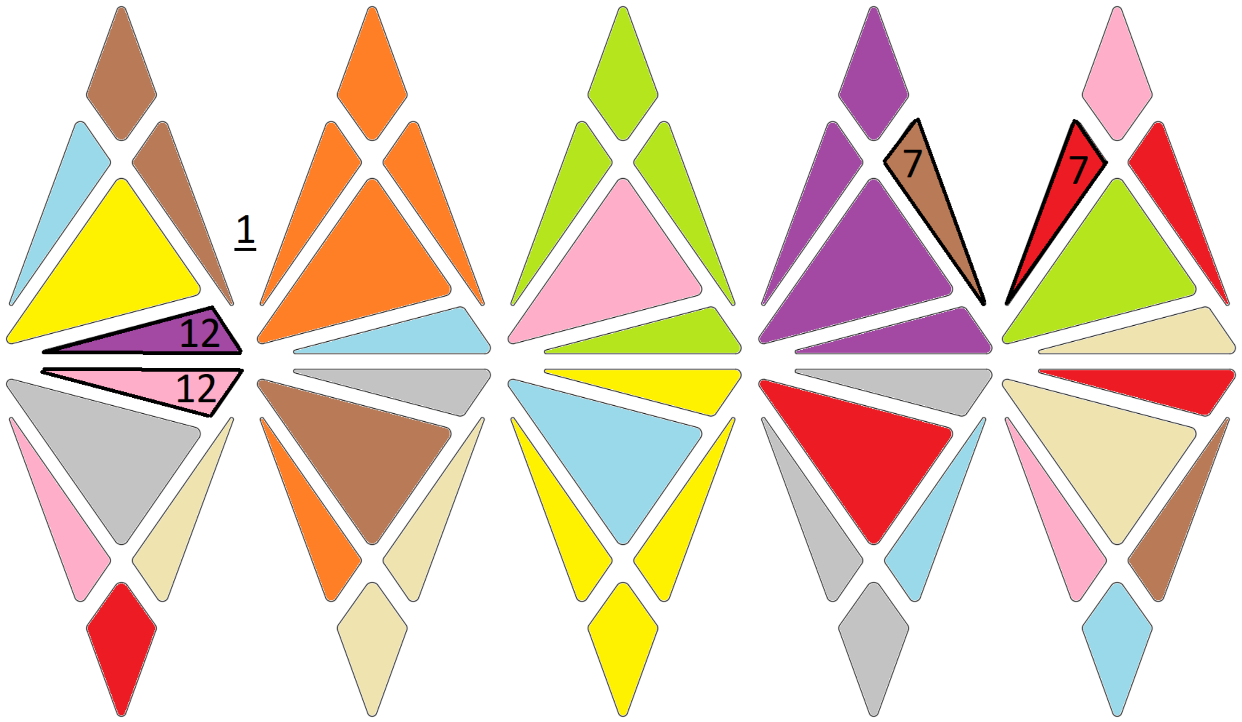
Edge 2 is now in Location 2 and we can turn our attention to Location 3. Edge 3 is in Location 12 and needs to be permuted to Location 3. Perform Permuting the Third Edge Location 12.



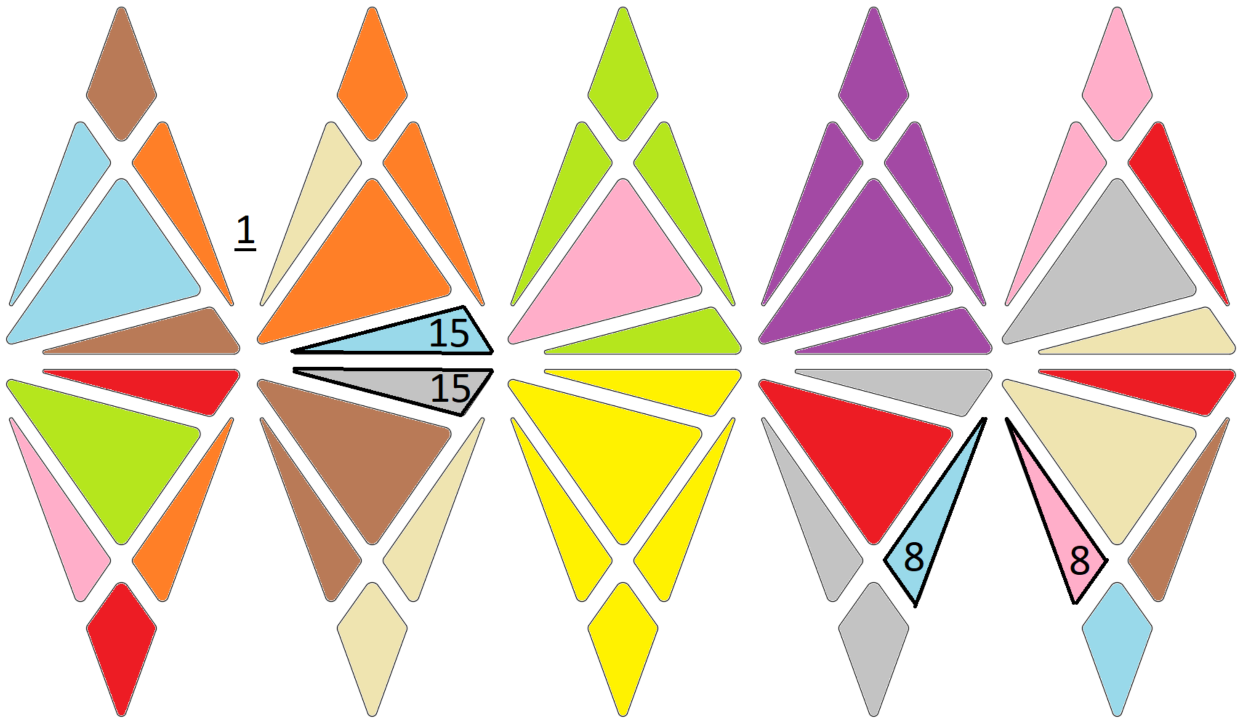
After permuting Edge 3, we turn our attention to Edge 4. Edge 4 is located in Location 4; Edge 4 does not need any permuting. Edge 5 is in location 8. Perform Permuting the Fifth Edge Location 8.



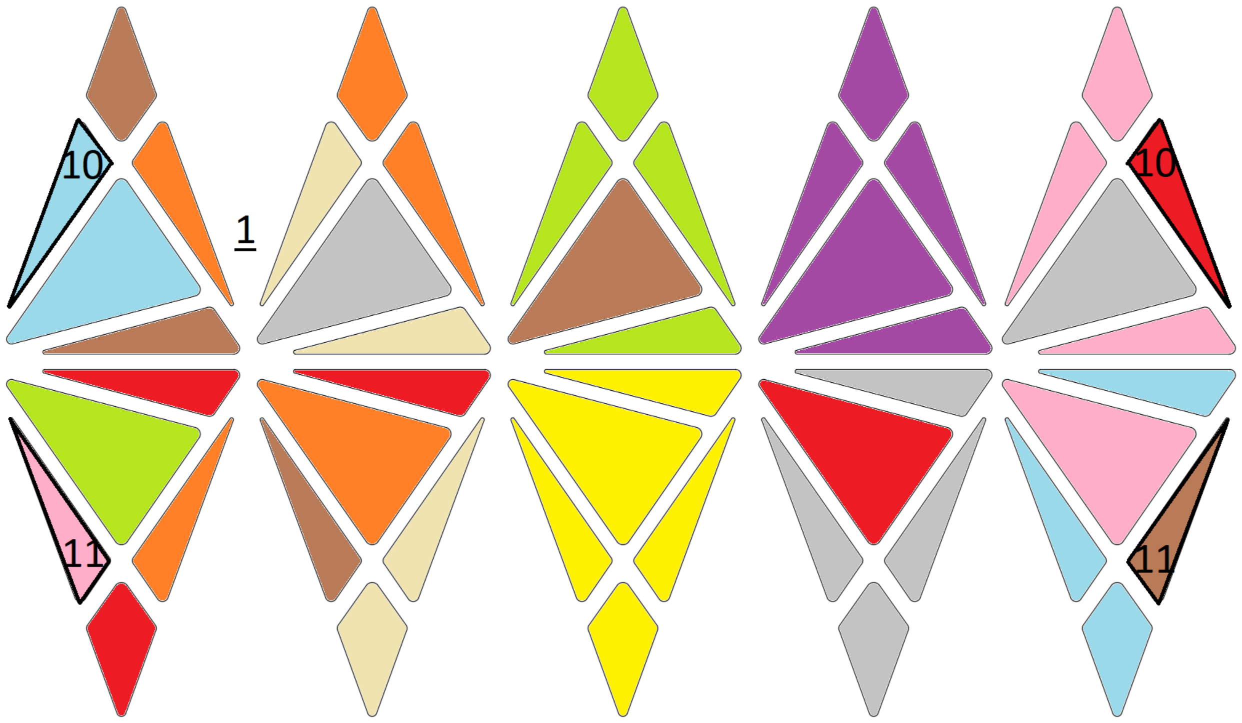
Perform Permuting the Sixth Edge Location 7.



Perform Permuting the Seventh Edge Location 12.

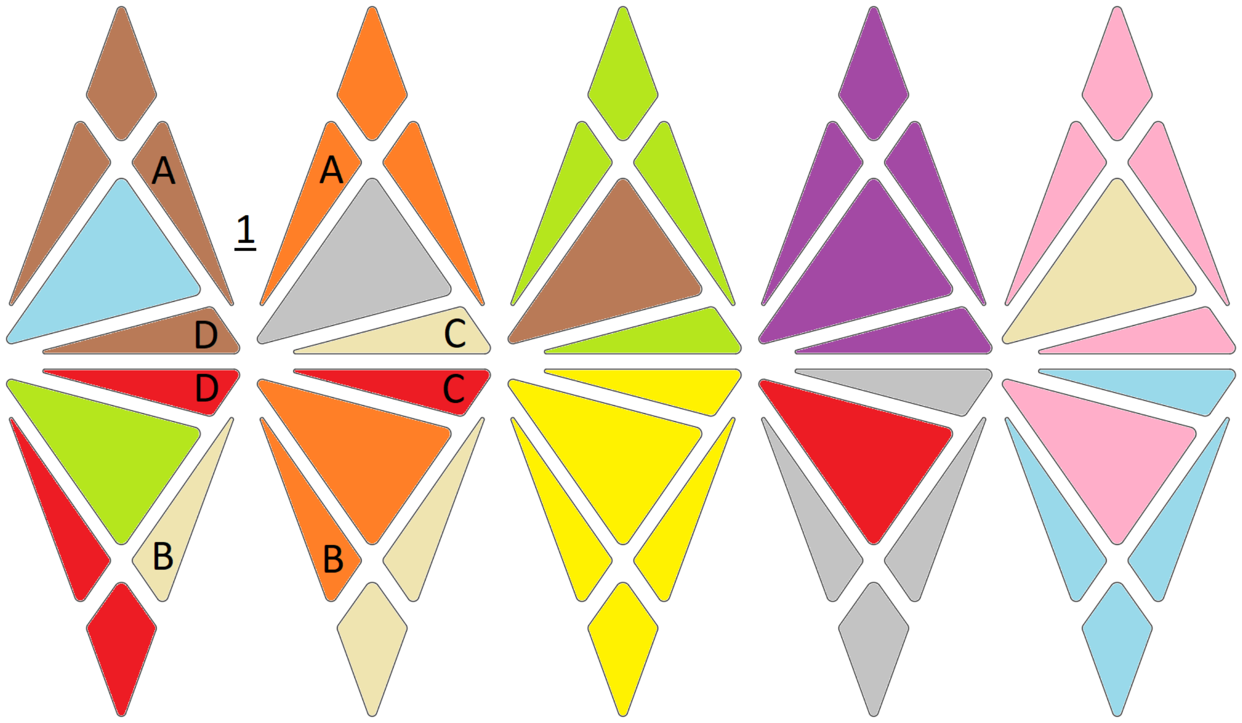


Perform Permuting the Eighth Edge Location 15.

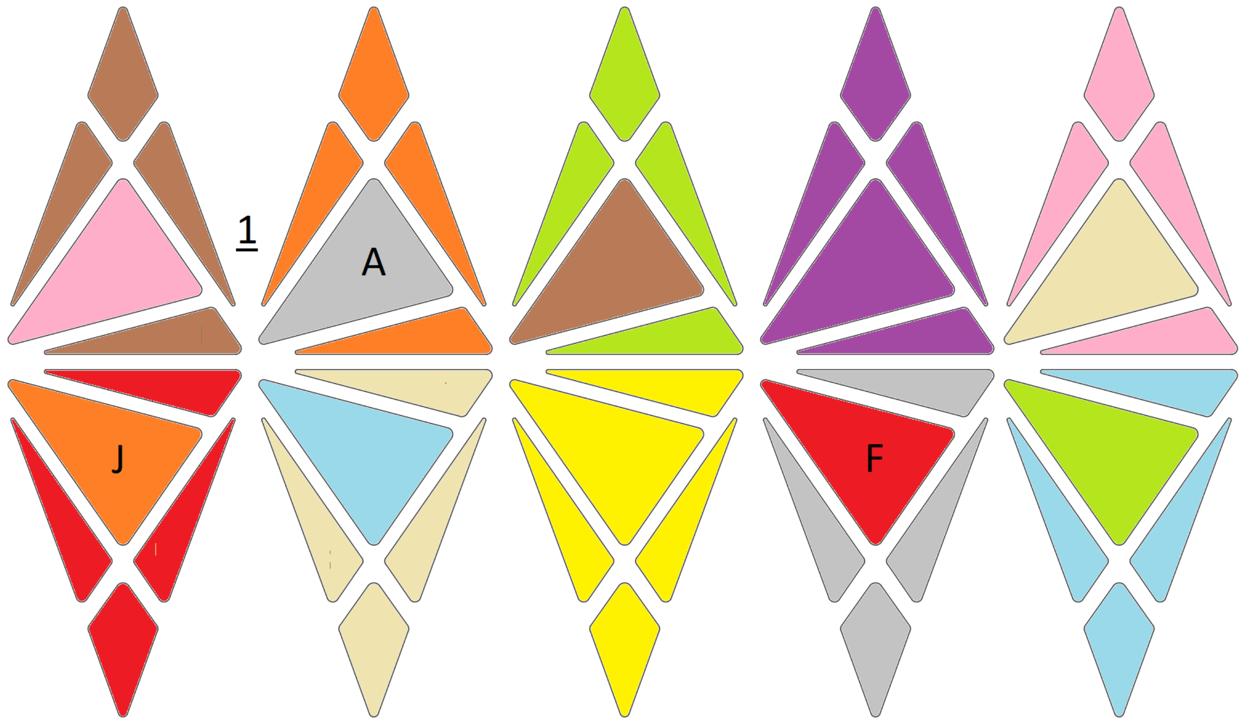


Perform Permuting the Tenth Edge Location 11. This also permutes Edge 11 to Location 11 (as well as correctly permuting Edge 12 and Edge 13). This brings us to the next phase: Permuting the Last Four Edges.

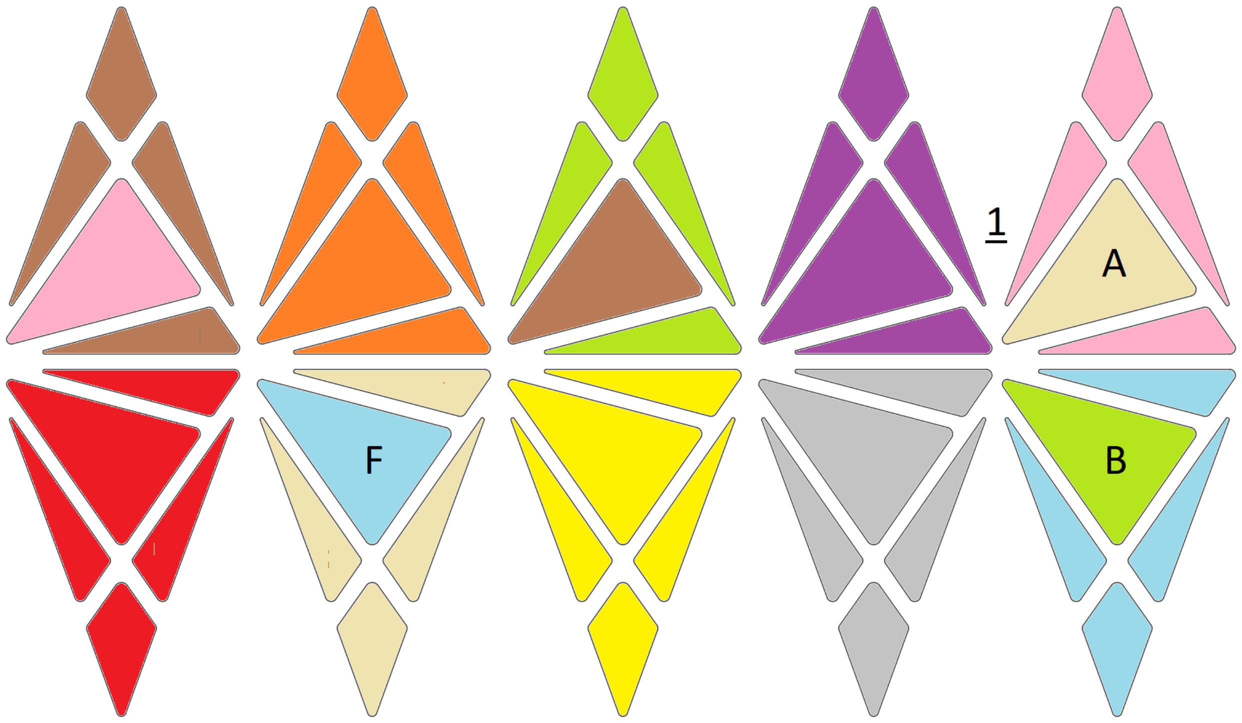
.



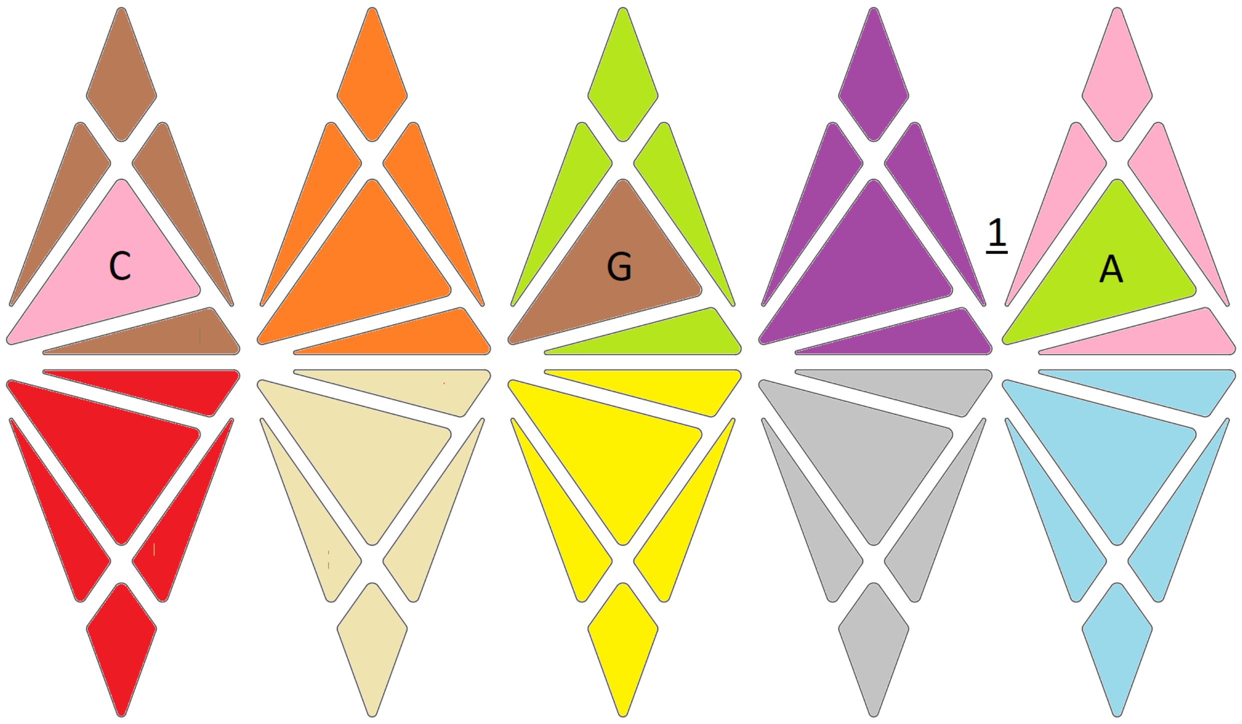
This particular case is: ACBD as Edge A is in Location a, Edge C is in Location b, Edge B is in Location c, and Edge D is in Location d. Perform the ACBD permutation. This brings us to the next phase: Permuting the Faces.



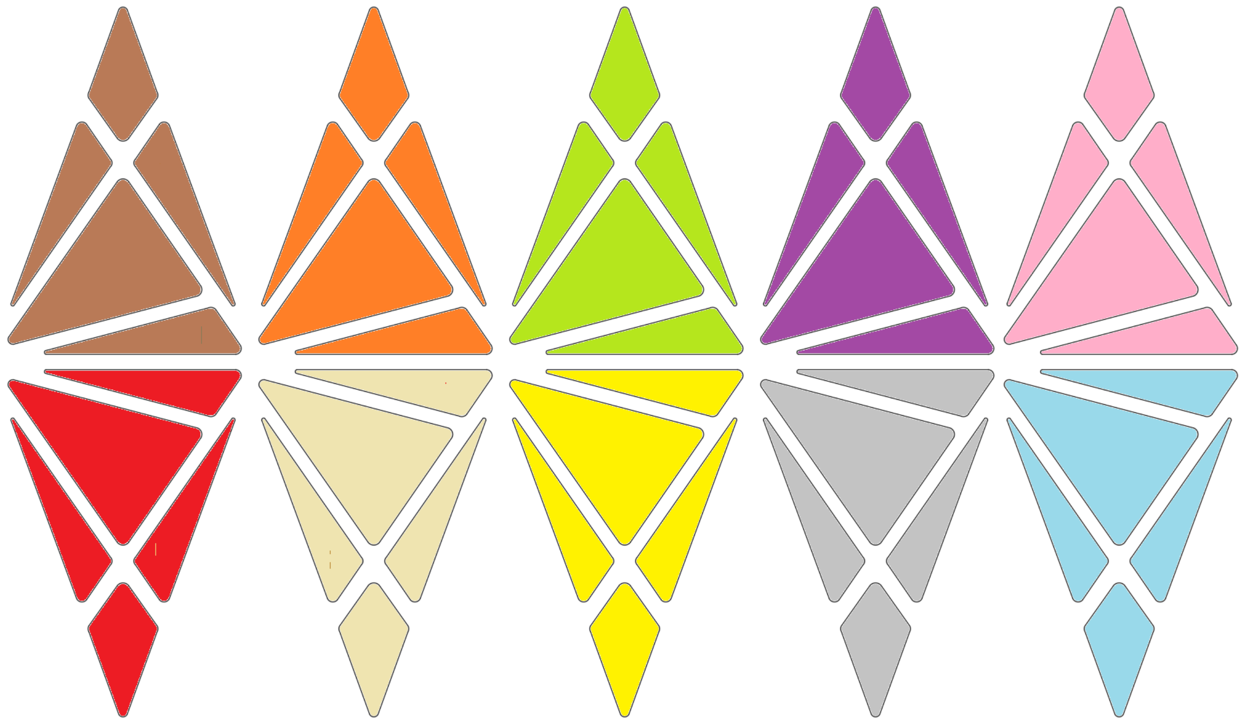
The first Face piece chosen in the Permuting Strategy is arbitrary. I chose the orange Face (labeled J) as the first Face. The target Location of the orange Face is the Location that currently holds the grey Face (labeled A). As such, the grey Face will be the second Face in the 3-cycle. The target Location of the grey Face is the Location that currently holds the red Face (labeled F). Face F is the third Face in the face 3-cycle. By placing axis 1 in the location above, the three faces can be permuted using an (A,F,J) cycle. Perform an (A,F,J) permutation.



Again, the choice of the first Face is arbitrary and can be any of the incorrectly permuted Faces. I have chosen the beige Face (labeled A) to be the first Face in the 3-cycle. The target Location for the beige Face currently holds the blue Face (labeled F). The blue Face is the 2nd piece in the face 3-cycle. The target Location for the blue Face currently contains the green Face (labeled B). The green Face will be the last piece in the face 3-cycle. The 1 axis is indexed to allow the three faces to be permuted under an (A,F,B) cycle.



With only 3 incorrectly permuted faces remaining, the choice of which 3 face pieces to permute next is predetermined. The 1 axis is indexed to allow for an (A,G,C) permutation.



All of the edge and face pieces have been permuted. As orientation was necessarily previously solved, the puzzle is now solved.

Quantifying Difficulty

Number of Unique Positions

One of the most common metrics given for assessing the "difficulty" of a permutation puzzle is its number of unique positions. This metric tends to correlate stronger with tedium than difficulty, but it is by far the easiest "difficulty" metric to calculate.

Any puzzle that exists will exist in at least one state. The lower bound of "Number of Unique Positions" is 1. There is no upper bound. I prefer to express the number of unique positions on a logarithmic (base 10) scale with three significant figures. This puzzle has $\frac{15! * 10!}{2} * 16 * 31 \approx 10^{21.07}$ unique positions. As such, I give this puzzle a unique position score of 21.1

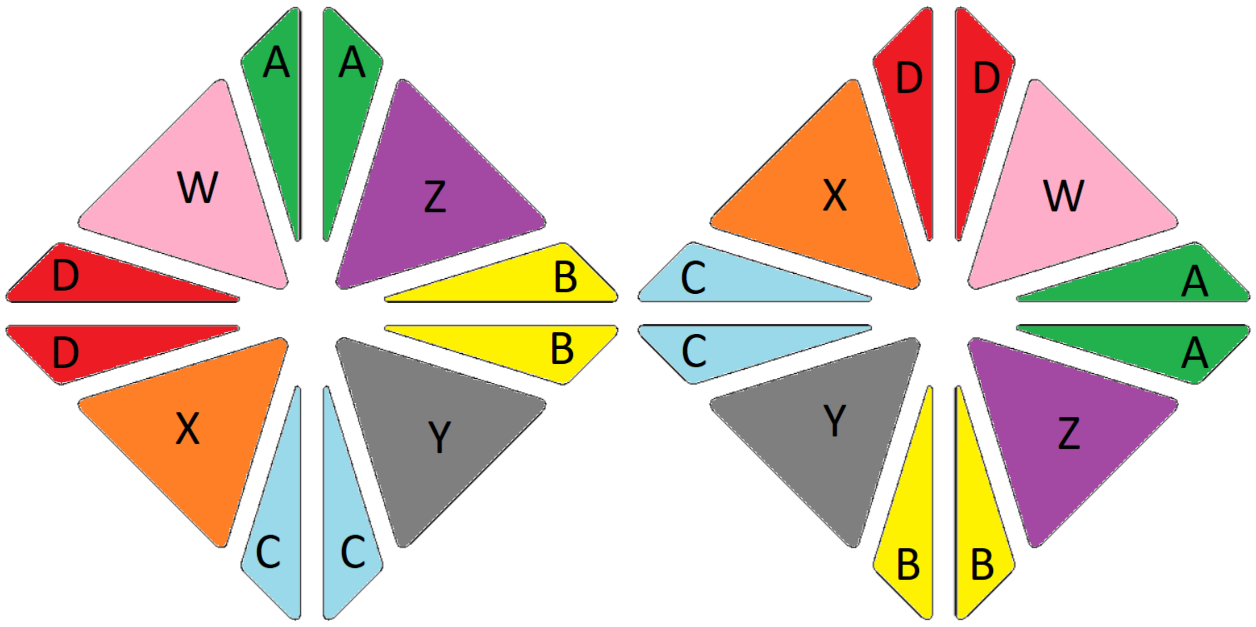
Number of Unique Permutations

Assuming standard orientation, the edge pieces can be arranged in $15!$ different ways while the face pieces can be arranged in $10!$ different ways. This would imply that the number of ways to uniquely permute the pieces of the puzzle would be $15! * 10!$

Parity

Every possible move of this puzzle has even parity. That is to say if we decomposed any given move down to an expression of a series of two-cycles, there will always be an even number of two cycles.

Figure 45 Parity Example



Going from the left image to the right, the applied cycle is (A,B,C,D)(Z,Y,X,W). These can be decomposed into the following series of two cycles: (A,B)(A,C)(A,D)(Z,Y)(Z,X)(Z,W). The decomposition has a total of 6 two-cycles. As 6 is even, this move has an even parity.

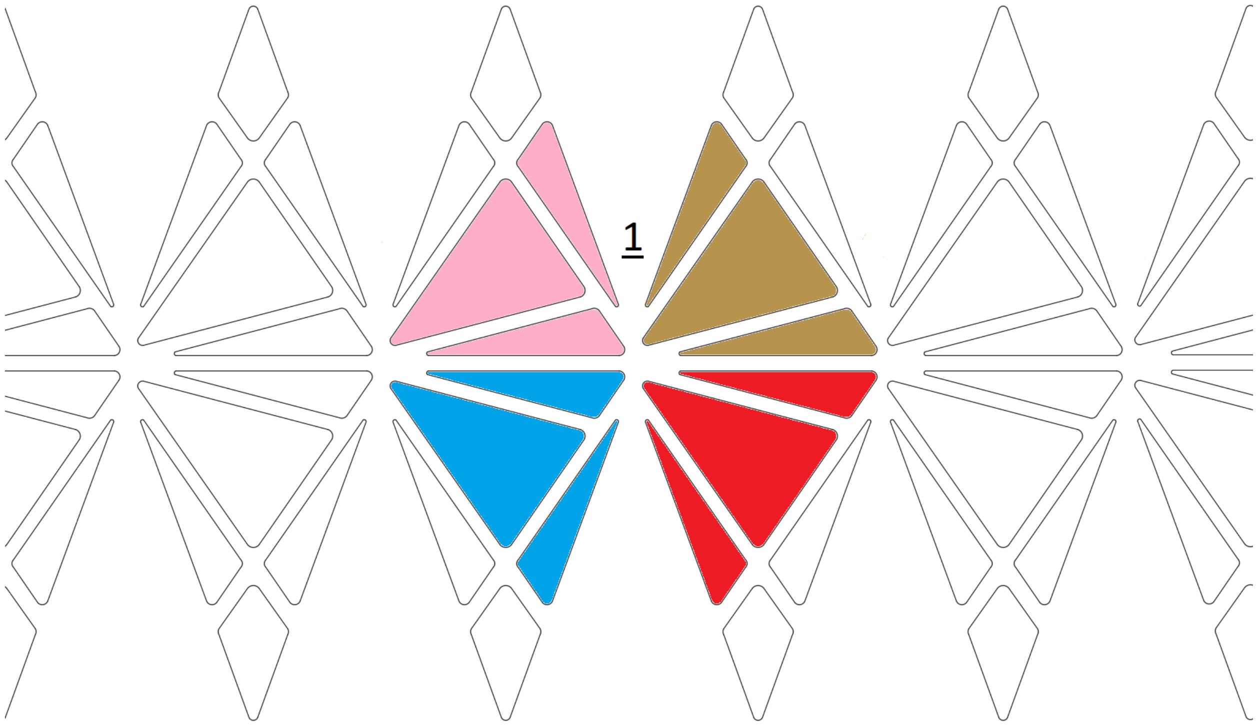
Regardless of the orientation of the individual pieces, every possible single axis move will necessarily involve four edge pieces alternating with four face pieces in the following cycle: $(\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D})^n\ (\mathrm{Z}, \mathrm{Y}, \mathrm{X}, \mathrm{W})^n$, where n is an integer. As such, the above example case is sufficient to prove that every possible move has even parity. Furthermore, as all algorithms are necessarily composed of a series of possible moves, all algorithms will necessarily inherit this even parity; the sum of even numbers is even.

A puzzle with even parity is unable to swap just two pieces (a single 2-cycle) but a pair of two cycles is possible (a double 2-cycle). As such, the parity of the edge pieces must be the same as the parity of the face pieces. If it takes an odd number of edge 2-cycles to permute the edge pieces, it must also take an odd number of 2-cycles to permute the face pieces (or alternatively even and even).

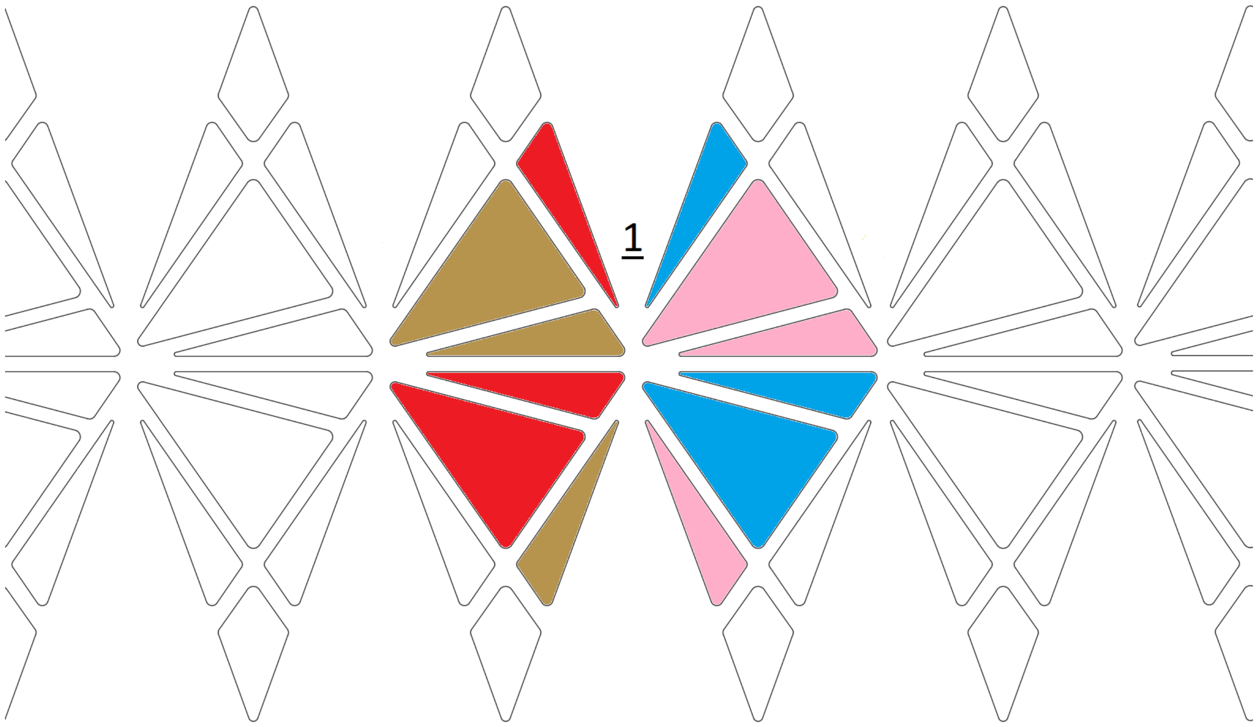
Half of the “possible” $15! * 10!$ arrangements would have mismatched parity between face and edge pieces. The true quantity of possible permutations (ignoring orientation) is $\frac{15! * 10!}{2}$

Orientation

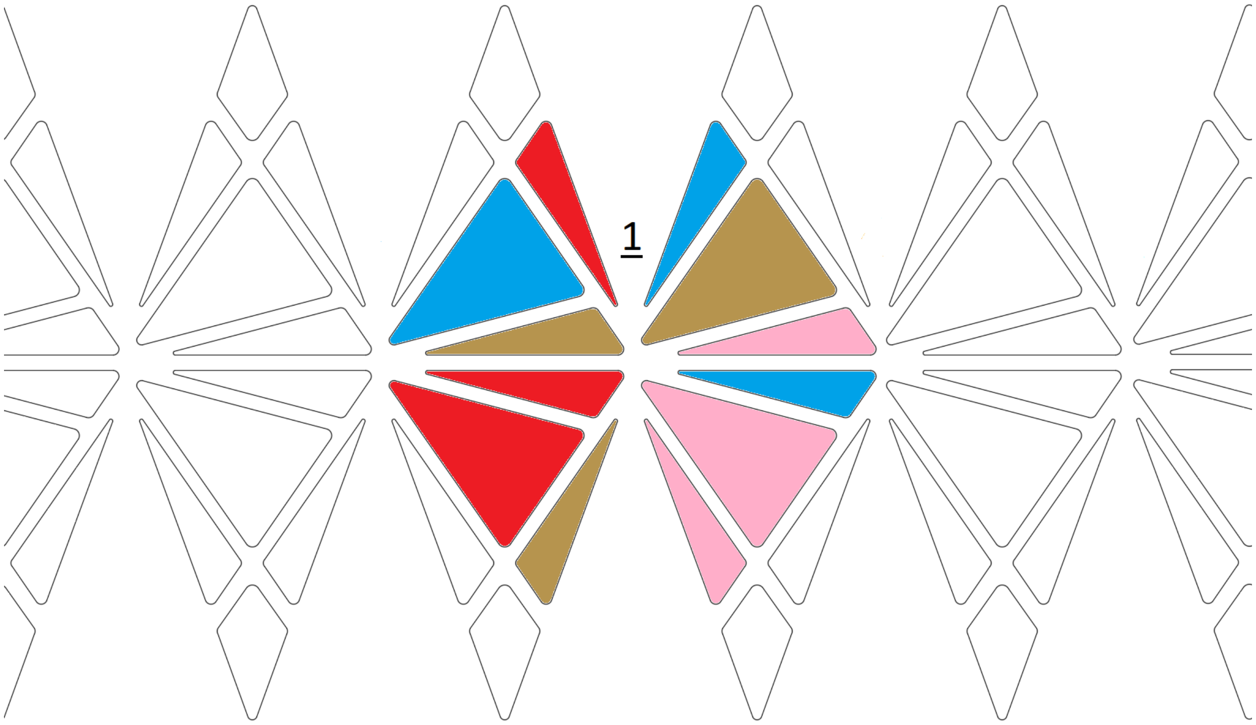
Equatorial Orientation



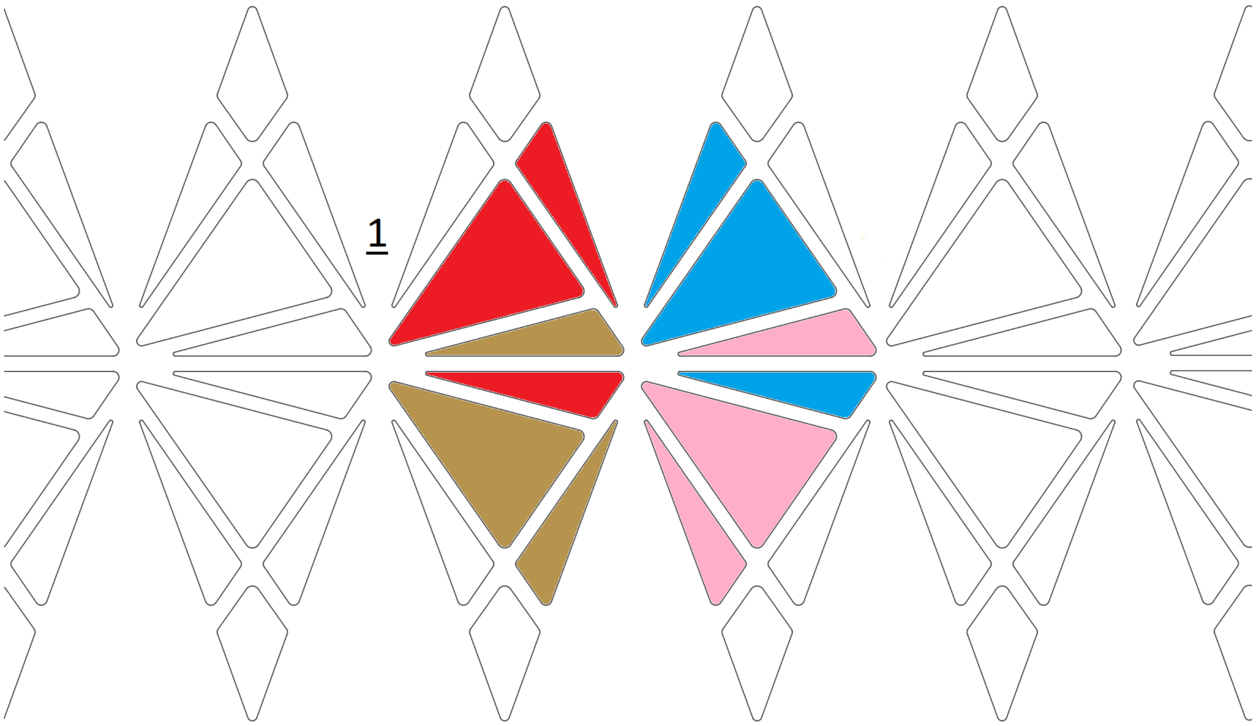
Perform Permuting the Last Four Edges case (C,D,A,B).



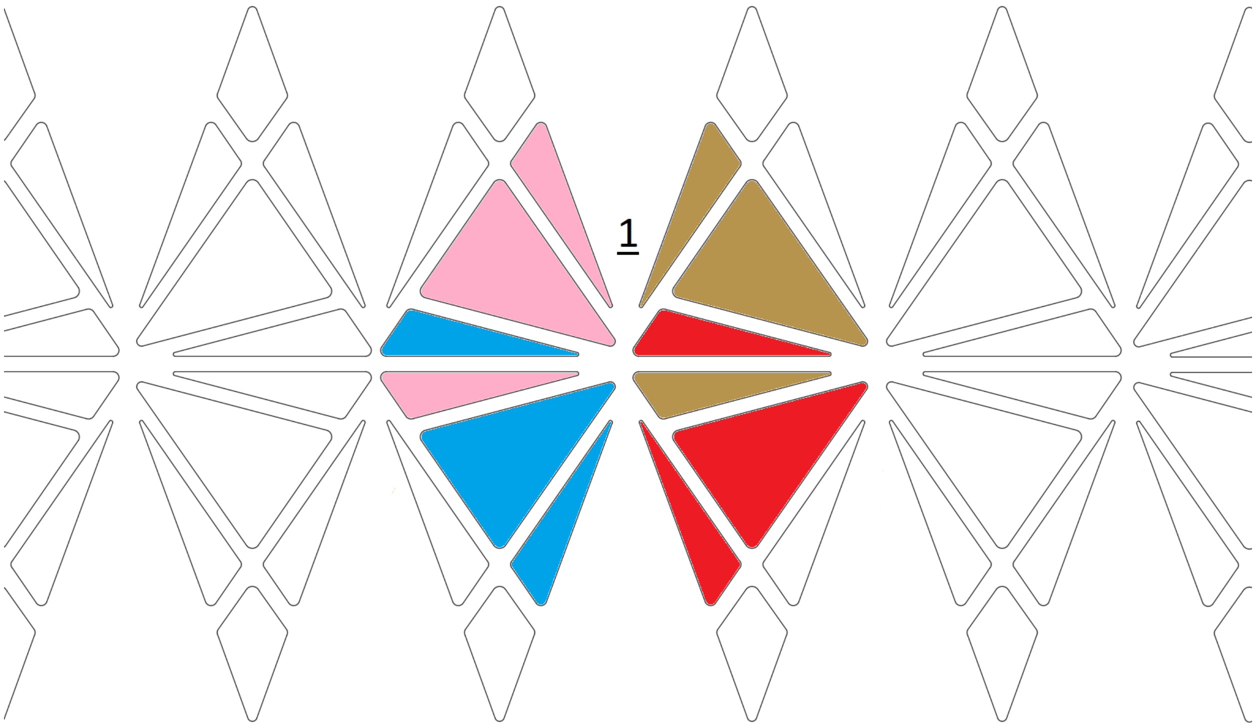
Perform Selectively Permuting Any 3 Faces (A,B,I)



Perform Selectively Permuting Any 3 Faces (A,C,B)



Reorient with a 1′′



The end result is a maintained Global Permutation, a maintained Polar Orientation, and two adjacent equatorial edges having their orientation flipped (along with the required reorientation of the four faces). There is one Equatorial Orientation with Standard Orientation, five Equatorial Orientations with two adjacent edges flipped, five Equatorial Orientations with two non-adjacent edges flipped, and five Equatorial Orientations with four edges flipped. As such, there are 16 times as many possible arrangements of pieces when Equatorial Orientation isn't necessarily maintained. Note: this is half of the maximum 2^5 cases outlined in The Size of the Equatorial Orientation Space as half of the cases are covered by equatorial mirror symmetry.

Polar Orientation

When accounting for Polar Orientation, there is one configuration which maintains Polar Orientation. There are also five axes which could be rotated 90-degrees in one of two directions. For each of those ten possible moves, there are 5 possible follow-up actions: either no further rotation or rotation of either the $n+2$ axis or $n-2$ axis by ±90-degrees. This results in a multiplier of 31 when Polar Orientation isn't necessarily maintained as: $1+(5*2)(1+2)=31$

Algorithms

This is a tricky metric to define and my definition is imperfect. There is a concept called the "Devil's Algorithm" which is a single algorithm that cycles through every possible state of a puzzle at least once. By that definition, every puzzle can be solved with only one (impossibly long and complex) algorithm. This really defeats the purpose of a comparison metric when every puzzle has a score of "1". Alternatively, the length of the "Devil's Algorithm" might be a reasonable metric but could be very difficult to calculate and would likely be strongly correlated to the Number of Unique Positions a puzzle can exist in. Instead, this metric will report the number of characteristics which an algorithm set must handle in order to be considered complete:

Orientation

At a minimum, there needs to be at least one algorithm that:

1. Orients either the edges or the faces (their orientations are coupled such that orienting one orients the other)

Permutation

At a minimum, there needs to be at least one algorithm that:

1. Permutes edge pieces with an odd Parity
2. Permutes face pieces to with odd Parity
3. Cycles a different number of edge pieces and face pieces

A single algorithm may possess multiple characteristics; Devil's Algorithm necessarily possesses all of these characteristics. For example, G4 satisfies the two "odd parity" requirements; G4 has an even parity but 4-cycles the edges and 4-cycles the faces. E3 satisfies the asymmetric n-cycle requirement; it 3-cycles the edge pieces and 5-cycles the face pieces. G2 is not strictly necessary as all permutation requirements are met with G4 and E3, but G2 greatly simplifies the setup conjugates and affects the Required and Expected Number of Moves as well as the Required Memory.

Required Memory

While Algorithms deals with characteristics which must be addressed by the algorithm set, Required Memory deals with the "memory" allocation needed to record the algorithm set. This correlates moderately well to the difficulty of memorizing *a solution set* but is specific to said solution set. Based on the following calculations, 52 units of memory are required to perform the solution described in this paper.

Read-Only Memory

The first part of the "memory" calculations will deal with read-only memory. A unit of read-only memory will be assessed for each of the following:

1. Each base algorithm (example: the "G4" algorithm would account for 1 unit of memory)
2. Each conjugate or commutator structure used in a base algorithm (example: every { , } or [,] counts as 1 unit of memory)
3. Each slot for a move in a base algorithm (example: (n-1)''(n)(n-1)'' would count for 3 units of memory. The algorithm would also be assessed 3 memory units if written as {(n-1)'',(n)}: 1 unit for the conjugate { , }, and 1 each for both the (n-1)'', and (n) move slots)
4. Each variable input for a base algorithm, except for variables corresponding purely to Sub-cycle Iteration (examples: $G4_1^3$ would count for 1 unit as the 1 is a variable and the 3 corresponds to Sub-cycle Iteration, $E3_3^{1,-1}$ would count for 2 units as the 3 and the 1 are variables and the -1 corresponds to an inverse, and $G2_3^4$ *should* count for 2 units as 4 and 3 are variables but will only count for 1* unit for reasons discussed below)
5. One unit will be assessed for each algorithm that could have cycle iteration or looping (example: the "p" of G4 and the "m" of E3 both correspond to the number of times a portion should be looped. G2 is its own inverse and as such will never be looped)

Table 16 Read-Only Memory

Name	Algorithm	Structures	Slots	Variables	Loops	Units
Polar	Polar = n	0	1	1	0	3
Equatorial	Solving Equatorial Orientation	0	3	2	0	6
G4	$G4_n^p = \{(n-1)'', (n)^p\}$	1	2	1	1	6
E3	$E3_q^{n,m} = \{(q-1)''(q+2)'', [(q+1)'', (q)^n]^m\}$	2	4	2	1	10
G2*	$G2_{n-1}^n = \{(n)'', (n-1)''\}$	1	2	1*	0	5

Polar Orientation

Polar Orientation requires one (trivial) algorithm: selectively turn an axis by 90 degrees. The Polar Orientation algorithm consists of only one move slot and one variable (which axis to index to).

Equatorial Orientation

Equatorial Orientation requires one algorithm, which is the process described in Solving Equatorial Orientation. The Equatorial Orientation requires the solver to correctly orient the axes and identify where in the algorithm the puzzle currently exists. I consider these 2 distinct and sufficient variables in the Read-Only Memory analysis. The solution presented in this paper is assessed 30 units of read-only memory.

Writable Memory

The second part of the "memory" calculation will deal with writable memory. The total size of the writable memory is the minimum number of units required to express any possible algorithm required to perform incremental progress as defined by a given solution method. A unit of writable memory will be assessed for each of the following:

1. Each base algorithm called from the read-only memory
2. Each variable associated with each instance of a base algorithm
3. Each conjugate or commutator structure

Table 17 Writable Memory Cost of Each Base Algorithm

Name	Cost in Writable Memory Units
G4	3
E3	4
G2	2

The most expensive incremental progress that could be needed occurs in Expected Number of Moves: Permuting the Last Four Edges index 5.

#	Expression	Structures	G4	E3	G2	Cost
5	$\{G2_4^5, E3_1^{1,1}\}\{G2_2^1 G2_3^4, G4_1^{-1}\}\{G2_4^5, E3_1^{-1,-1}\}$	3	1	2	4	22

*For memory assessment purposes, it is more efficient to express G2 as seen in Table 16 which excludes the $G4_n^2$ variant.

Required and Expected Number of Moves

Required Number of Moves

The number of moves required to solve a puzzle from its “most scrambled” state when using the most efficient solution is called the “God’s Number”. This is equivalent to the radius of the graph of unique positions for a given puzzle. The God’s Number of a puzzle is incredibly difficult to determine and has only been calculated for an exceptionally small subset of permutation puzzles. Instead, the current upper bound of the God’s Number can be reported using an asterisk to denote that the lower bound hasn’t been proven to equal the reported upper bound.

For example: In 1981, the most efficient algorithm to solve a 3x3x3 Rubik’s cube could need up to 53 moves (half-turn metric: a half-turn counts as one move) so it’s “Required Number of Moves” score would be reported as “53*”. By 2010, the God’s Number for a Rubik’s cube was proven to be 20, which would from then on be reported as “20”.

Expected Number of Moves

The subset of positions requiring God’s Number of moves to efficiently navigate to the solved position is typically very small. The “Expected Number of Moves” is the average number of moves to efficiently traverse from any position to the solved position most efficiently. Compared to God’s Number, this is even more difficult to determine but is useful in further conveying expected difficulty.

Without using God’s Algorithm, Expected Number of Moves and Required Number of Moves calculations are going to be specific to a given solution method. The following Expected Number of Moves analysis is an unoptimized conservative approximation. I’m not claiming that the method derived in this paper is optimized for the least number of moves. As such, an approximation of Expected Number of Moves should be sufficient.

Each of the following tables are calculated independently and with assumptions that simplify the estimations at the expense of overestimating move counts. For example, the direction of axis rotation while solving Polar Orientation could (but not necessarily) result in a more favorable Equatorial Orientation. This is not accounted for and results in an inflated estimate for “Expected Number of Moves.

Based on the instructions provided in this paper, the expected move count is approximately 178 moves with a worst-case scenario of 280 moves.

Expected Number of Moves: Polar Orientation

	Combinations: 31	Required Moves	Product: 50
5 Available Axes	1	0	0
3 Available Axes	10	1	10
2 Available Axes	20	2	40

This results in a maximum move count of 2 and a weighted average move count of $\frac{50}{31} \approx 1.61$.

Expected Number of Moves: Equatorial Orientation

	Combinations: 16	Required Moves	Product: 30
0 Anomalies	1	0	0
2 Anomalies (Touching)	5	1	5

1 Anomaly	5	2	10
2 Anomalies (Gapped)	5	3	15

This results in a maximum move count of 3 and a weighted average move count of $\frac{30}{16} = 1.875$.

Expected Number of Moves: Edge Permutation 1

	Combinations: 15	Required Moves	Product: 98
Location 1	1	0	0
Location 2	1	3	3
Location 3	1	3	3
Location 4	1	9	9
Location 5	1	9	9
Location 6	1	6	6
Location 7	1	6	6
Location 8	1	6	6
Location 9	1	8	8
Location 10	1	6	6
Location 11	1	6	6
Location 12	1	3	3
Location 13	1	11	11
Location 14	1	11	11
Location 15	1	11	11

This results in a maximum move count of 11 and a weighted average move count of $\frac{98}{15} \approx 6.533$.

Expected Number of Moves: Edge Permutation 2

	Combinations: 14	Required Moves	Product: 131
Location 2	1	0	0
Location 3	1	8	8
Location 4	1	12	12
Location 5	1	12	12
Location 6	1	9	9
Location 7	1	11	11
Location 8	1	11	11
Location 9	1	8	8
Location 10	1	9	9
Location 11	1	9	9
Location 12	1	9	9
Location 13	1	11	11
Location 14	1	11	11
Location 15	1	11	11

This results in a maximum move count of 12 and a weighted average move count of $\frac{131}{14} \approx 9.357$.

Expected Number of Moves: Edge Permutation 3

	Combinations: 13	Required Moves	Product: 66
Location 3	1	0	0
Location 4	1	9	9
Location 5	1	9	9
Location 6	1	6	6
Location 7	1	3	3
Location 8	1	3	3
Location 9	1	3	3
Location 10	1	6	6
Location 11	1	6	6
Location 12	1	3	3
Location 13	1	6	6
Location 14	1	6	6
Location 15	1	6	6

This results in a maximum move count of 9 and a weighted average move count of $\frac{66}{13} \approx 5.077$.

Expected Number of Moves: Edge Permutation 4

	Combinations: 12	Required Moves	Product: 79
Location 4	1	0	0
Location 5	1	3	3
Location 6	1	3	3
Location 7	1	11	11
Location 8	1	11	11
Location 9	1	6	6
Location 10	1	11	11
Location 11	1	11	11
Location 12	1	8	8
Location 13	1	6	6
Location 14	1	6	6
Location 15	1	3	3

This results in a maximum move count of 11 and a weighted average move count of $\frac{79}{12} \approx 6.583$.

Expected Number of Moves: Edge Permutation 5

	Combinations: 11	Required Moves	Product: 110
Location 5	1	0	0
Location 6	1	8	8
Location 7	1	14	14
Location 8	1	14	14
Location 9	1	14	14
Location 10	1	11	11
Location 11	1	11	11
Location 12	1	8	8

Location 13	1	9	9
Location 14	1	9	9
Location 15	1	12	12

This results in a maximum move count of 14 and a weighted average move count of $\frac{110}{11} = 10$.

Expected Number of Moves: Edge Permutation 6

	Combinations: 10	Required Moves	Product: 63
Location 6	1	0	0
Location 7	1	10	10
Location 8	1	10	10
Location 9	1	10	10
Location 10	1	3	3
Location 11	1	3	3
Location 12	1	3	3
Location 13	1	8	8
Location 14	1	8	8
Location 15	1	8	8

This results in a maximum move count of 10 and a weighted average move count of $\frac{63}{10} \approx 6.3$.

Expected Number of Moves: Edge Permutation 7

	Combinations: 9	Required Moves	Product: 77
Location 7	1	0	0
Location 8	1	6	6
Location 9	1	8	8
Location 10	1	14	14
Location 11	1	14	14
Location 12	1	9	9
Location 13	1	9	9
Location 14	1	9	9
Location 15	1	8	8

This results in a maximum move count of 14 and a weighted average move count of $\frac{77}{9} \approx 8.556$.

Expected Number of Moves: Edge Permutation 8

	Combinations: 8	Required Moves	Product: 80
Location 8	1	0	0
Location 9	1	8	8
Location 10	1	14	14
Location 11	1	14	14
Location 12	1	14	14
Location 13	1	11	11
Location 14	1	11	11

Location 15	1	8	8

This results in a maximum move count of 14 and a weighted average move count of $\frac{80}{8} = 10$.

Expected Number of Moves: Edge Permutation 9

	Combinations: 7	Required Moves	Product: 39
Location 9	1	0	0
Location 10	1	10	10
Location 11	1	10	10
Location 12	1	10	10
Location 13	1	3	3
Location 14	1	3	3
Location 15	1	3	3

This results in a maximum move count of 10 and a weighted average move count of $\frac{39}{7} \approx 5.571$.

Expected Number of Moves: Edge Permutation 10

	Combinations: 6	Required Moves	Product: 57
Location 10	1	0	0
Location 11	1	6	6
Location 12	1	9	9
Location 13	1	14	14
Location 14	1	14	14
Location 15	1	14	14

This results in a maximum move count of 14 and a weighted average move count of $\frac{57}{6} = 9.5$.

Expected Number of Moves: Edge Permutation 11

	Combinations: 5	Required Moves	Product: 58
Location 11	1	0	0
Location 12	1	16	16
Location 13	1	14	14
Location 14	1	14	14
Location 15	1	14	14

This results in a maximum move count of 16 and a weighted average move count of $\frac{58}{5} = 11.6$.

Expected Number of Moves: Permuting the Last Four Edges

	Combinations: 24	Required Moves	Product: 427
0	1	0	0
1	1	25	25
2	1	25	25
3	1	10	10

4	1	10	10
5	1	35	35
6	1	25	25
7	1	14	14
8	1	11	11
9	1	15	15
10	1	24	24
11	1	10	10
12	1	11	11
13	1	25	25
14	1	35	35
15	1	12	12
16	1	15	15
17	1	25	25
18	1	15	15
19	1	10	10
20	1	11	11
21	1	25	25
22	1	25	25
23	1	14	14

This results in a maximum move count of 35 and a weighted average move count of $\frac{427}{24} \approx 17.791$.

Face Permutation

	2-cycles	Variant	Combos	Formula	3-cycles	Product
1	0	0	1	${}_{10}C_{10}$	0	0
2	2	2,2	630	${}_{10}C_4 * {}_4C_2 * 1! * {}_2C_2 * 1! / 2!$	2	1260
3	2	3	240	${}_{10}C_3 * {}_3C_3 * 2! / 1!$	1	240
4	4	2,2,2,2	4725	${}_{10}C_8 * {}_8C_2 * 1! * {}_6C_2 * 1! * {}_4C_2 * 1! * {}_2C_2 * 1! / 4!$	4	18900
5	4	2,2,3	25200	${}_{10}C_7 * {}_4C_2 * 1! * {}_2C_2 * 1! / 2! * {}_7C_3 * 2! / 1!$	3	75600
6	4	2,4	18900	${}_{10}C_6 * {}_6C_2 * 1! / 1! * {}_4C_4 * 3! / 1!$	3	56700
7	4	3,3	8400	${}_{10}C_6 * {}_6C_3 * 2! * {}_3C_3 * 2! / 2!$	2	16800
8	4	5	6048	${}_{10}C_5 * {}_5C_5 * 4! / 1!$	2	12096
9	6	2,2,3,3	25200	${}_{10}C_{10} * {}_{10}C_2 * 1! * {}_8C_2 * 1! / 2! * {}_6C_3 * 2! * {}_3C_3 * 2! / 2!$	4	100800
10	6	2,2,5	90720	${}_{10}C_9 * {}_9C_2 * 1! * {}_7C_2 * 1! / 2! * {}_5C_5 * 4! / 1!$	4	362880
11	6	2,3,4	151200	${}_{10}C_9 * {}_9C_2 * 1! / 1! * {}_7C_3 * 2! / 1! * {}_4C_4 * 3! / 1!$	4	604800
12	6	2,6	151200	${}_{10}C_8 * {}_8C_2 * 1! / 1! * {}_6C_6 * 5! / 1!$	4	604800
13	6	3,3,3	22400	${}_{10}C_9 * {}_9C_3 * 2! * {}_6C_3 * 2! * {}_3C_3 * 2! / 3!$	3	67200
14	6	3,5	120960	${}_{10}C_8 * {}_8C_3 * 2! / 1! * {}_5C_5 * 4! / 1!$	3	362880
15	6	4,4	56700	${}_{10}C_8 * {}_8C_4 * 3! * {}_4C_4 * 3! / 2!$	4	226800
16	6	7	86400	${}_{10}C_7 * {}_7C_7 * 6! / 1!$	3	259200
17	8	2,8	226800	${}_{10}C_{10} * {}_{10}C_2 * 1! / 1! * {}_8C_8 * 7! / 1!$	5	1134000
18	8	3,7	172800	${}_{10}C_{10} * {}_{10}C_3 * 2! / 1! * {}_7C_7 * 6! / 1!$	4	691200
19	8	4,6	151200	${}_{10}C_{10} * {}_{10}C_4 * 3! / 1! * {}_6C_6 * 5! / 1!$	5	756000
20	8	5,5	72575	${}_{10}C_{10} * {}_{10}C_5 * 4! * {}_5C_5 * 4! / 2! - 1$	4	290304

21	8	9	403200	${}_{10}C_9 * {}_{9}C_9 * 8! / 1!$	4	1612800
22	?	?	18901	10! / 2 - ∑(1 to 21)	?	?

Assuming a maximum of 21 moves and an average of 16.75 moves for a face 3-cycle, then this results in a maximum move count of 105 and a weighted average move count of $\frac{121525605}{1795499} \approx 67.68$.

Qualitative Traits

This section is intended to address non-quantifiable aspects that impact the difficulty to solve the puzzle.

Blockage: Jumbling

This puzzle experience blockage due to its jumbling nature. This impacts the design of algorithms

Conclusion

The following is the difficulty score card for this puzzle:

Figure 46 Difficulty Scorecard

Combinations Magnitude	21.1
Algorithm Characteristics	4
Memory Units	52
Expected Move Count	178*
Maximum Move Count	280*
Qualitative Characteristics	Blockage: Jumbling

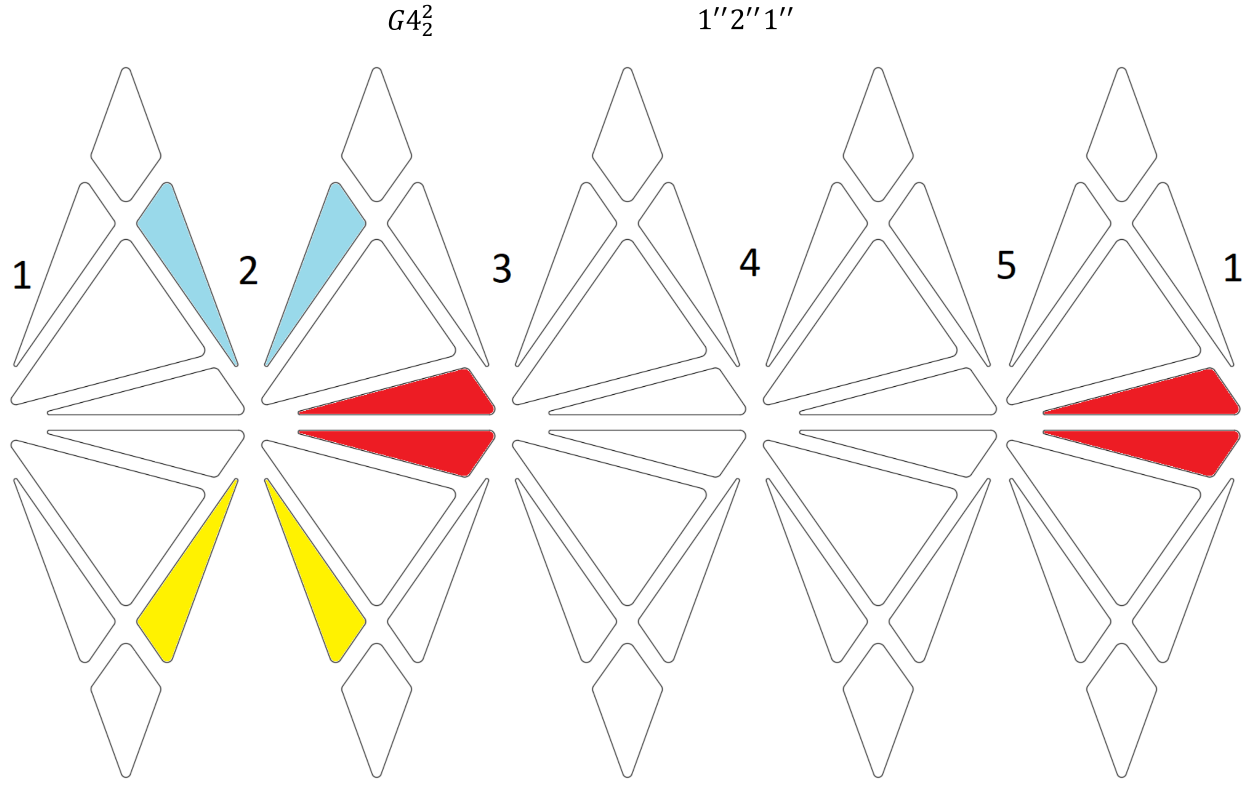
Appendix

Appendix 1: Edge Permutations - Graphical

The Blue edge is the target location (“hole”). The Yellow edge is the piece which should be relocated to the target location. The Red edges are the ancillary edges that get relocated in the process. The Green edges are previously solved edges that should remain solved at the end of each piece of incremental progress.

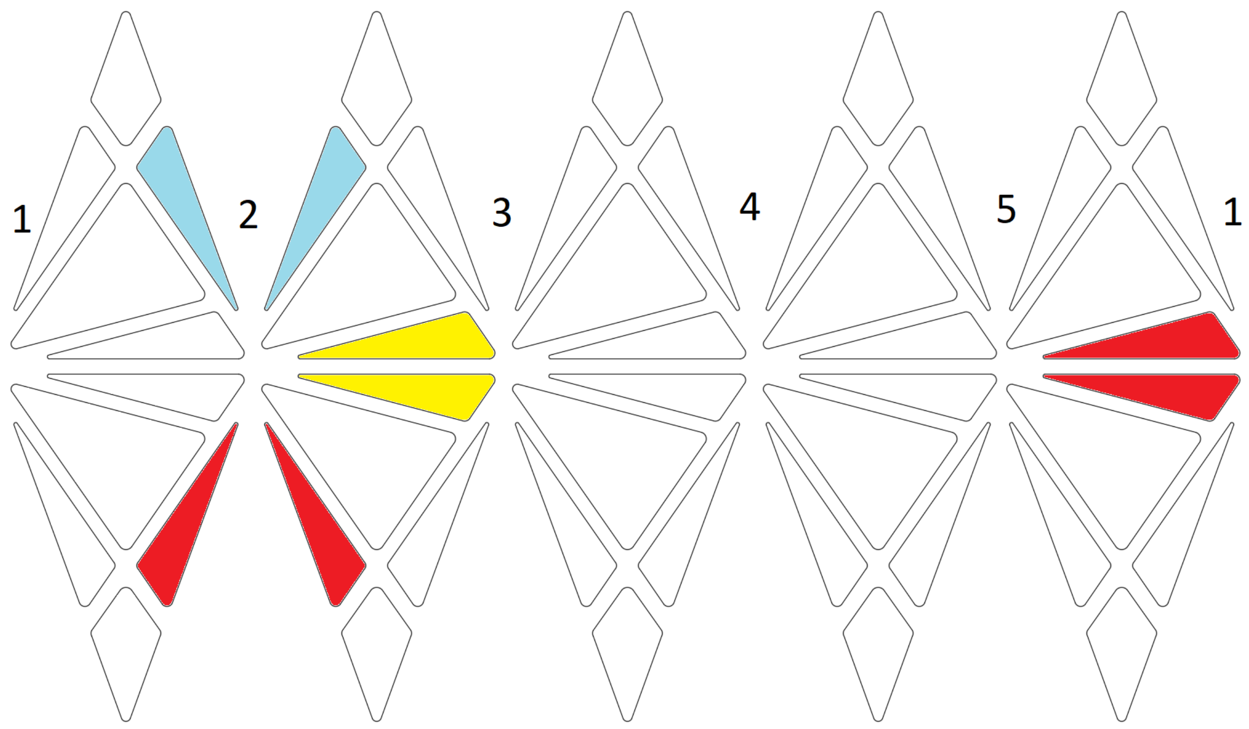
Permuting the First Edge

Location 2



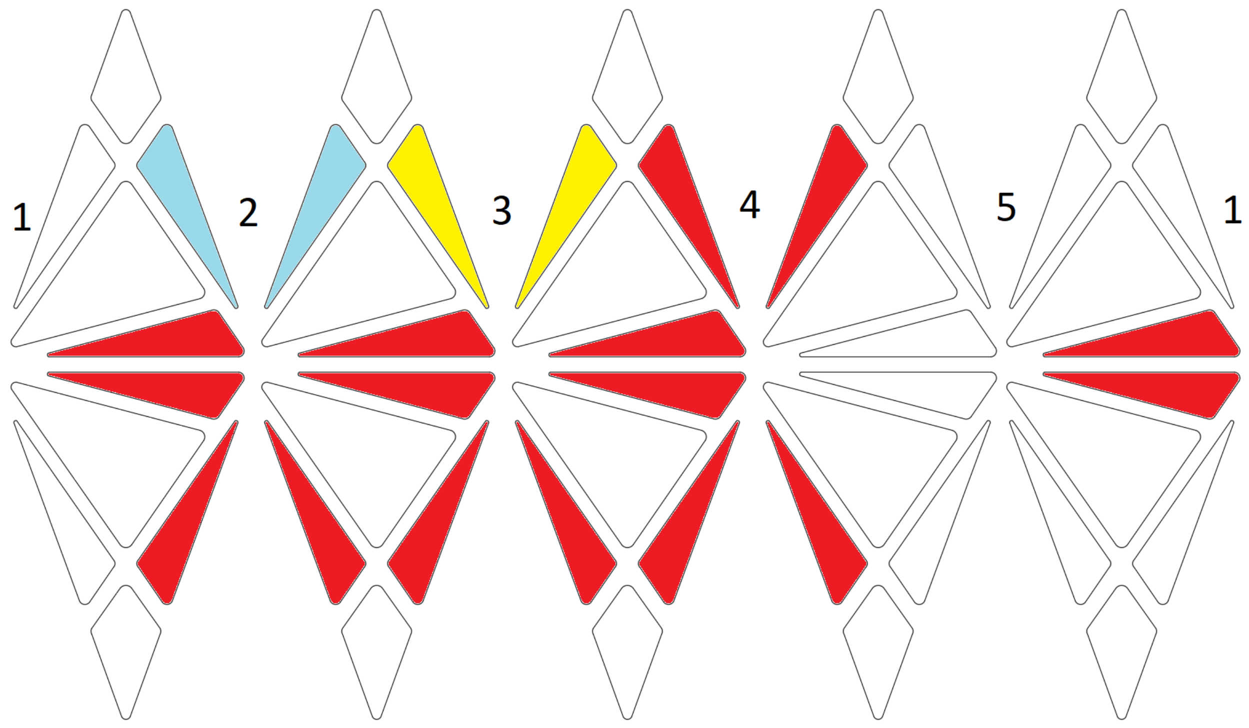
Location 3

$G4_2^{-1}$ $1''2'1''$



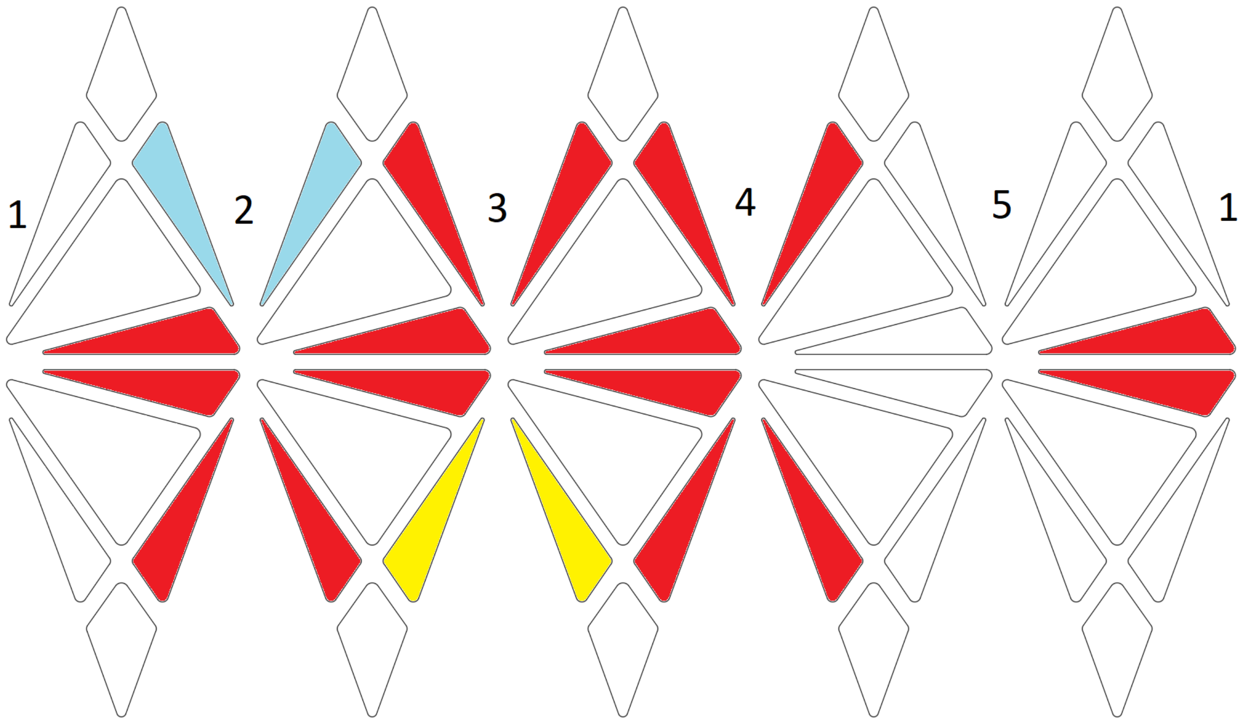
Location 4

$G4_3^1 G2_4^5 G4_2^1$ $2''32''5''4''5''1''21''$



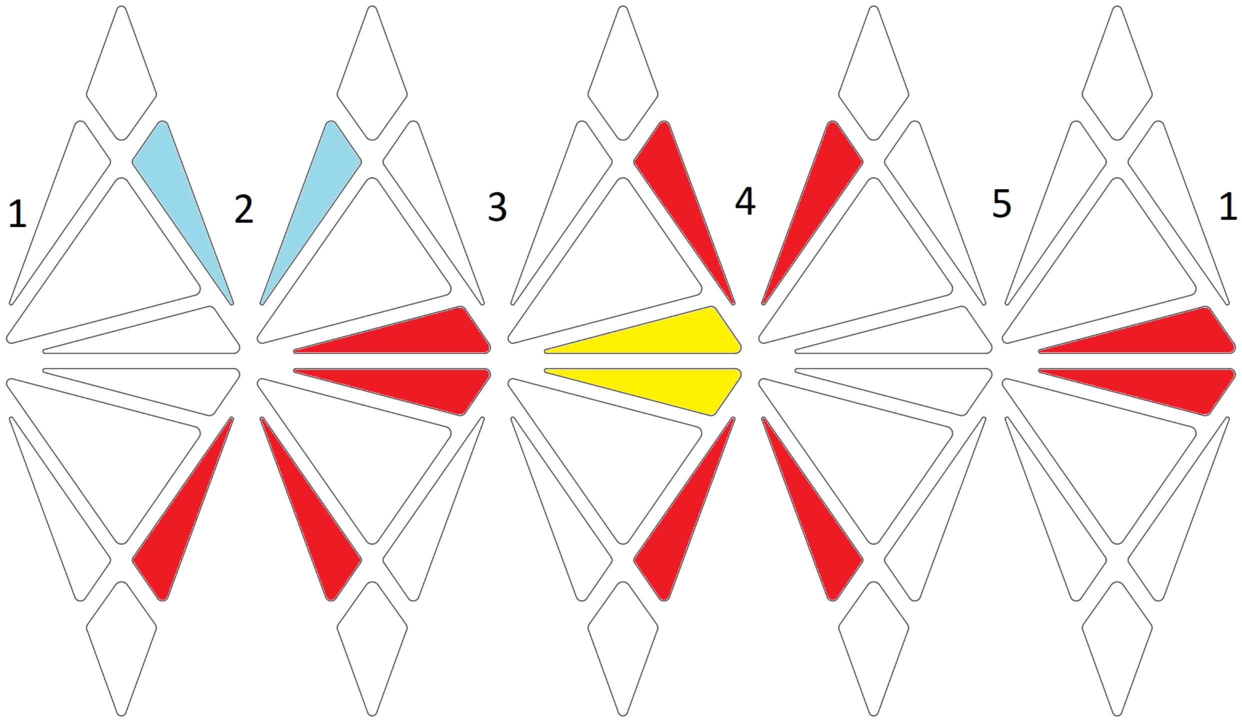
Location 5

$G4_3^{-1}G2_4^5G4_2^1$ $\quad$ $2''3'2''5''4''5''1''21''$



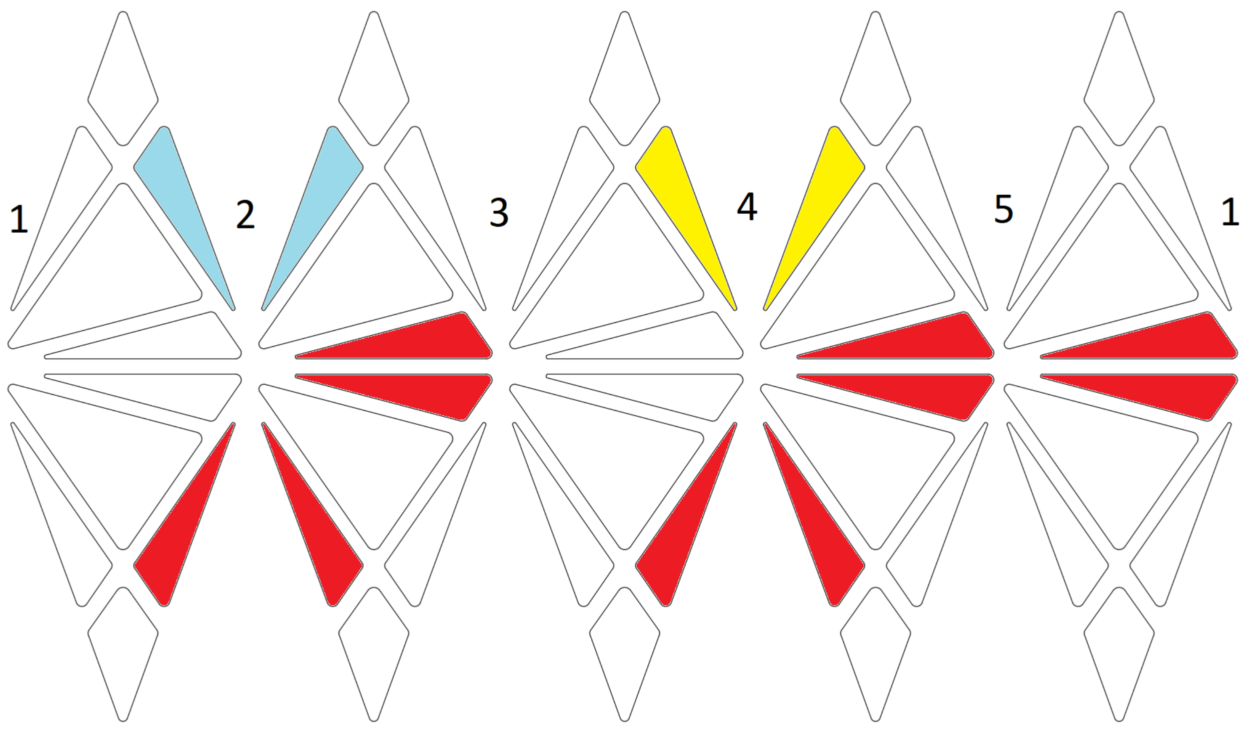
Location 6

$G2_4^5G4_2^1$ $\quad$ $5''4''5''1''21''$



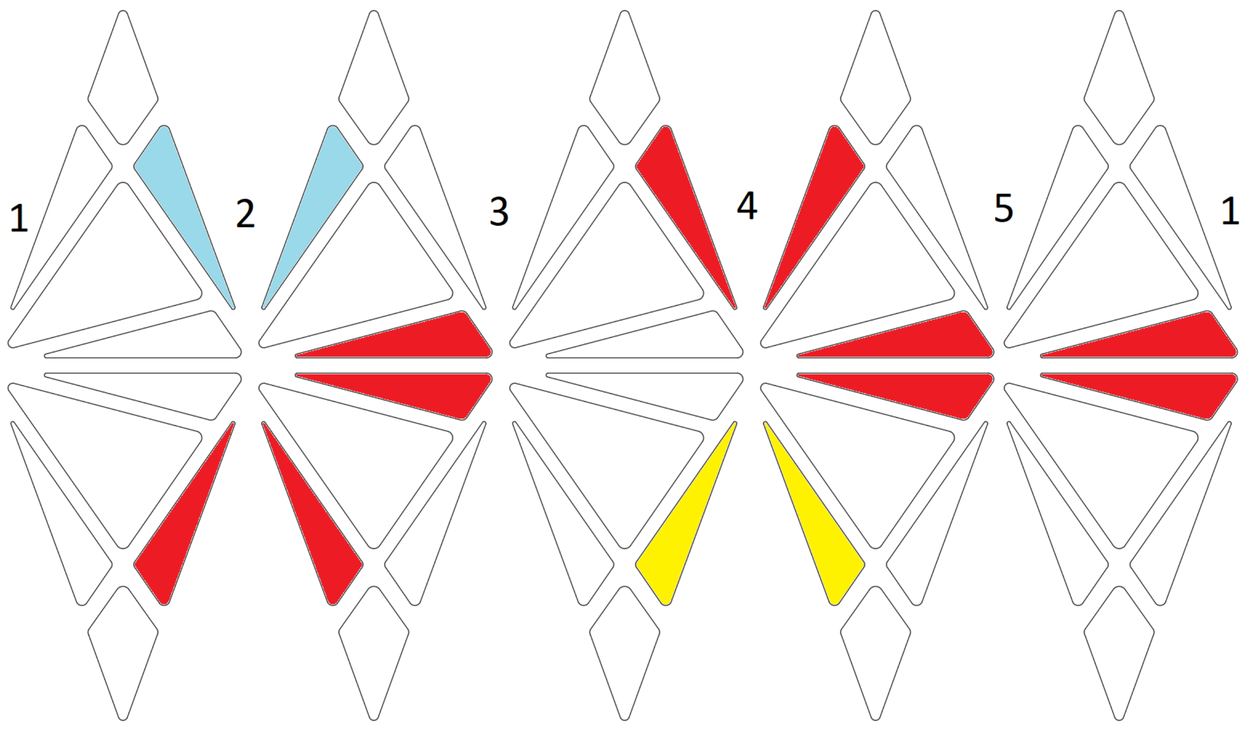
Location 7

$G4_4^{-1}G4_2^{-1}$ $3''4'3''1''2'1''$



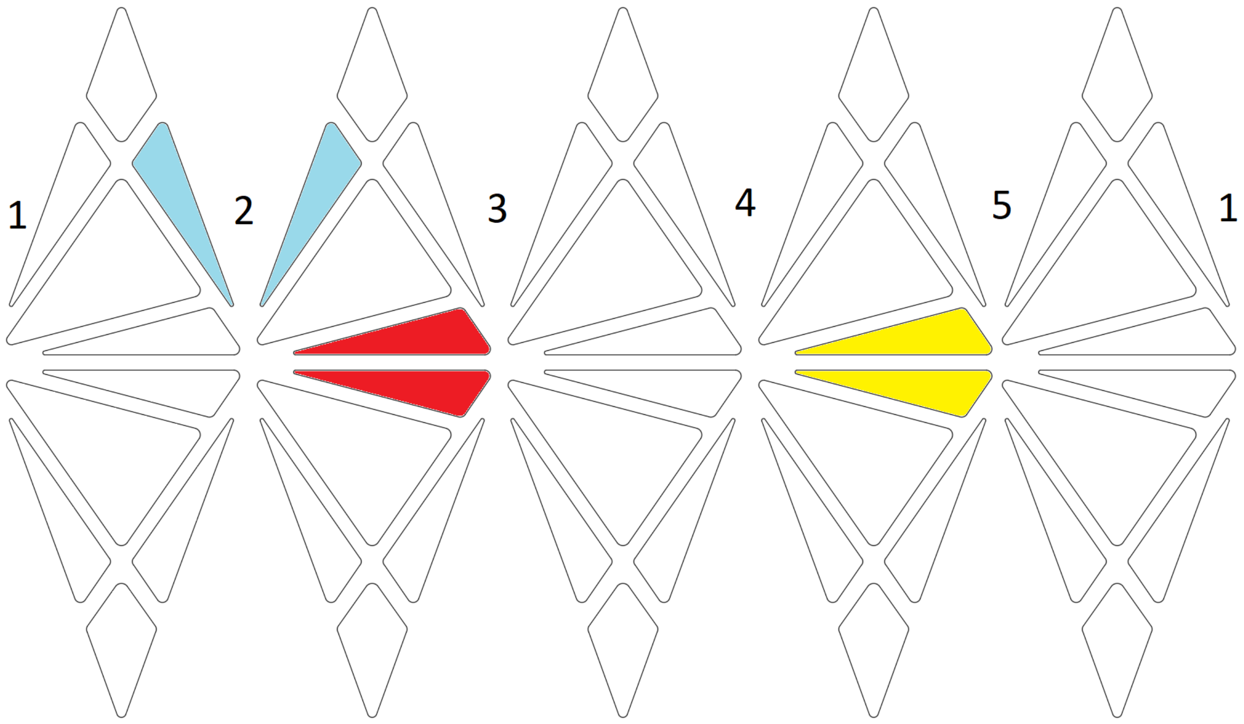
Location 8

$G4_4^{1}G4_2^{-1}$ $3''43''1''2'1''$



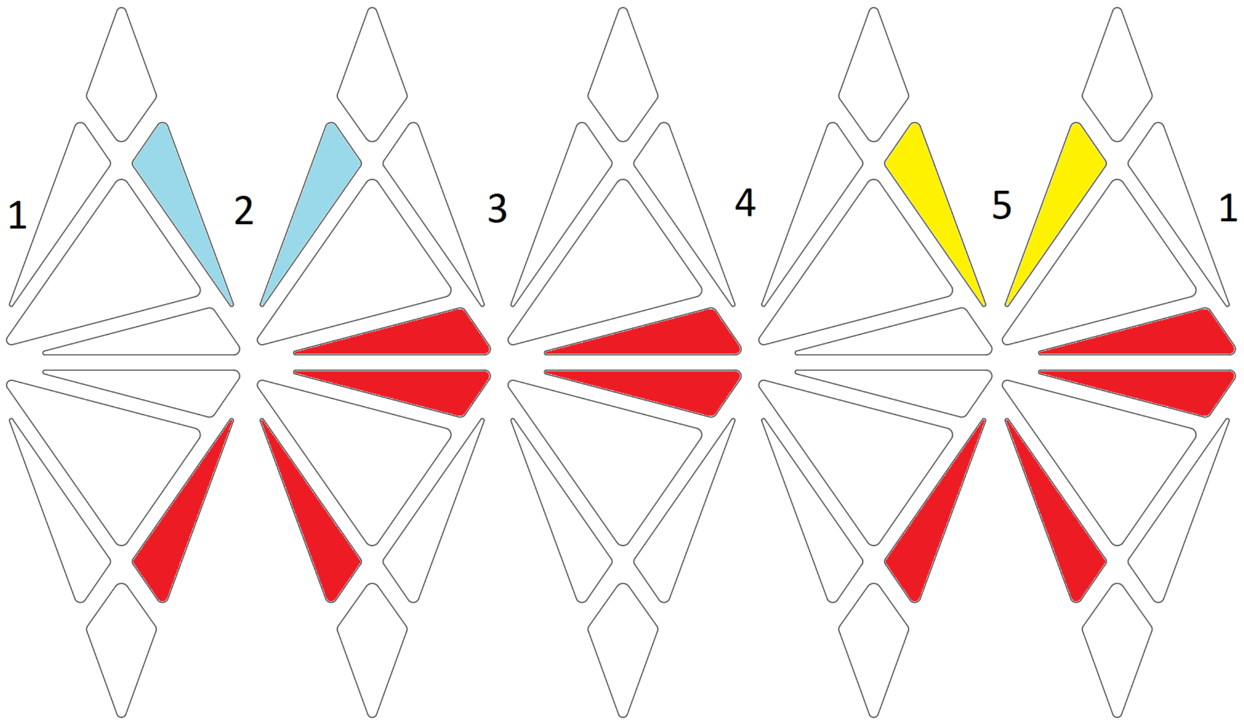
Location 9

$E3_2^{-1,-1}$ $1''4''23''2'3''4''1''$



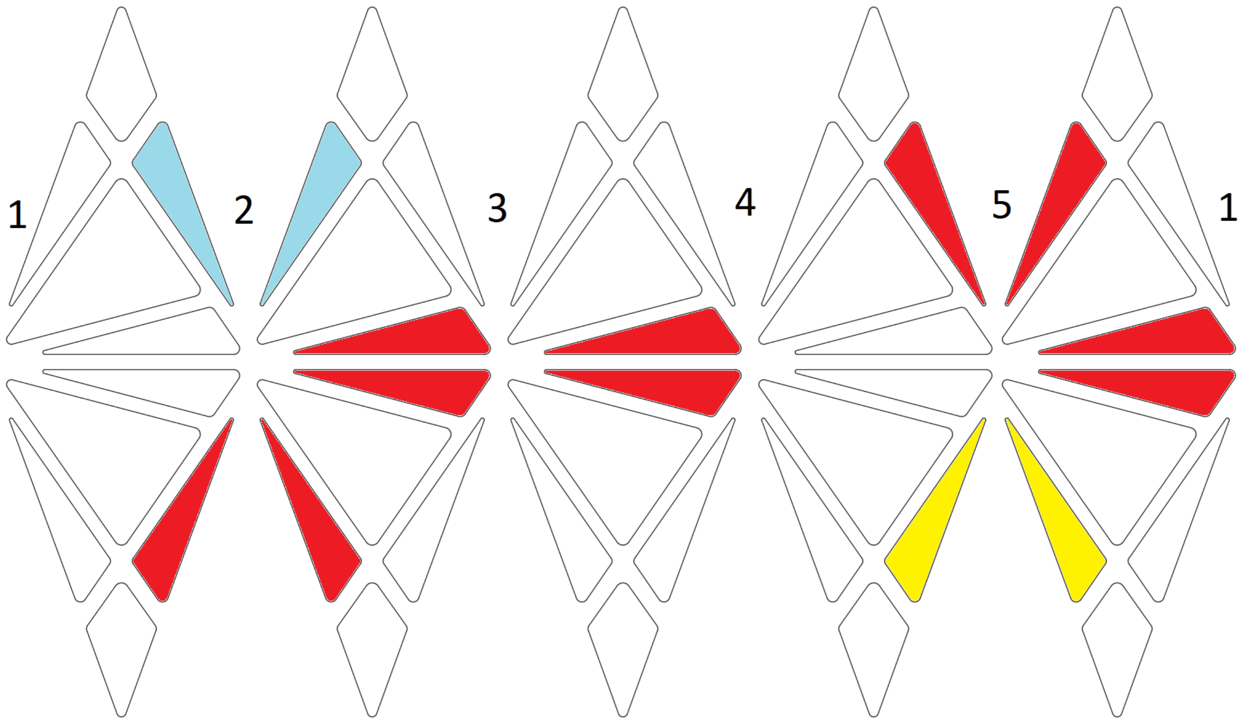
Location 10

$G4_5^1G4_2^1$ $4''54''1''21''$



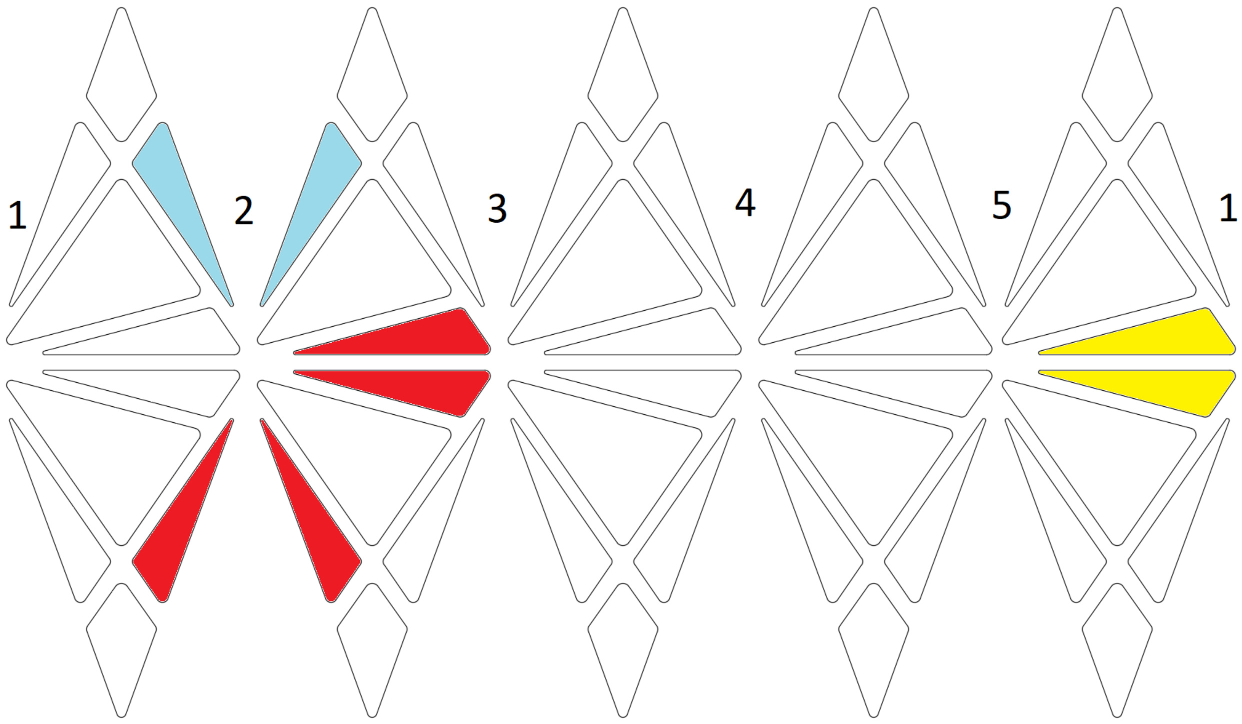
Location 11

$G4_5^{-1}G4_2^1$ $4''5'4''1''21''$



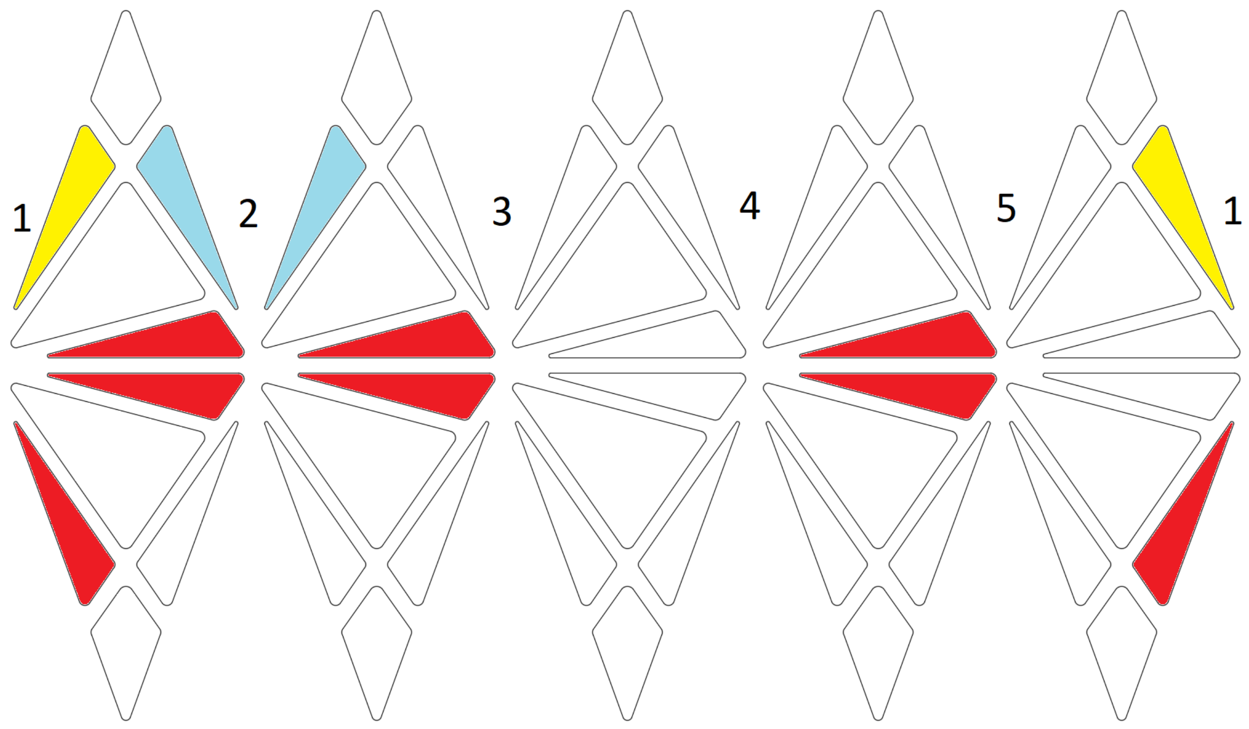
Location 12

$G4_2^1$ $1''21''$



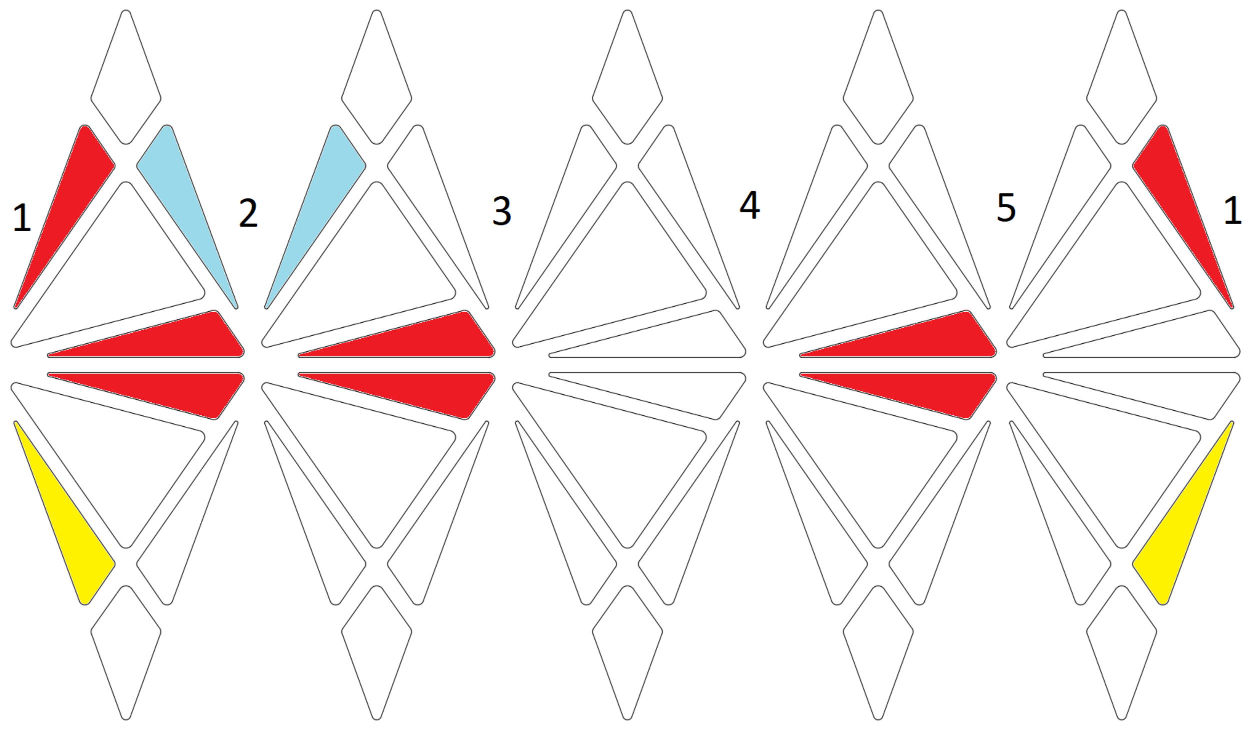
Location 13

$G4_1^{-1}E3_2^{-1,-1}$ 5′′1′5′′1′′4′′23′′2′3′′4′′1′′



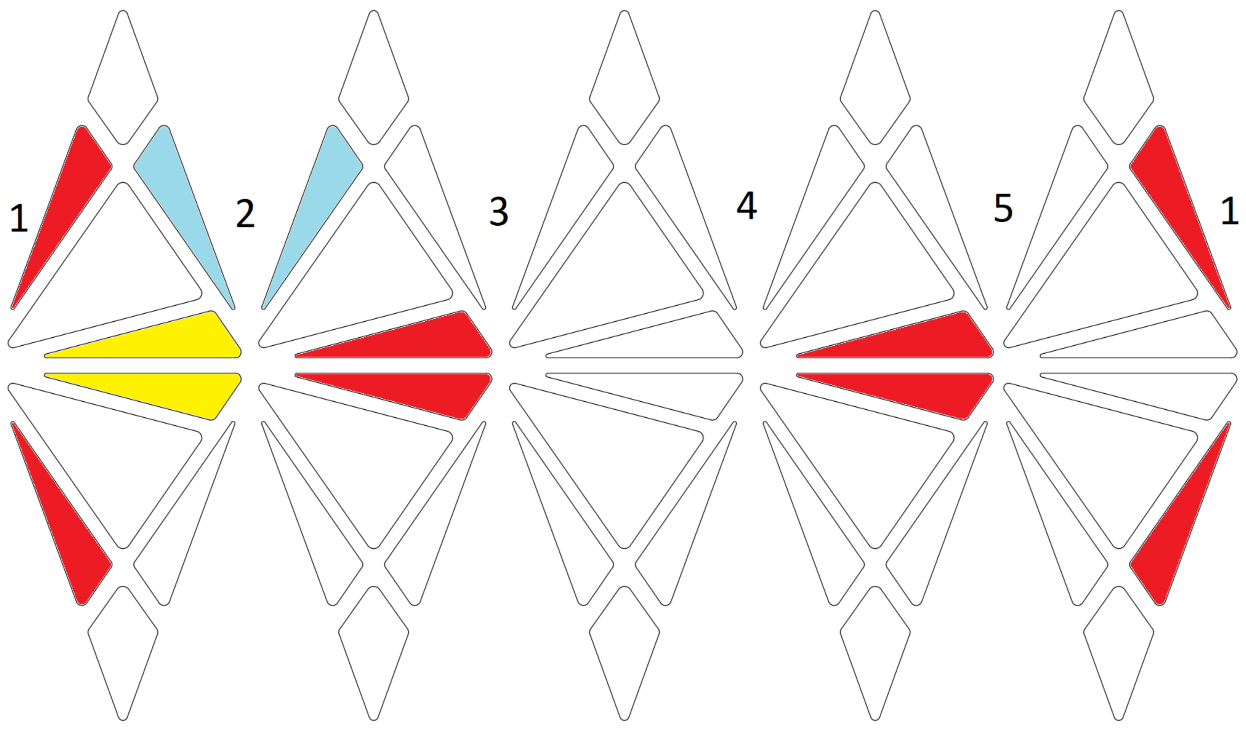
Location 14

$G4_1^{1}E3_2^{-1,-1}$ 5′′15′′1′′4′′23′′2′3′′4′′1′′



Location 15

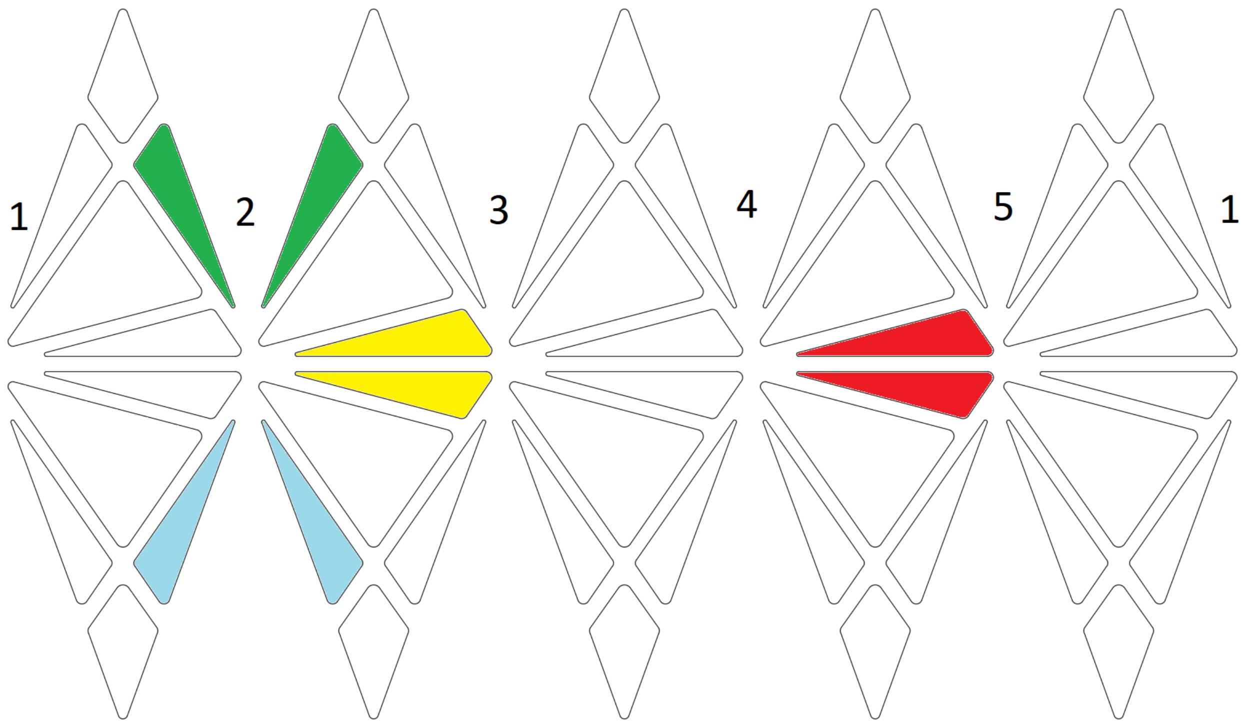
$G4_1^2E3_2^{-1,-1}$ $5''1''5''1''4''23''2'3''4''1''$



Permuting the Second Edge

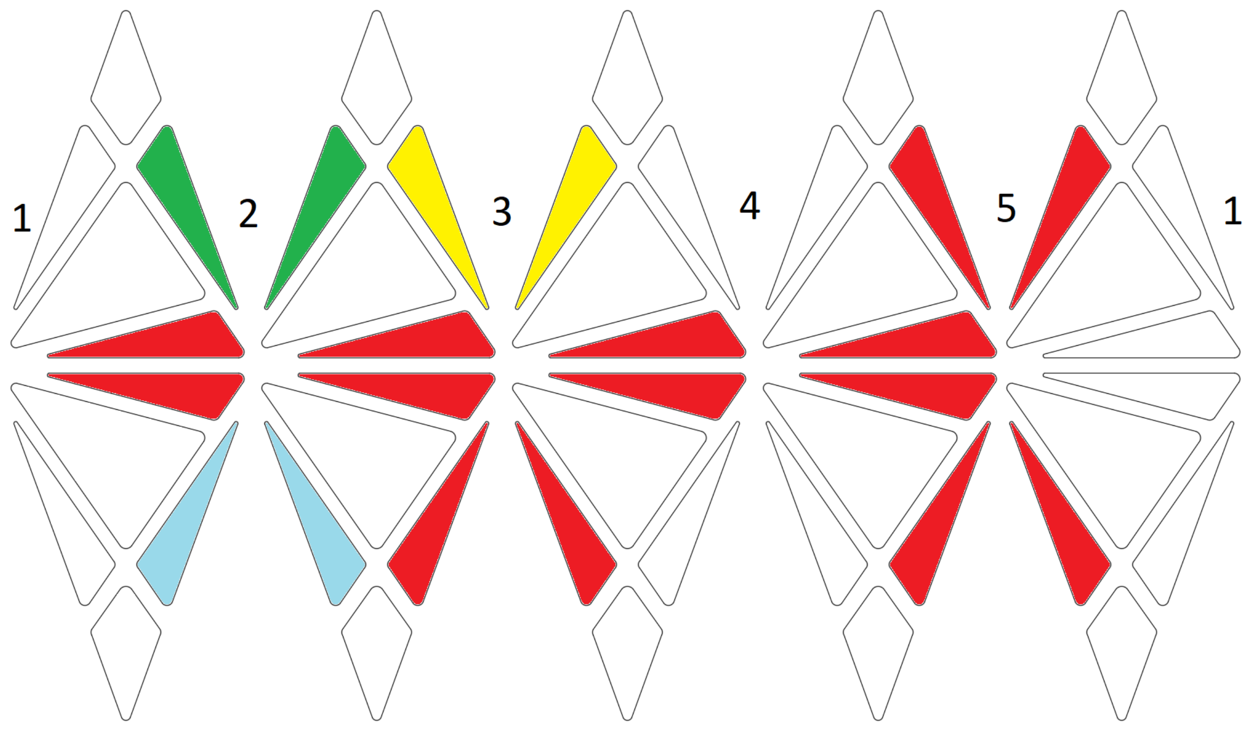
Location 3

$E3_2^{1,1}$ $1''4''3''23''2'4''1''$



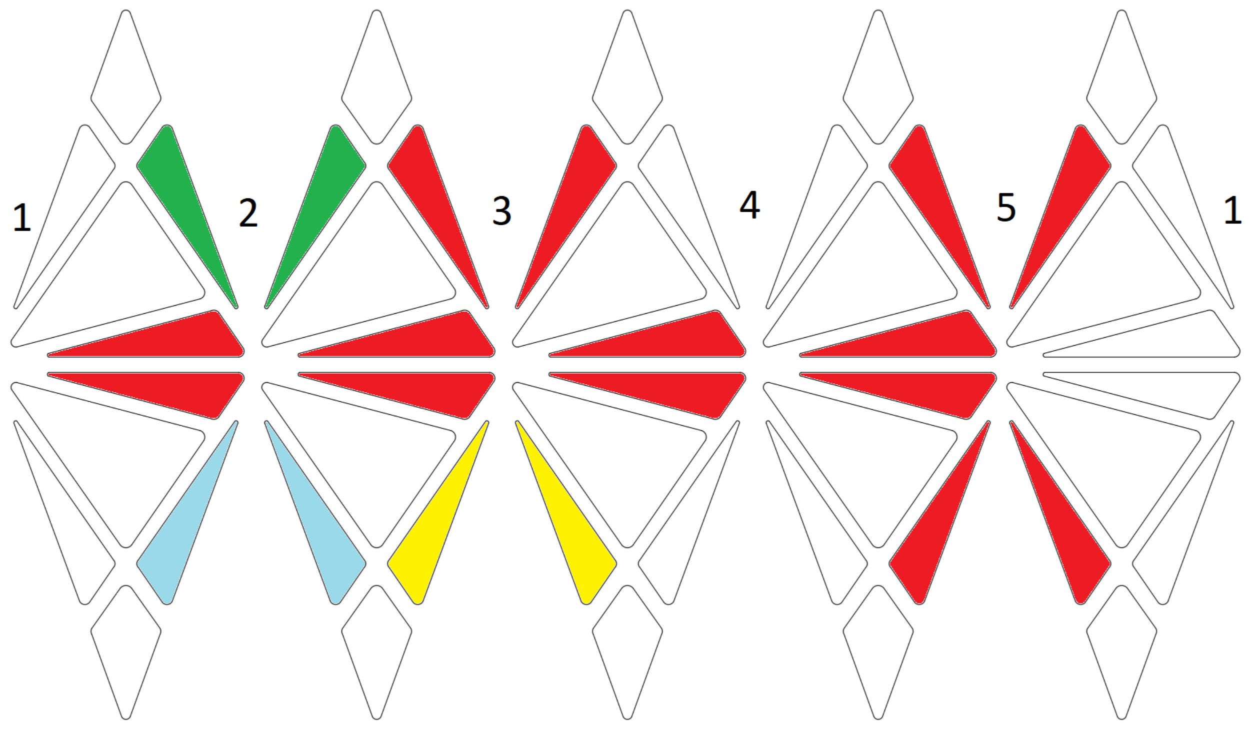
Location 4

$G4_3^{-1}G2_5^1E3_2^{1,-1}$ $2''3'2''1''5''4''2'3''23''4''1''$



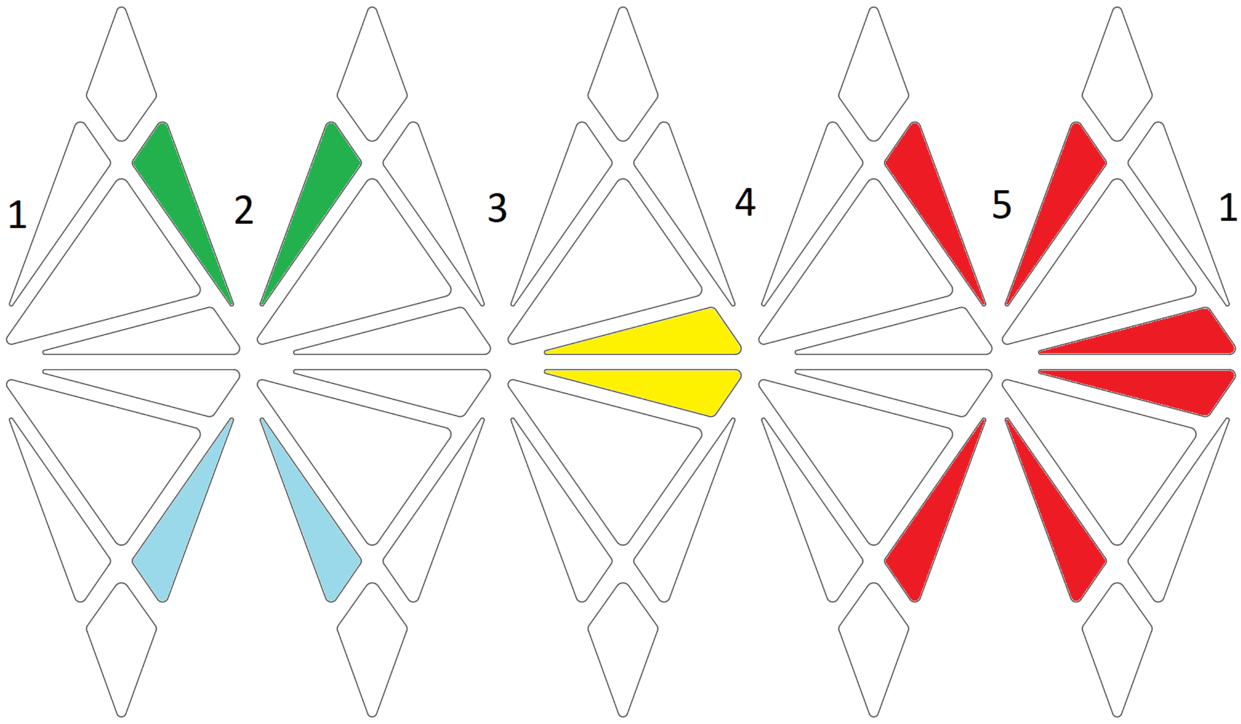
Location 5

$G4_3^{1}G2_5^1E3_2^{1,-1}$ $2''32''1''5''4''2'3''23''4''1''$



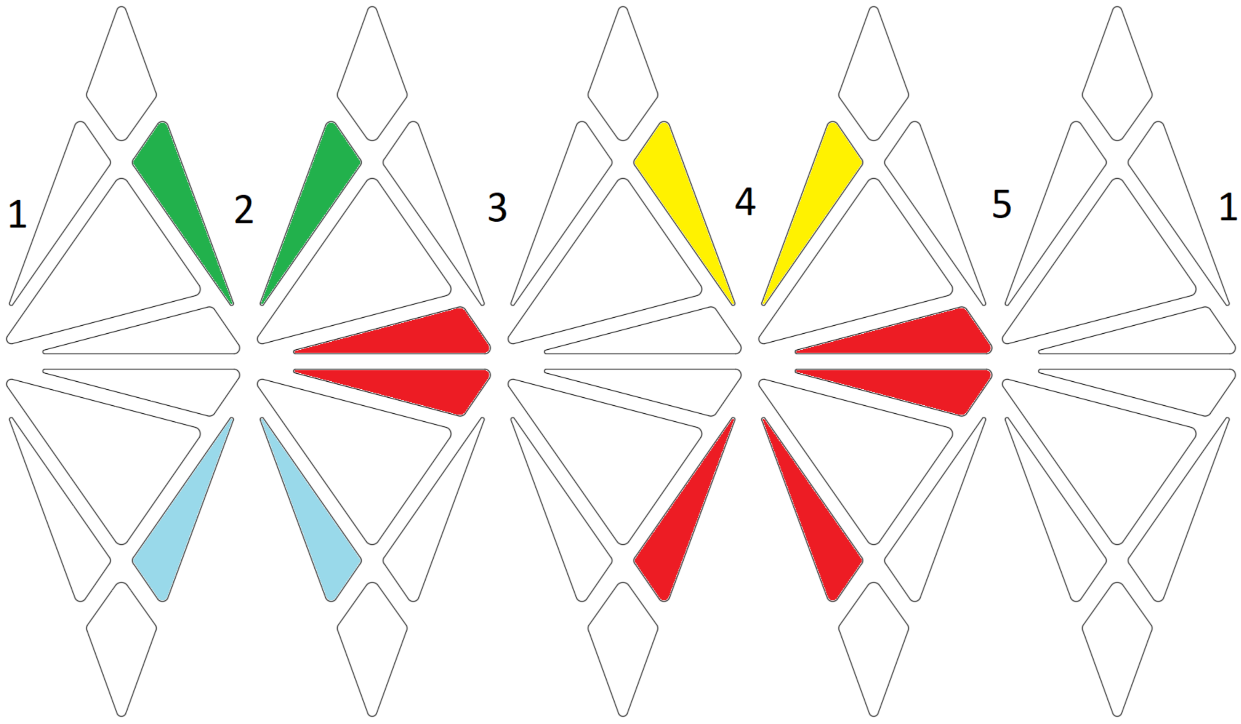
Location 6

$G4_2^1 G2_5^4 G4_2^{-1}$ $1''21''4''5''4''1''2'1''$



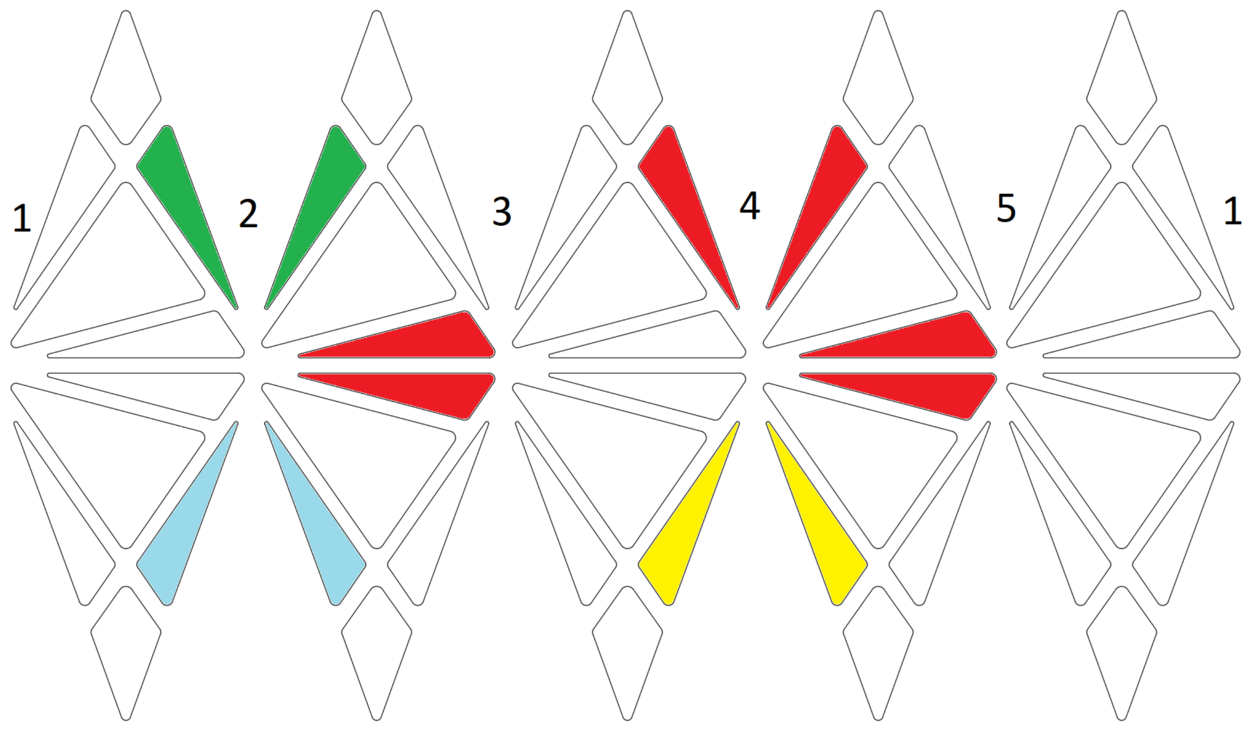
Location 7

$G4_4^1 E3_2^{1,-1}$ $3''43''1''4''2'3''23''4''1''$



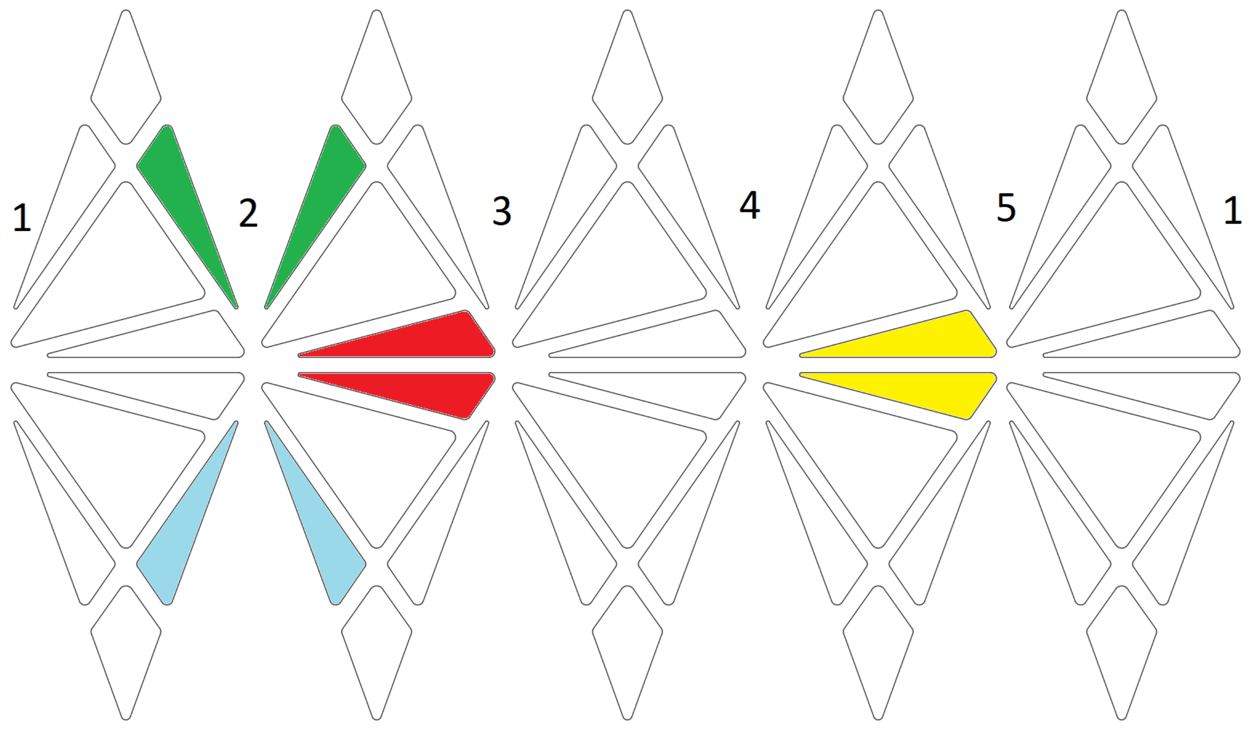
Location 8

$G4_4^{-1}E3_2^{1,-1}$ $3''4'3''1''4''2'3''23''4''1''$



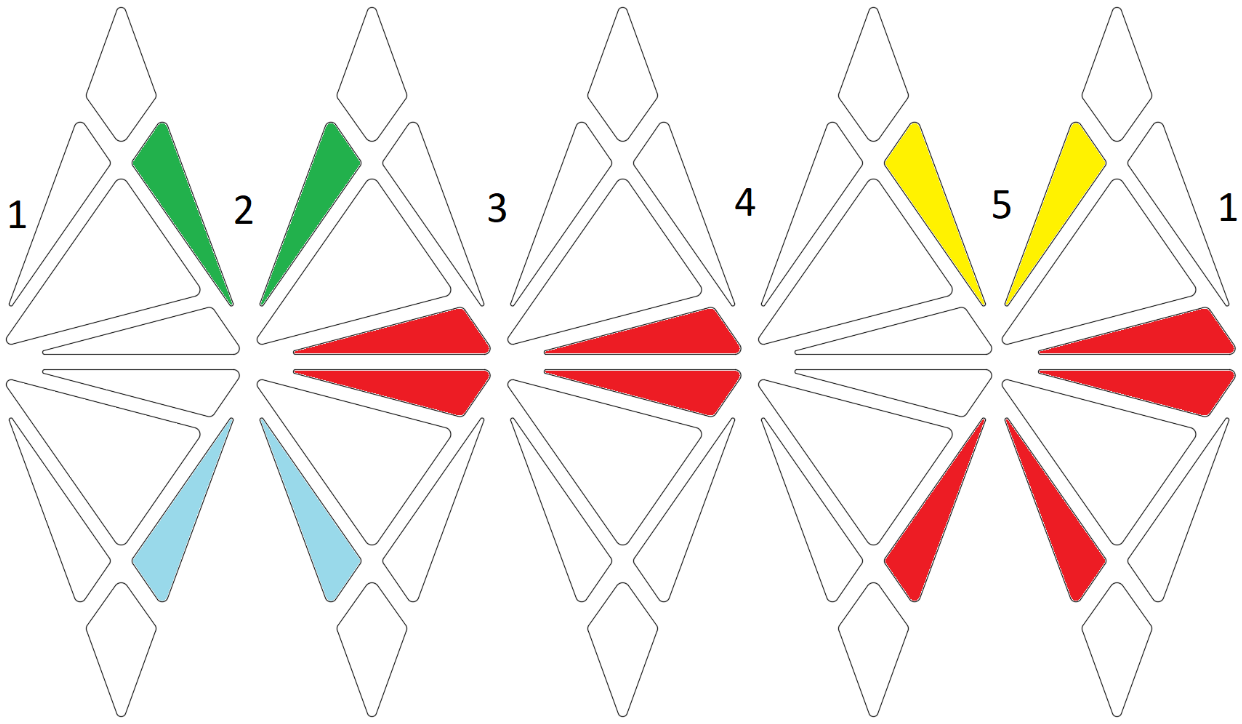
Location 9

$E3_2^{1,-1}$ $1''4''2'3''23''4''1''$



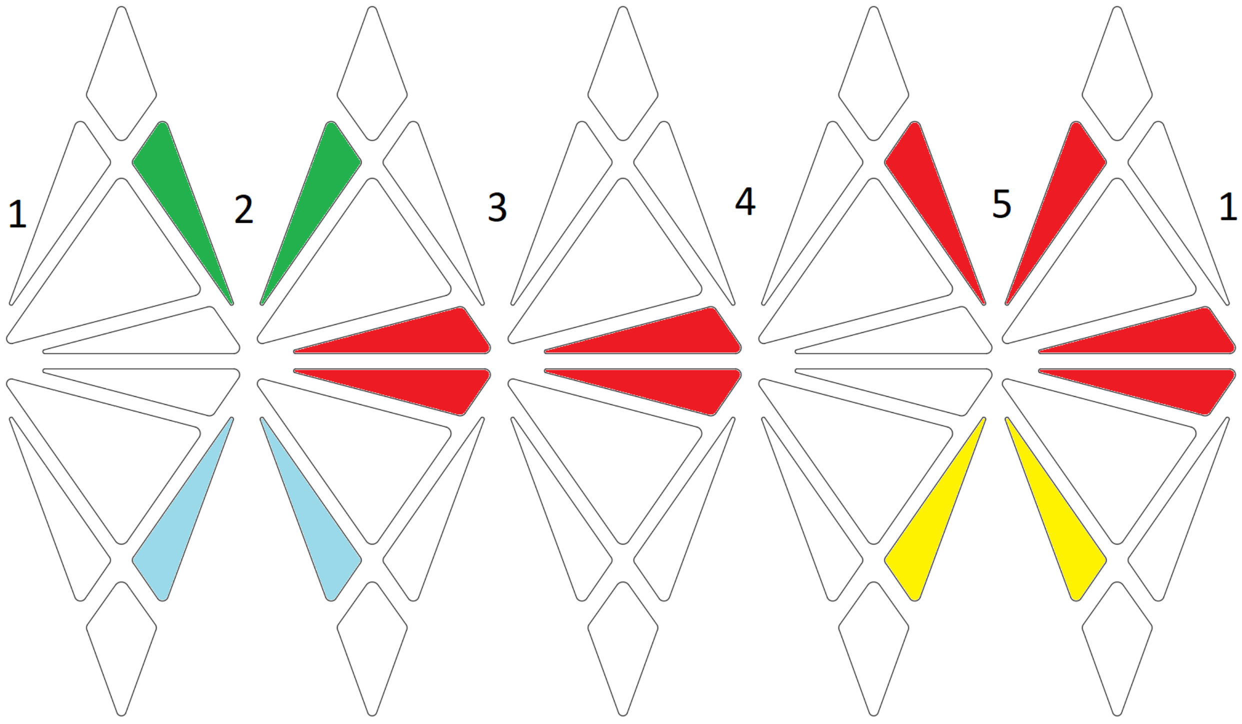
Location 10

$G4^{1}_{2}G4^{1}_{5}G4^{-1}_{2}$ $1''21''4''54''1''2'1''$



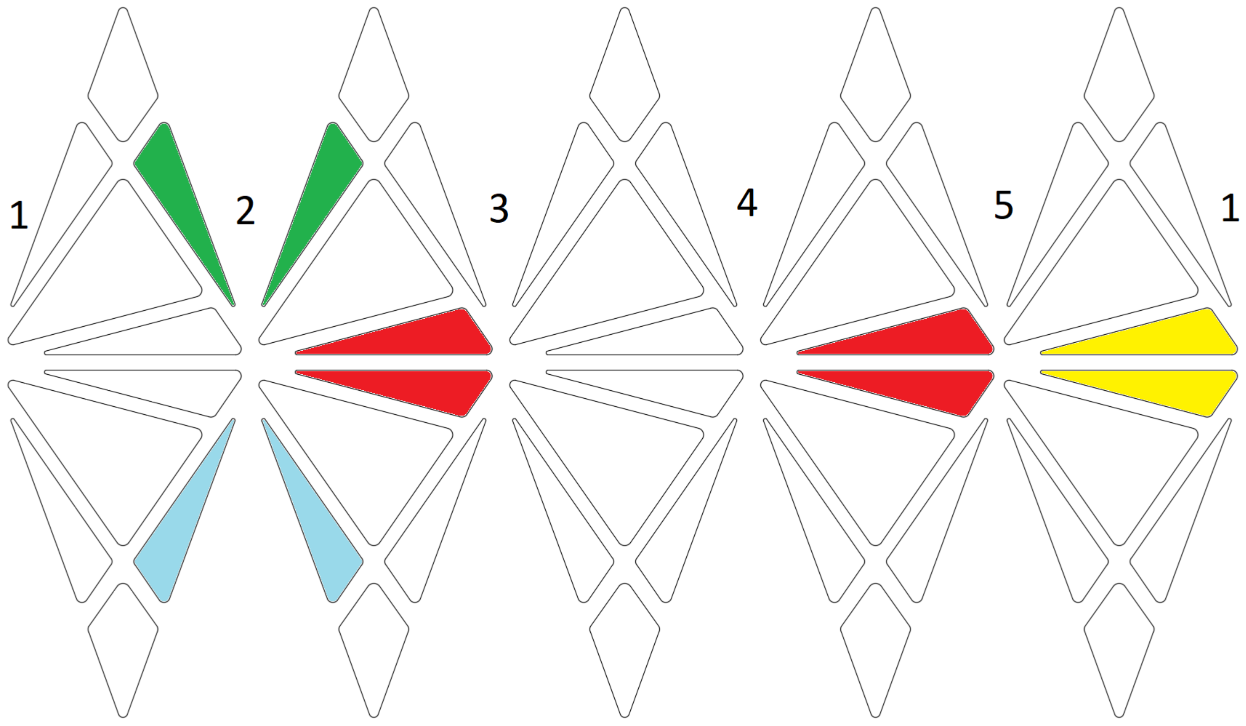
Location 11

$G4^{1}_{2}G4^{-1}_{5}G4^{-1}_{2}$ $1''21''4''5'4''1''2'1''$



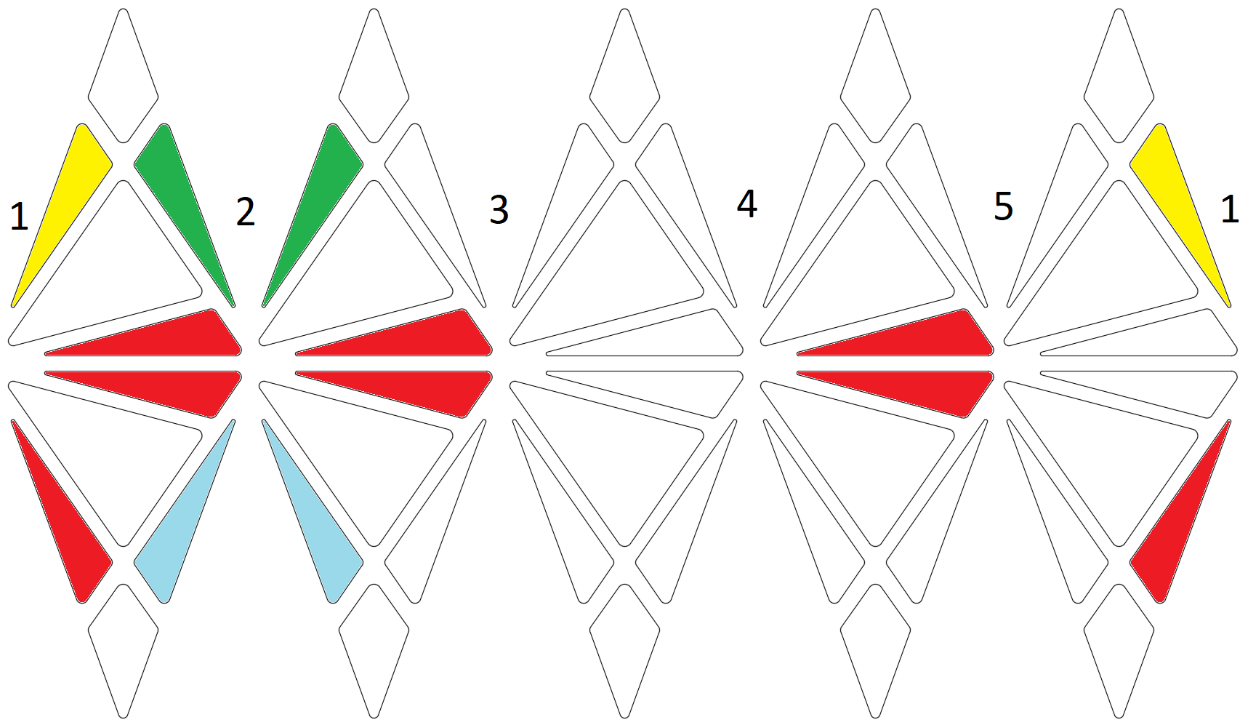
Location 12

$E3_2^{-1,-1}G4_2^{-1}$ $1''4''23''2'3''4''2'1''$



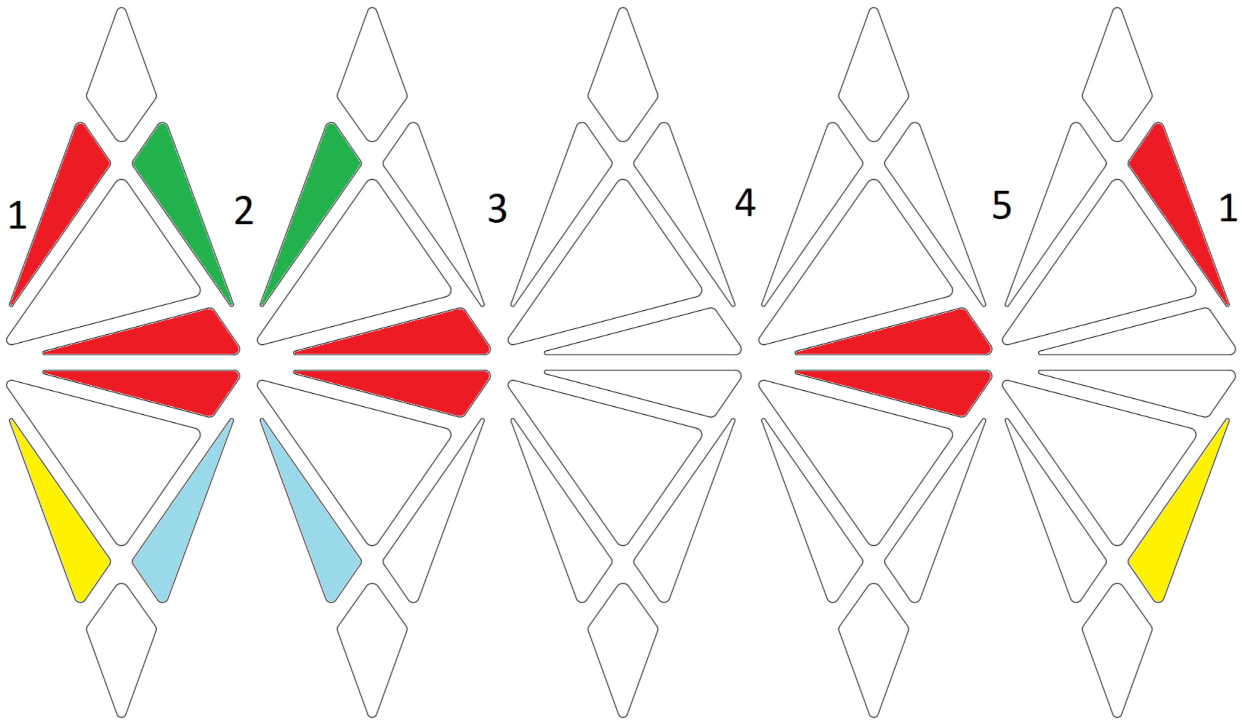
Location 13

$G4_1^{-1}E3_2^{1,-1}$ $5''1'5''1''4''2'3''23''4''1''$



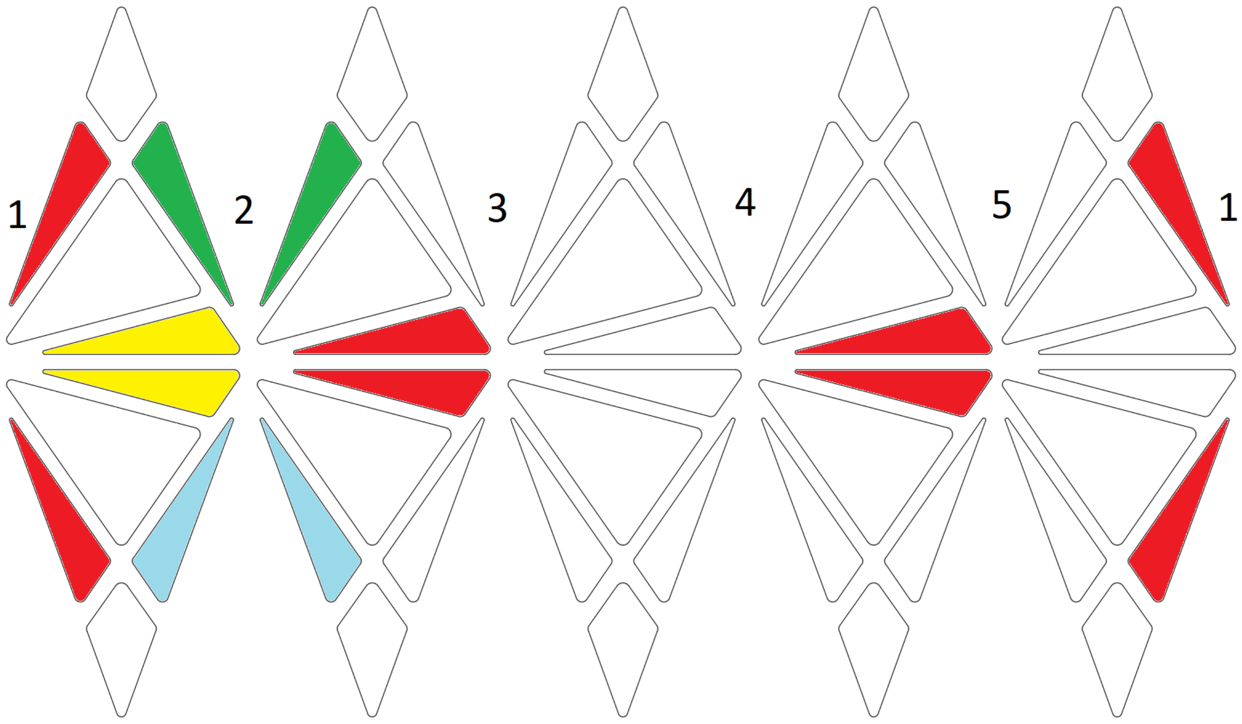
Location 14

$G4_1^1E3_2^{1,-1}$ 5″15″1″4″2′3″23″4″1″



Location 15

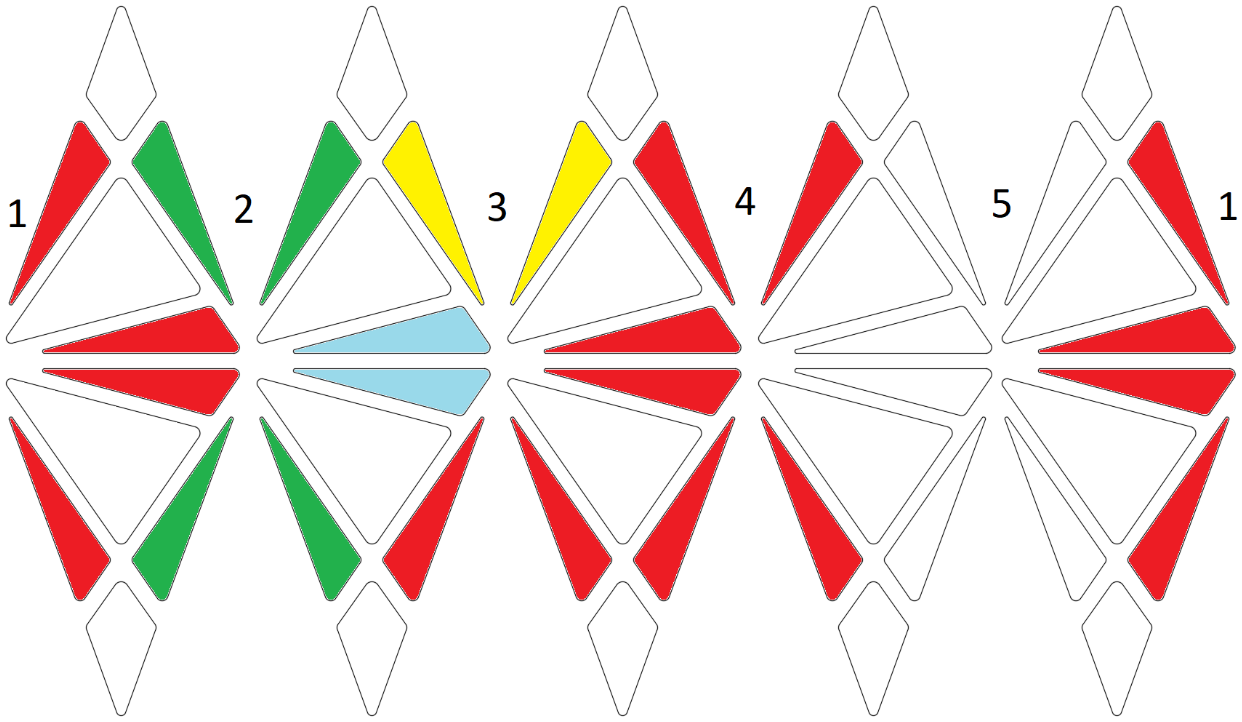
$G4_1^2E3_2^{1,-1}$ 5″1″5″1″4″2′3″23″4″1″



Permuting the Third Edge

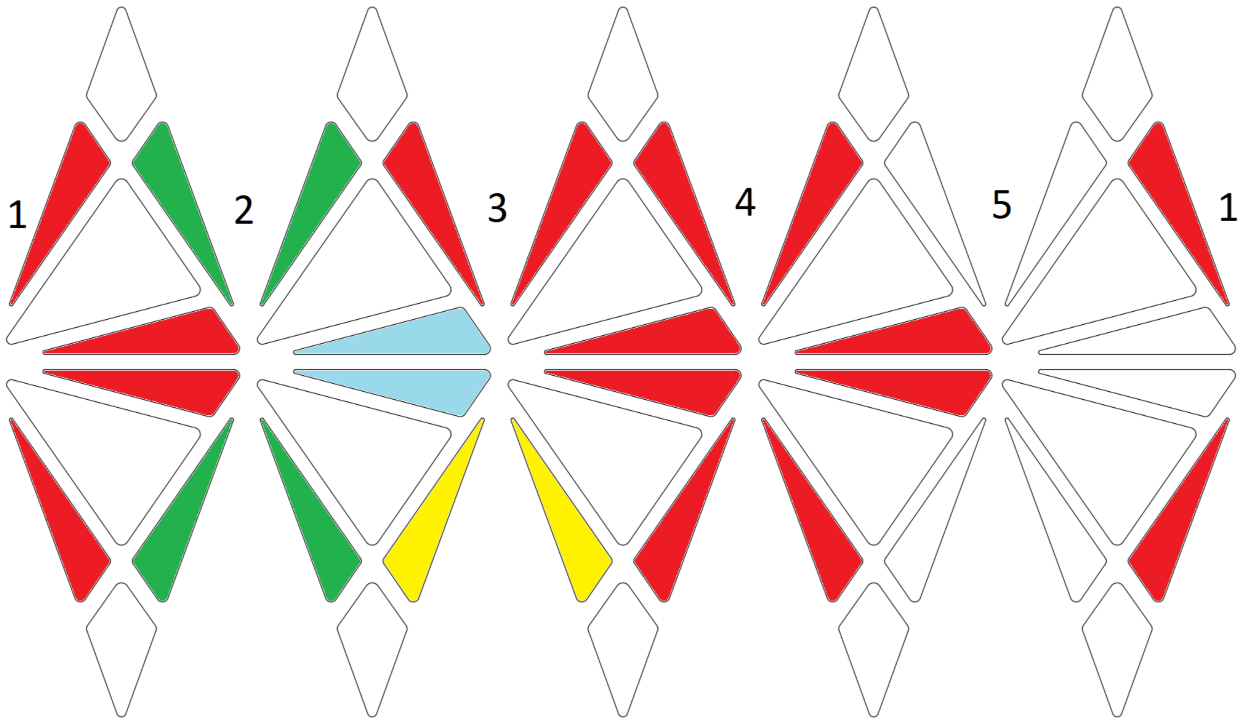
Location 4

$G4_3^1 G2_4^5 G2_1^2$ 2″32″5″4″5″2″1″2″



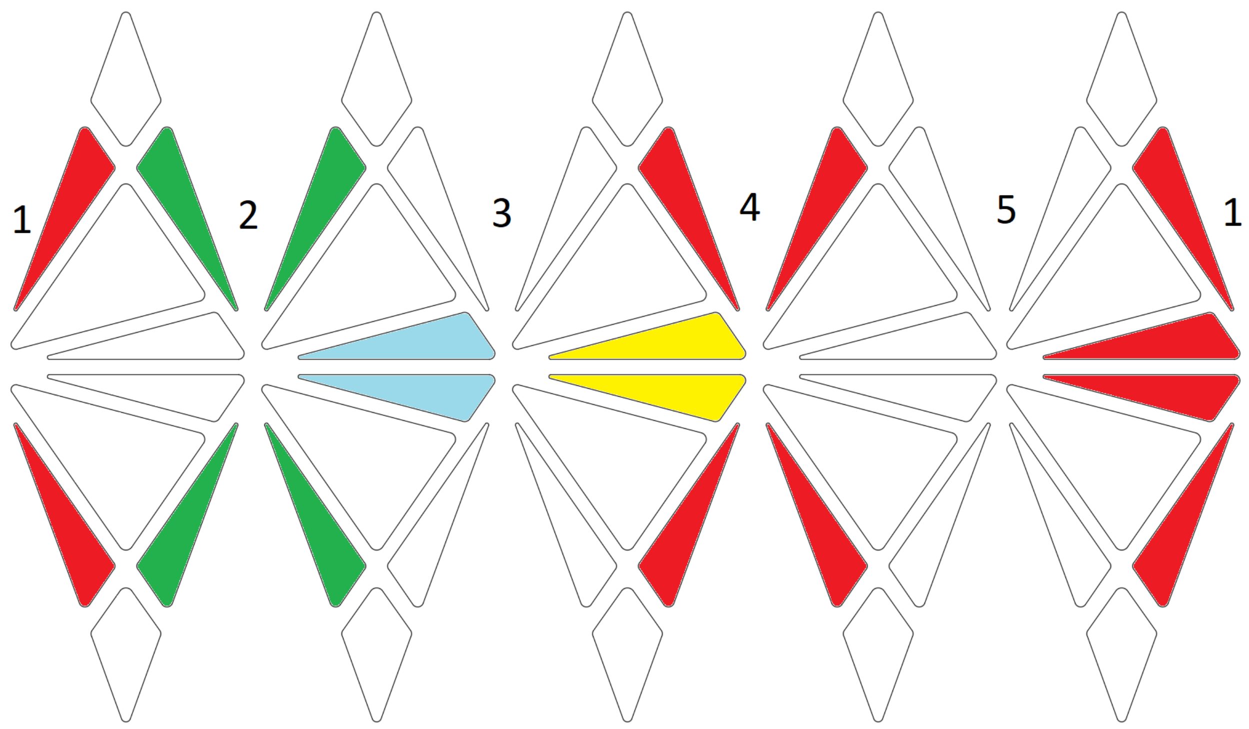
Location 5

$G4_3^{-1} G2_4^5 G2_1^2$ 2″3′2″5″4″5″2″1″2″



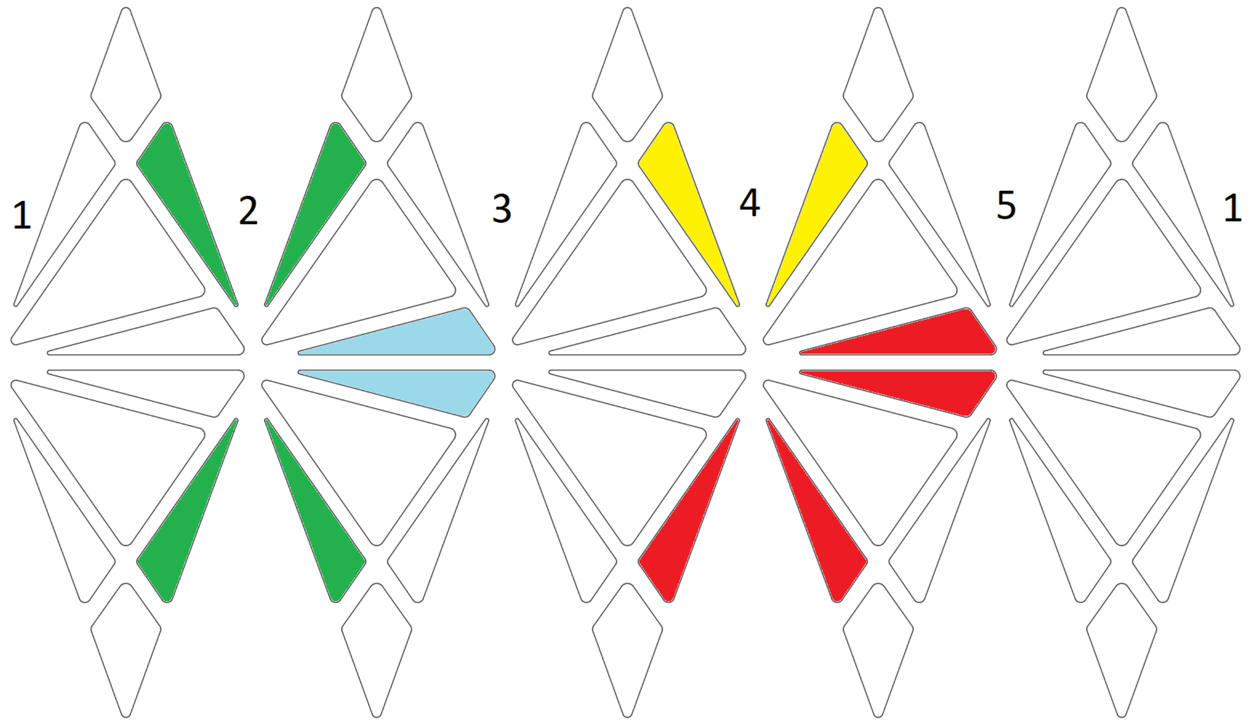
Location 6

$G2_4^5 G2_1^2$ $5''4''5''2''1''2''$



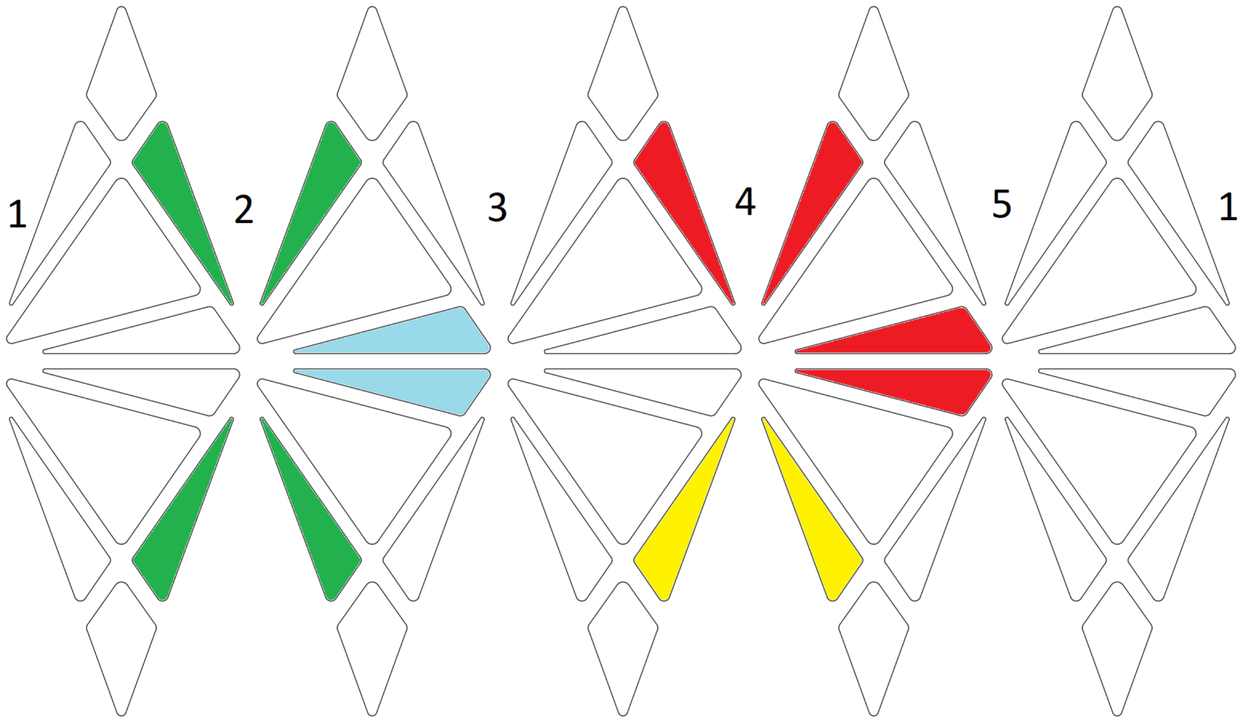
Location 7

$G4_4^{-1}$ $3''4'3''$



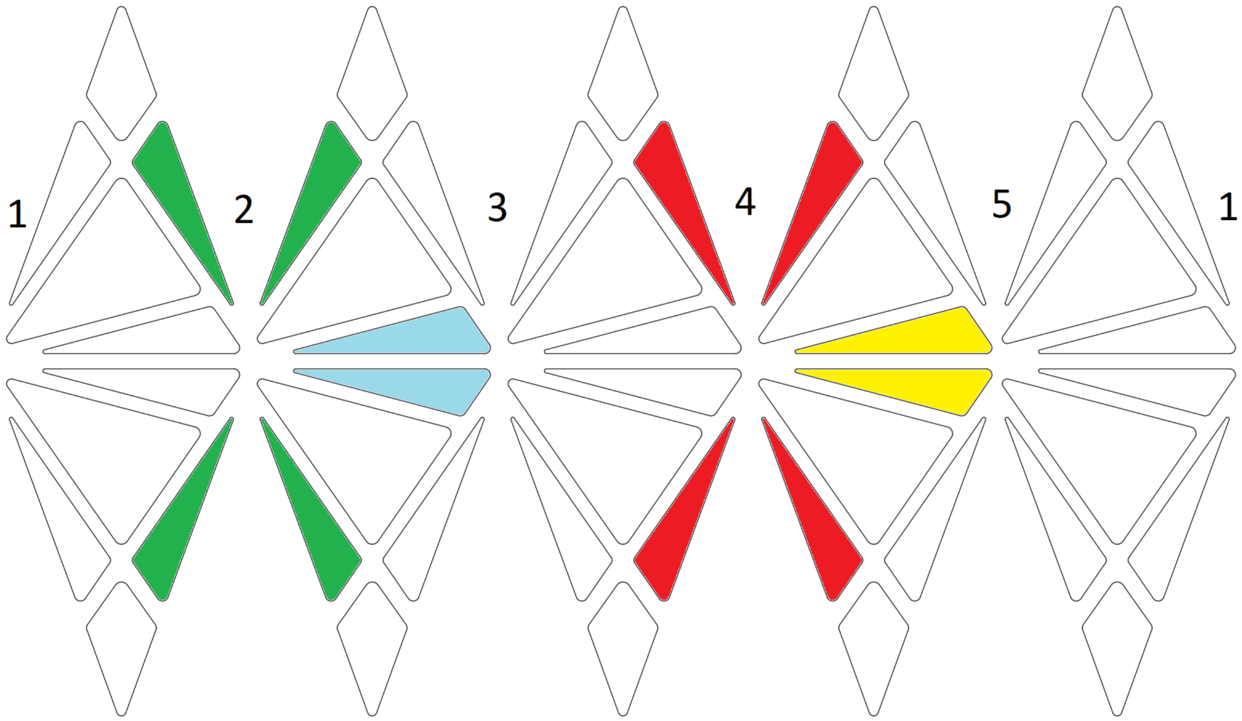
Location 8

$G4_4^1$ $3''43''$



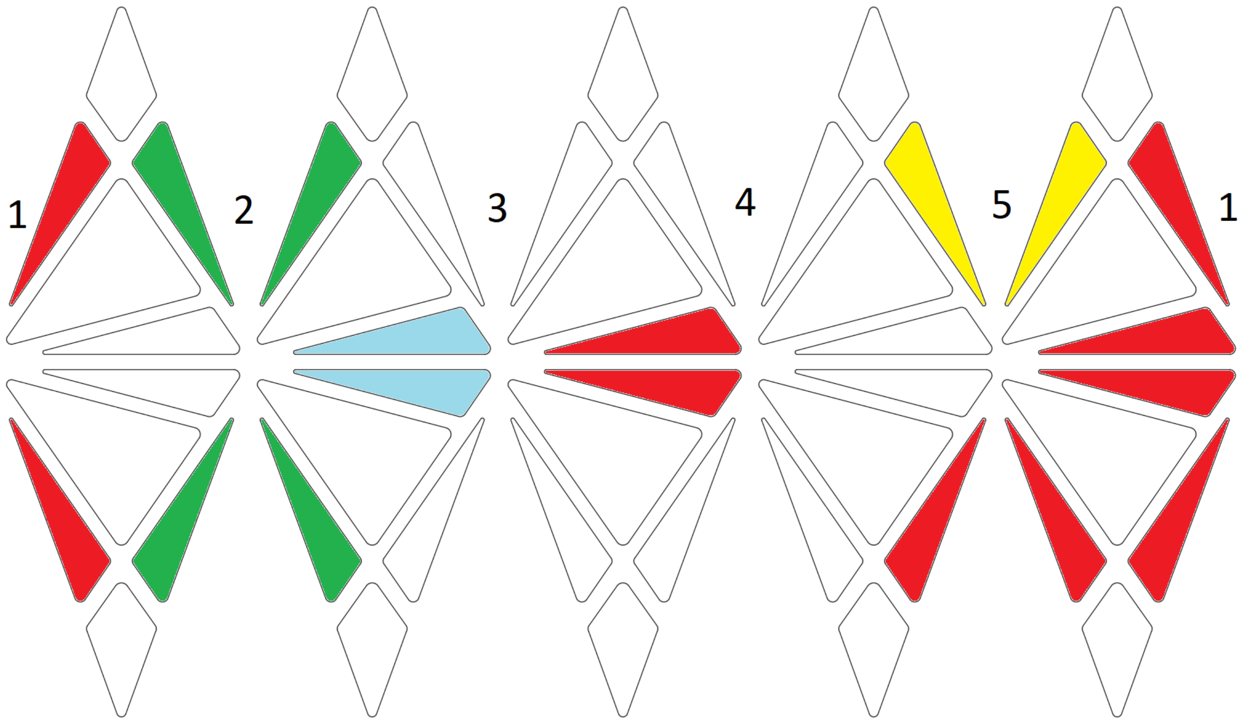
Location 9

$G4_4^2$ $3''4''3''$



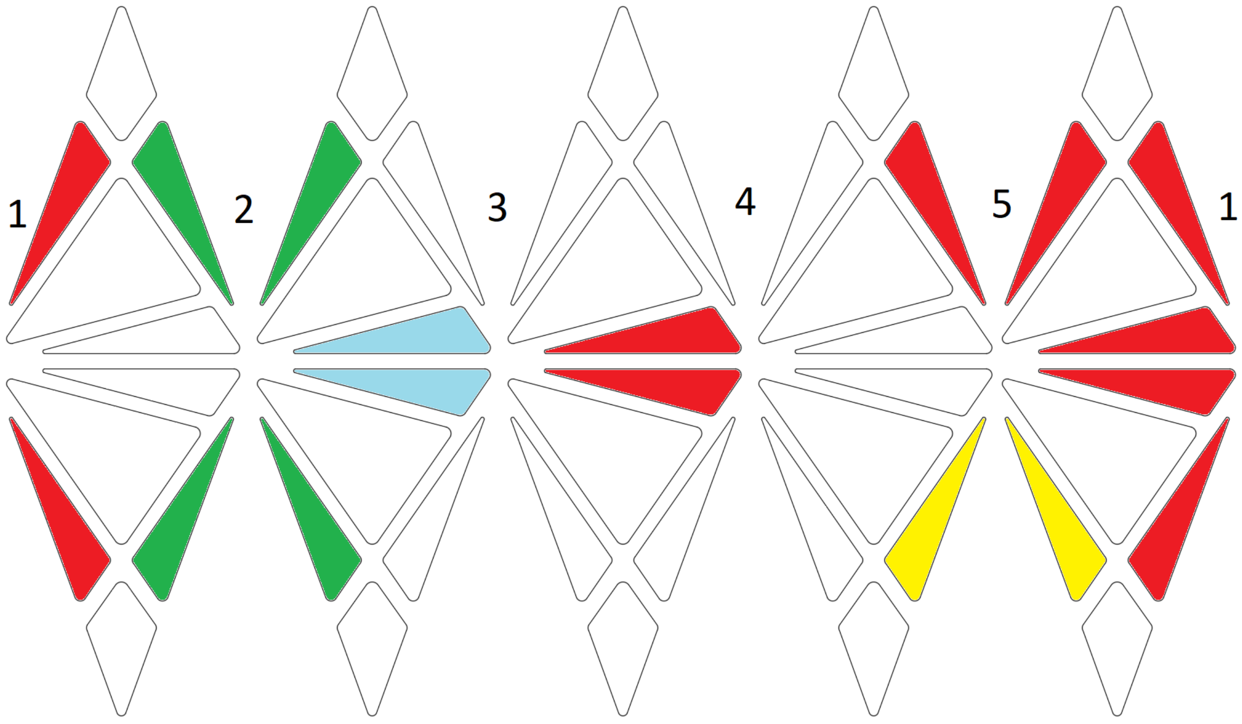
Location 10

$G4_5^1 G2_1^2$ $4''54''2''1''2''$



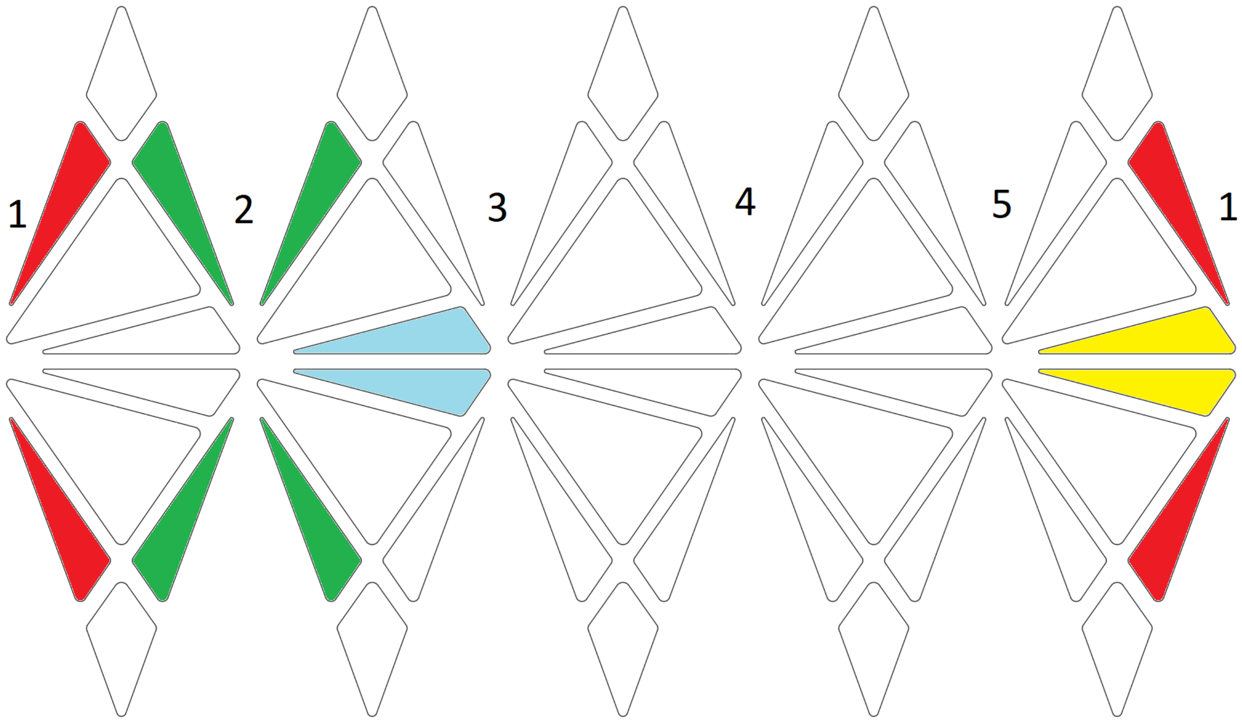
Location 11

$G4_5^{-1} G2_1^2$ $4''5'4''2''1''2''$



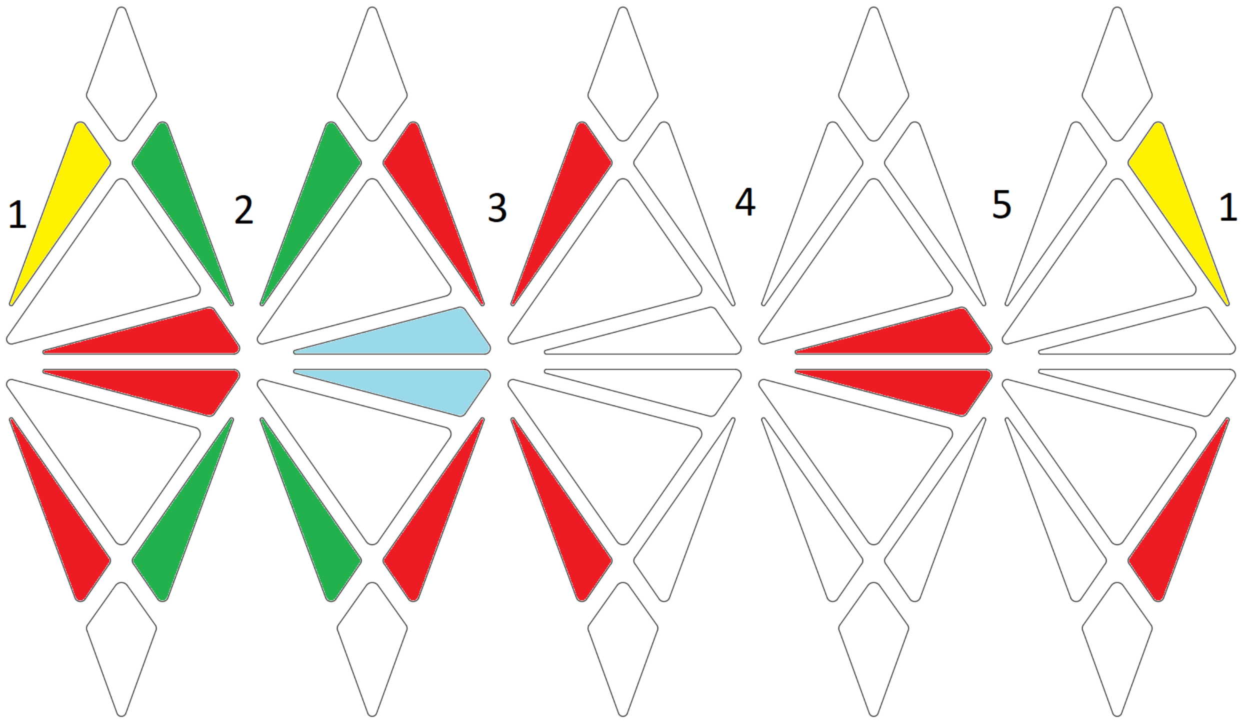
Location 12

$G2_1^2$ $2''1''2''$



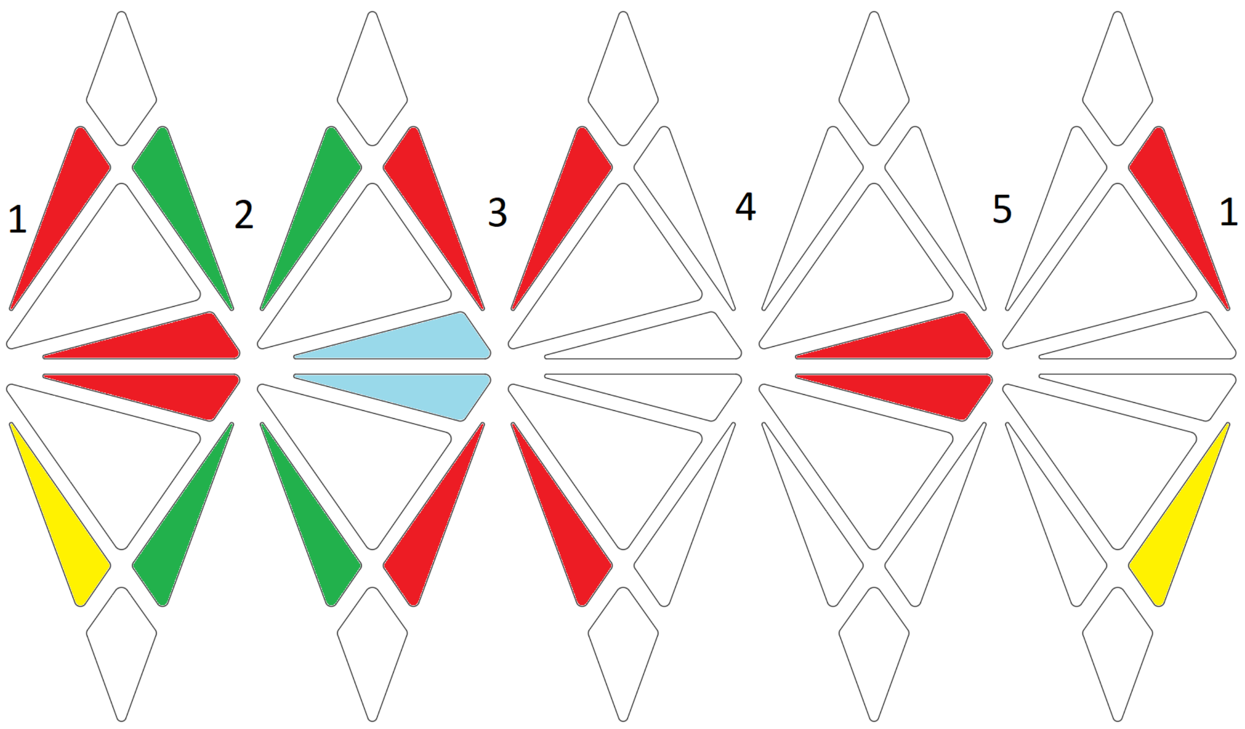
Location 13

$G4_1^{-1}G2_3^4$ $5''1'5''4''3''4''$



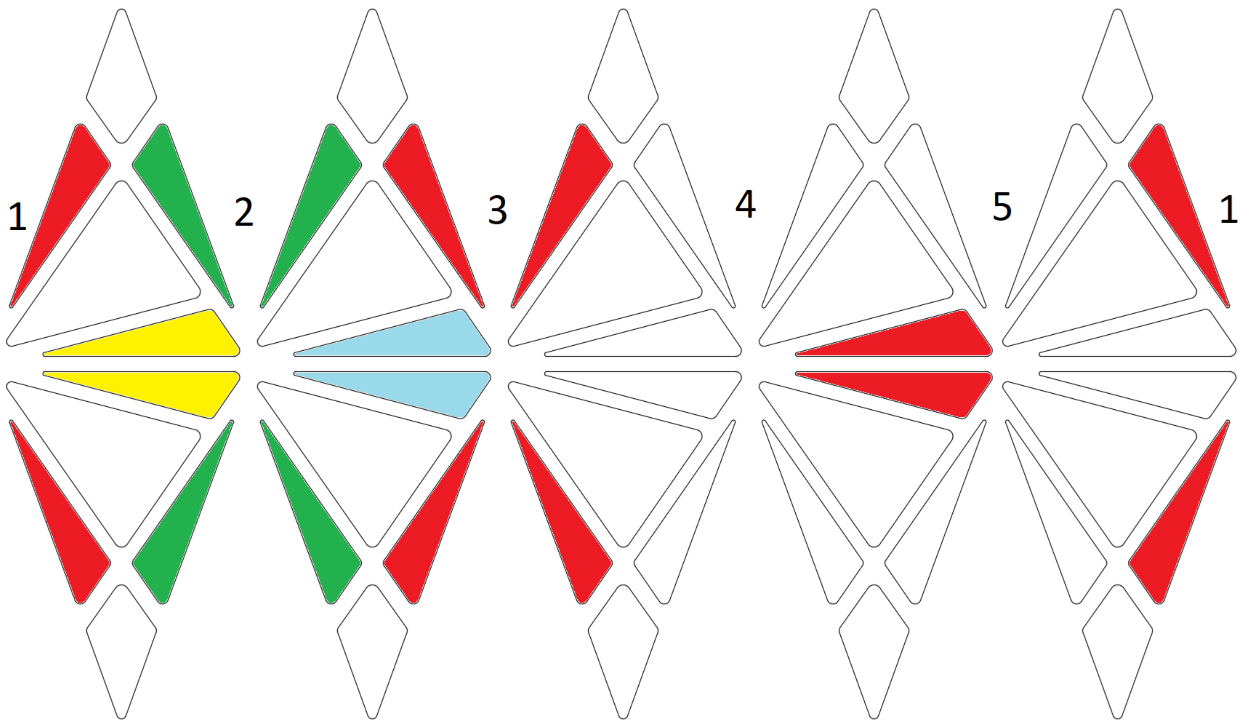
Location 14

$G4_1^1G2_3^4$ $5''15''4''3''4''$



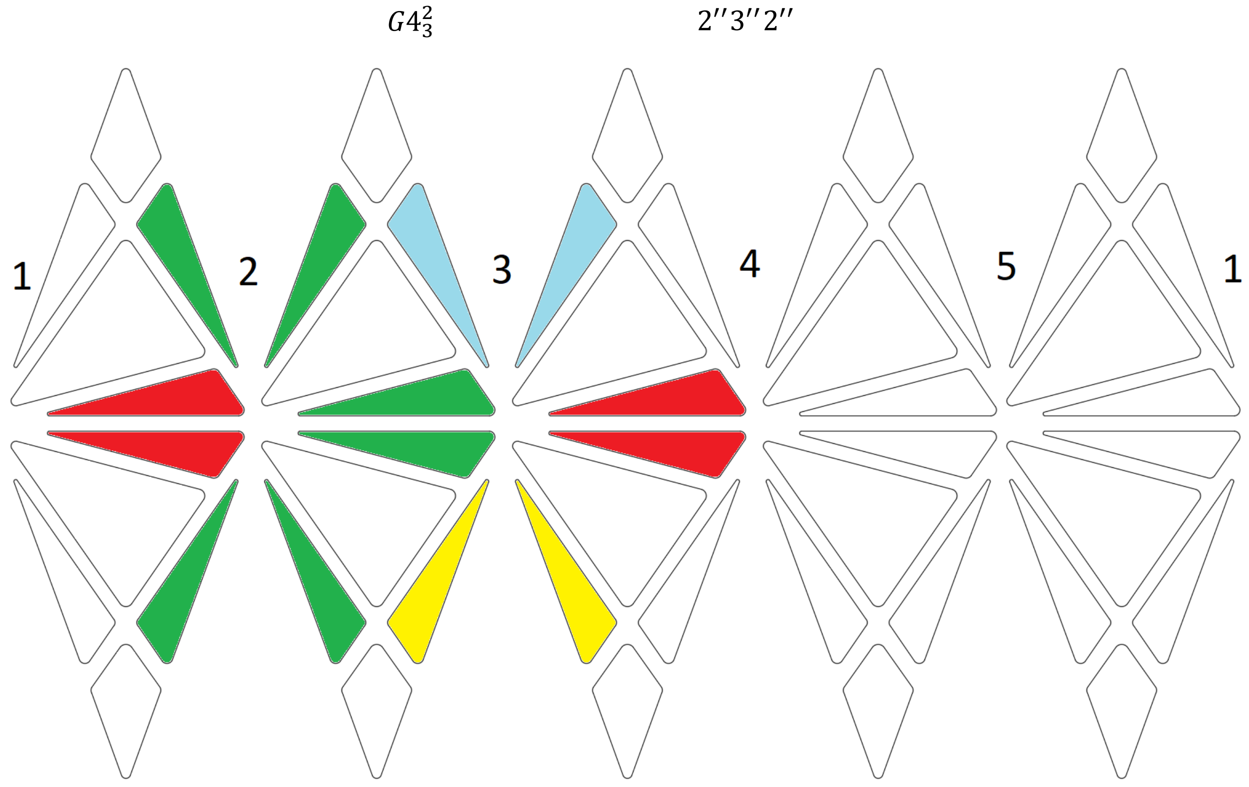
Location 15

$G4_1^2G2_3^4$ $5''1''5''4''3''4''$

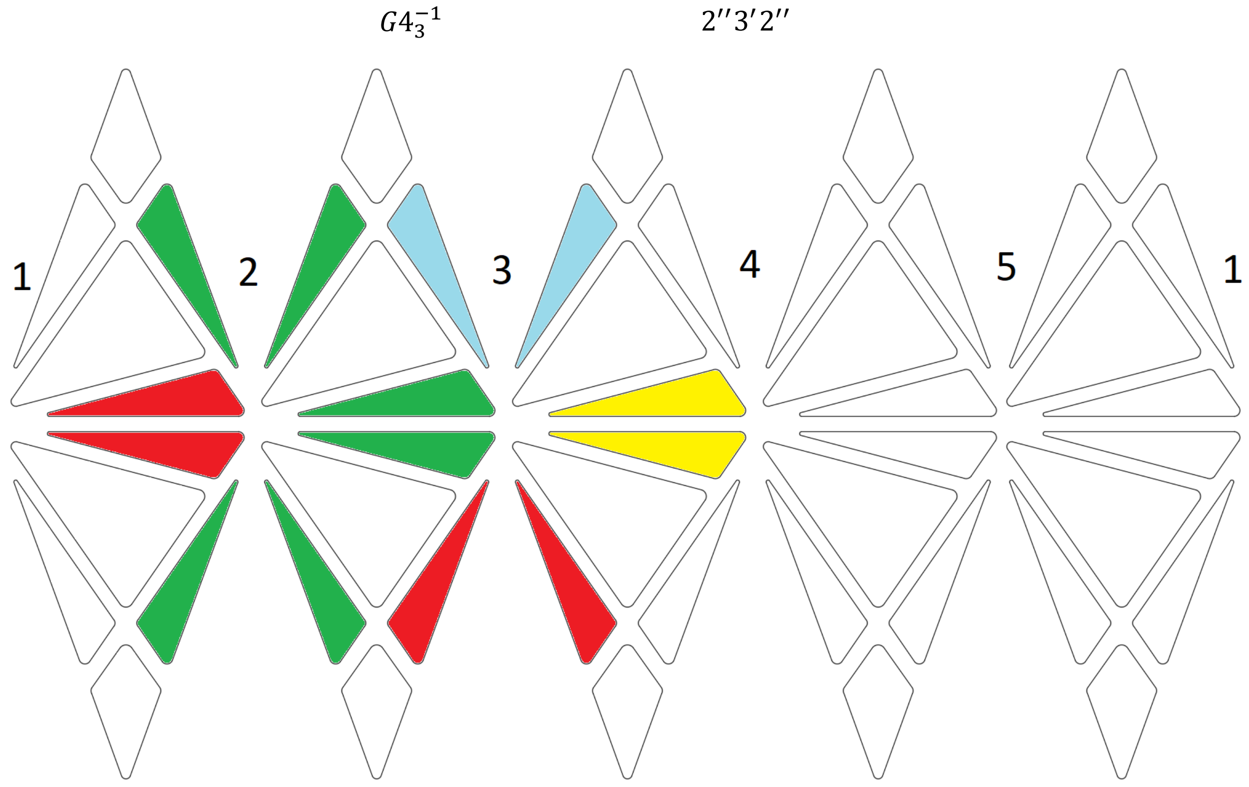


Permuting the Fourth Edge

Location 5

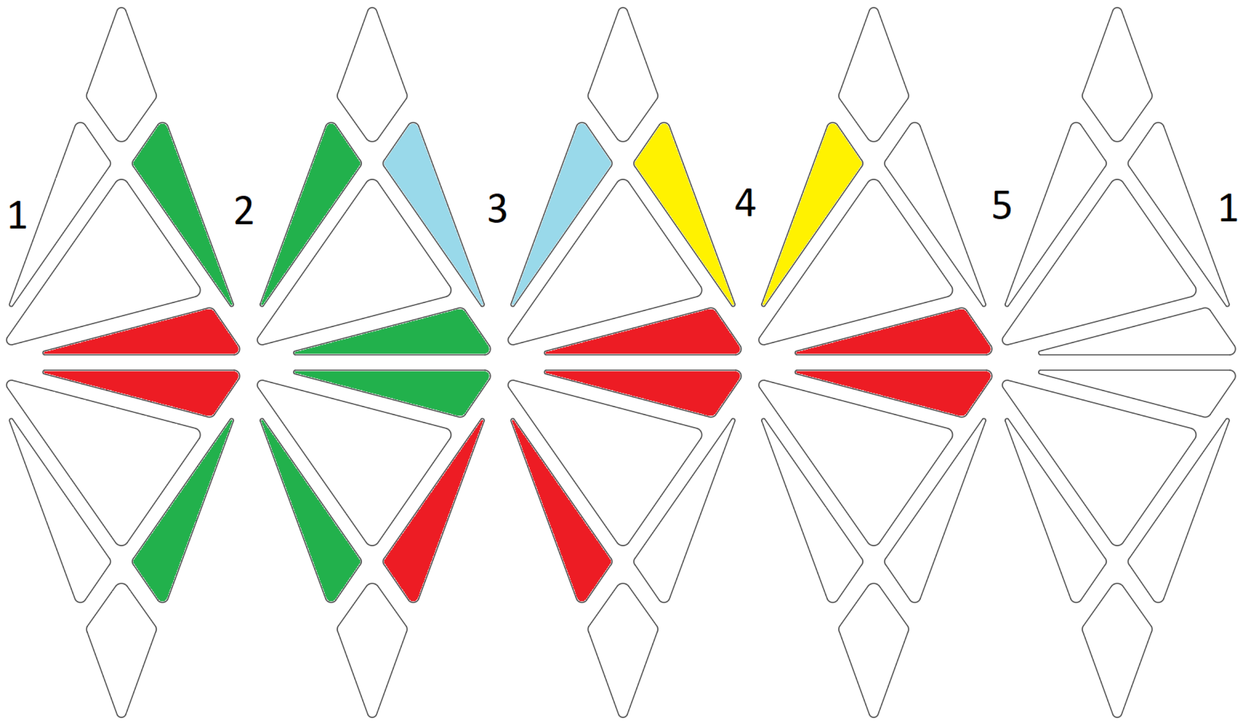


Location 6



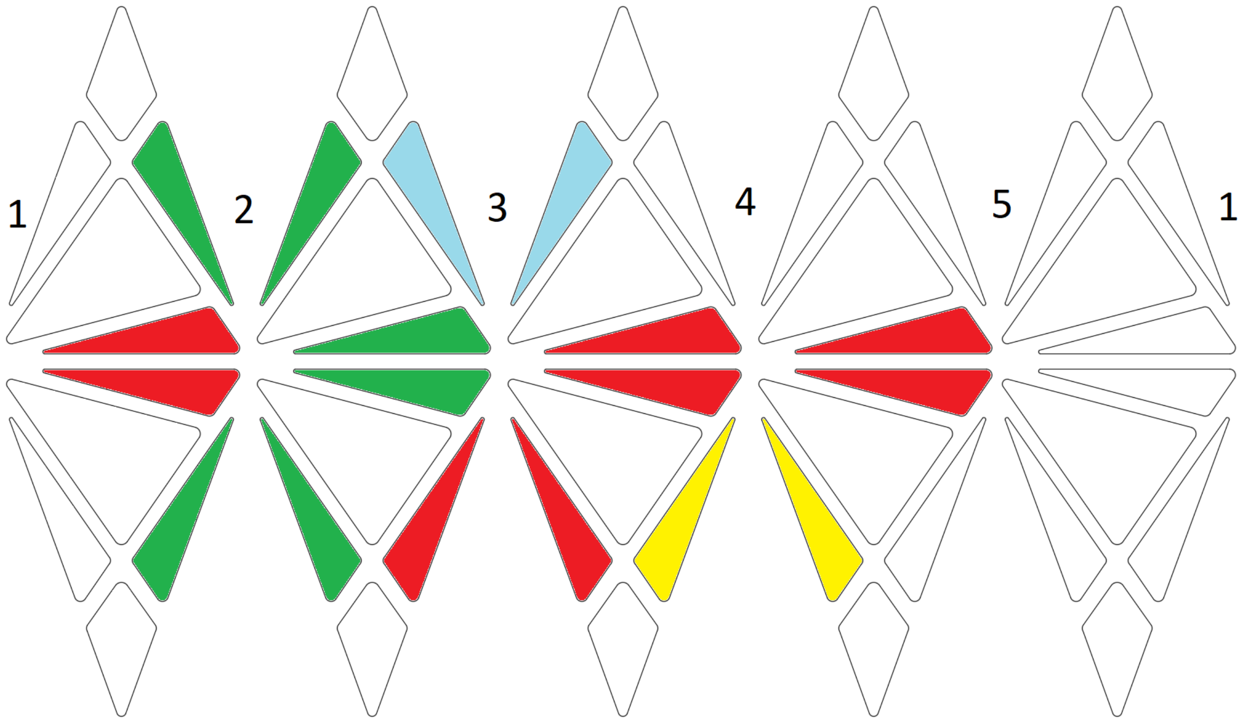
Location 7

$E3_4^{-1,1}G4_3^1$ $3''1''5''45''4'1''3''2''32''$



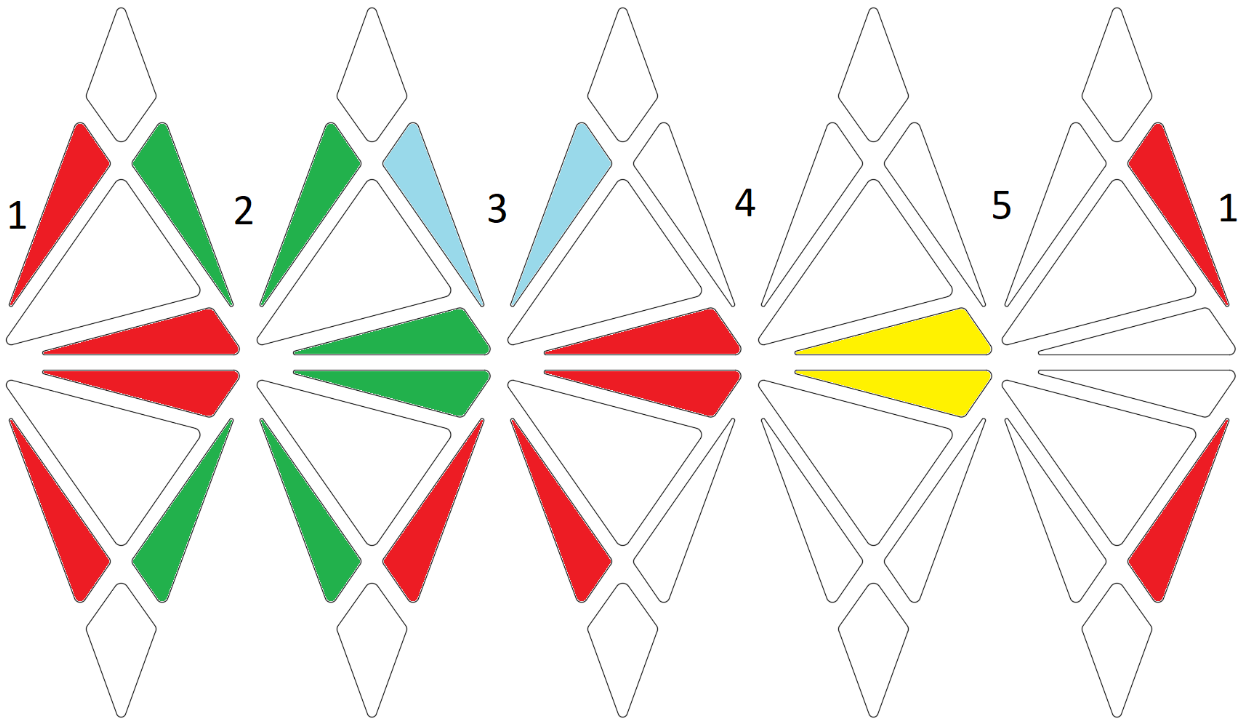
Location 8

$E3_4^{1,1}G4_3^1$ $3''1''5''4'5''41''3''2''32''$



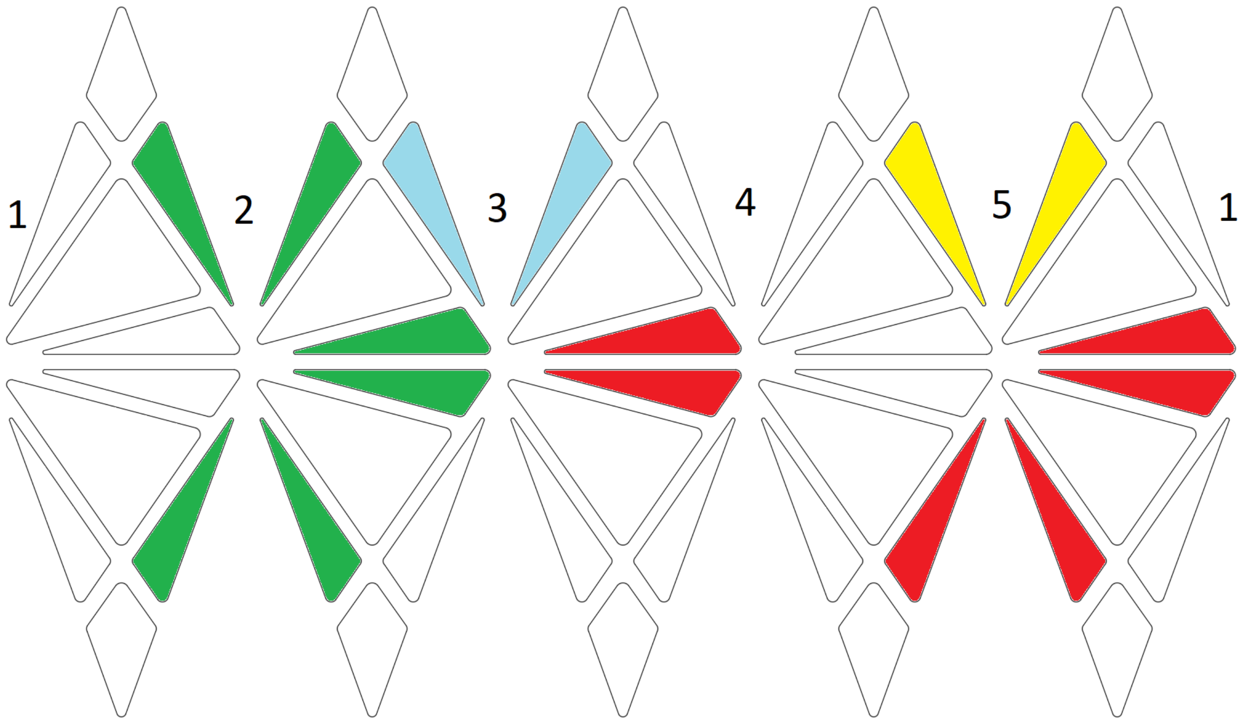
Location 9

$G2_1^5G4_3^1$ $5''1''5''2''32''$



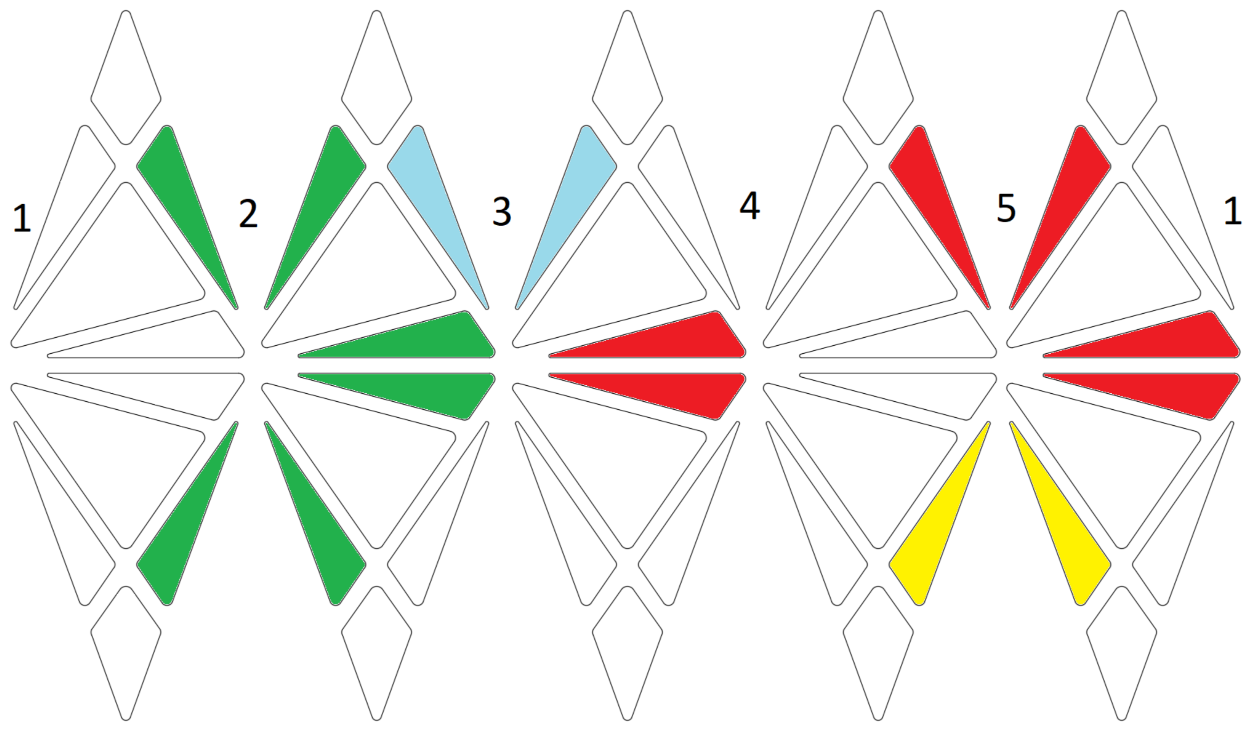
Location 10

$G4_5^1E3_3^{-1,-1}$ $4''54''2''5''34''3'4''5''2''$



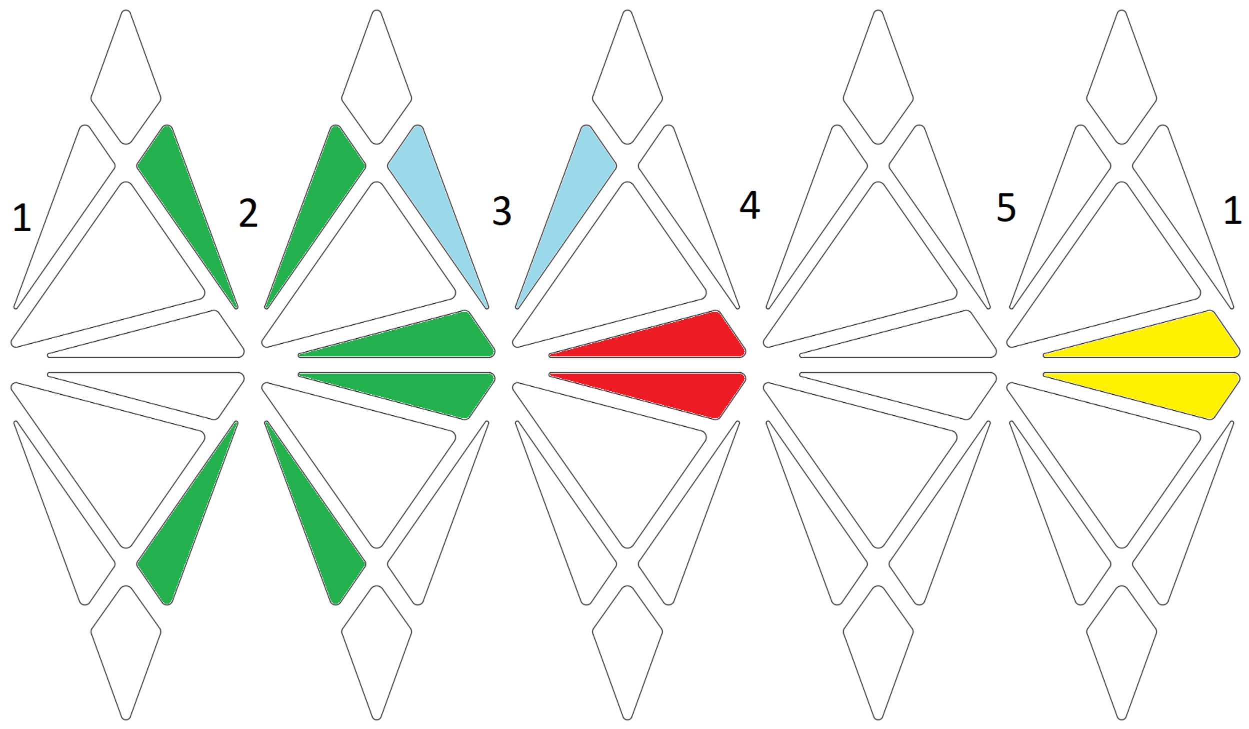
Location 11

$G4_5^{-1}E3_3^{-1,-1}$ $4''5'4''2''5''34''3'4''5''2''$



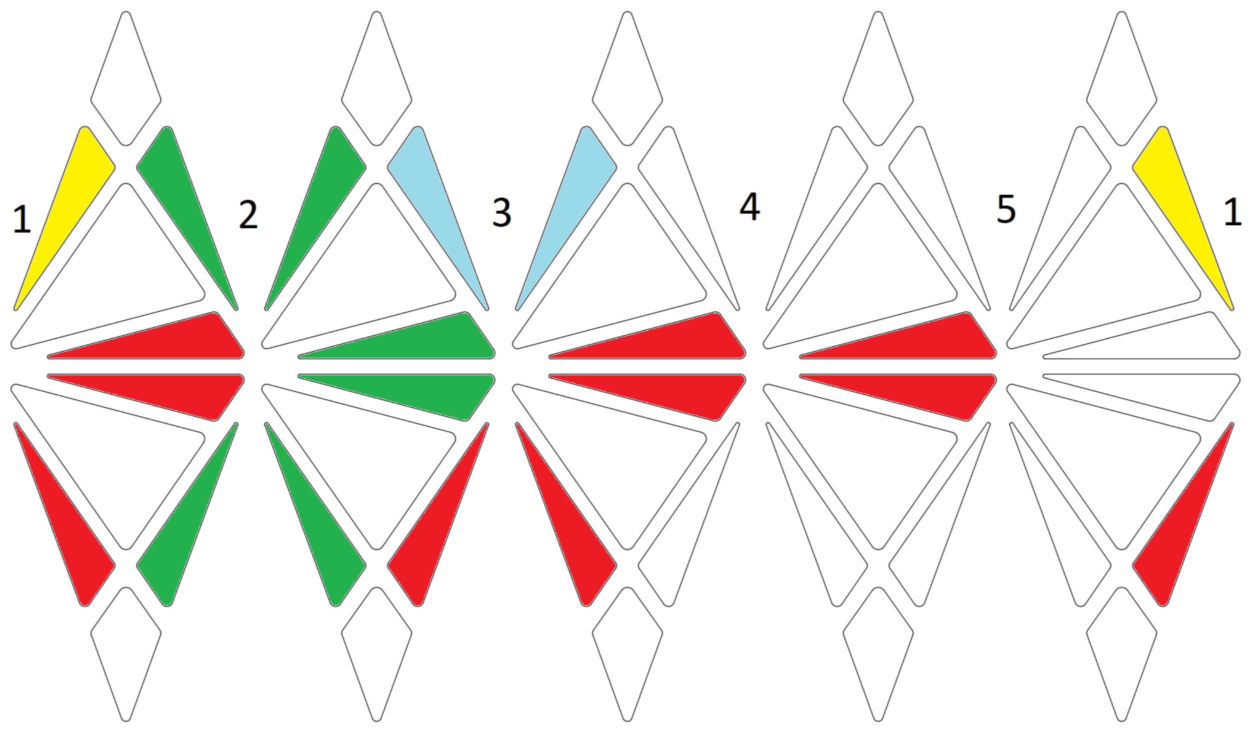
Location 12

$E3_3^{-1,-1}$ $2''5''34''3'4''5''2''$



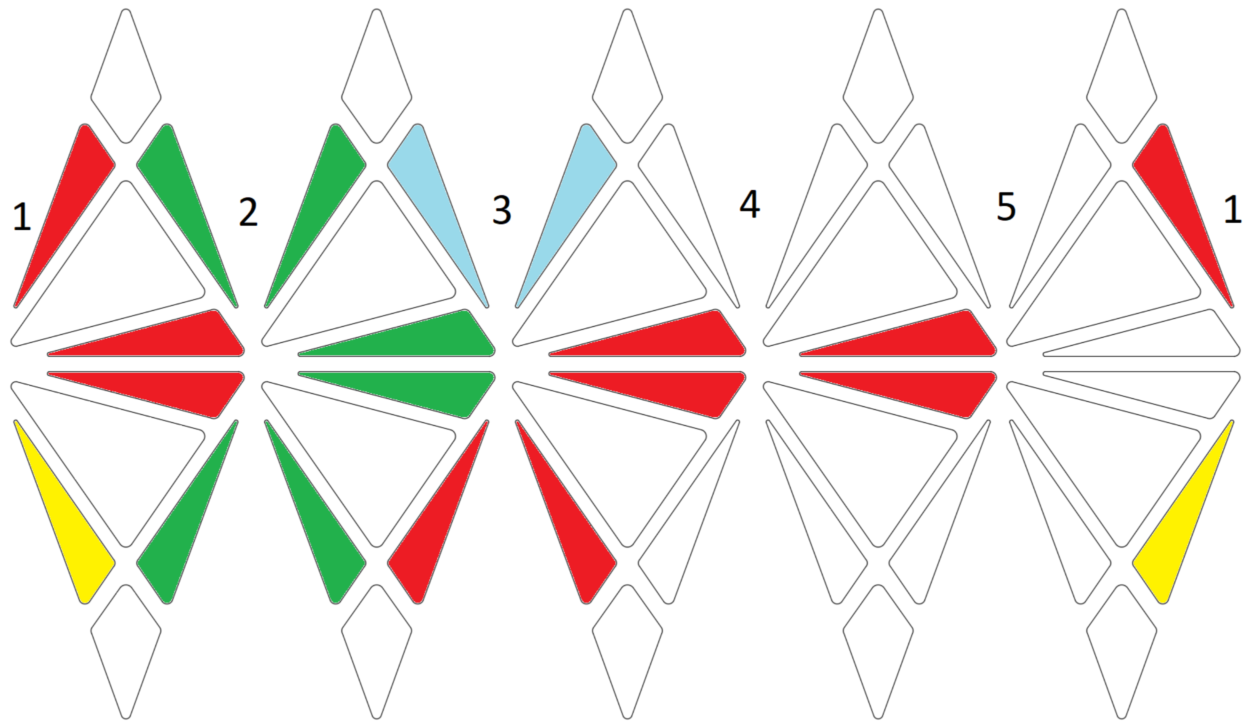
Location 13

$G4_1^1 G4_3^1$ $5''15''2''32''$



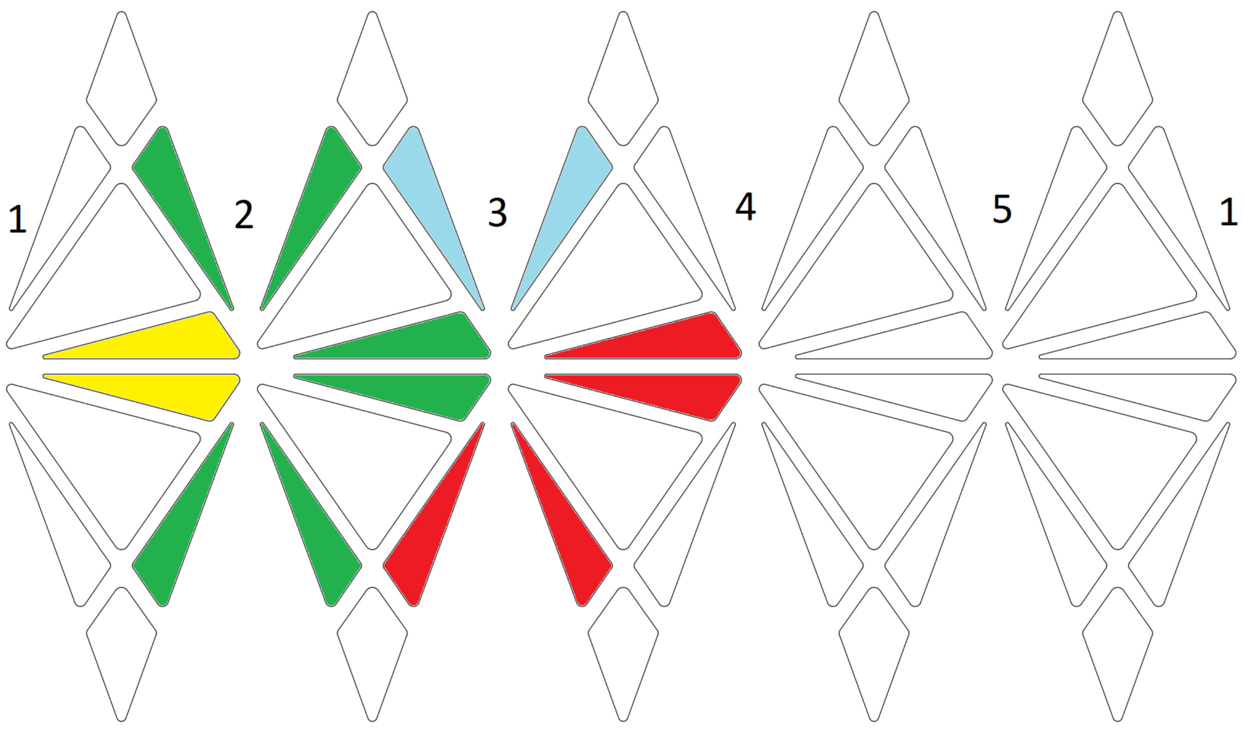
Location 14

$G4_1^{-1} G4_3^1$ $5''1'5''2''32''$



Location 15

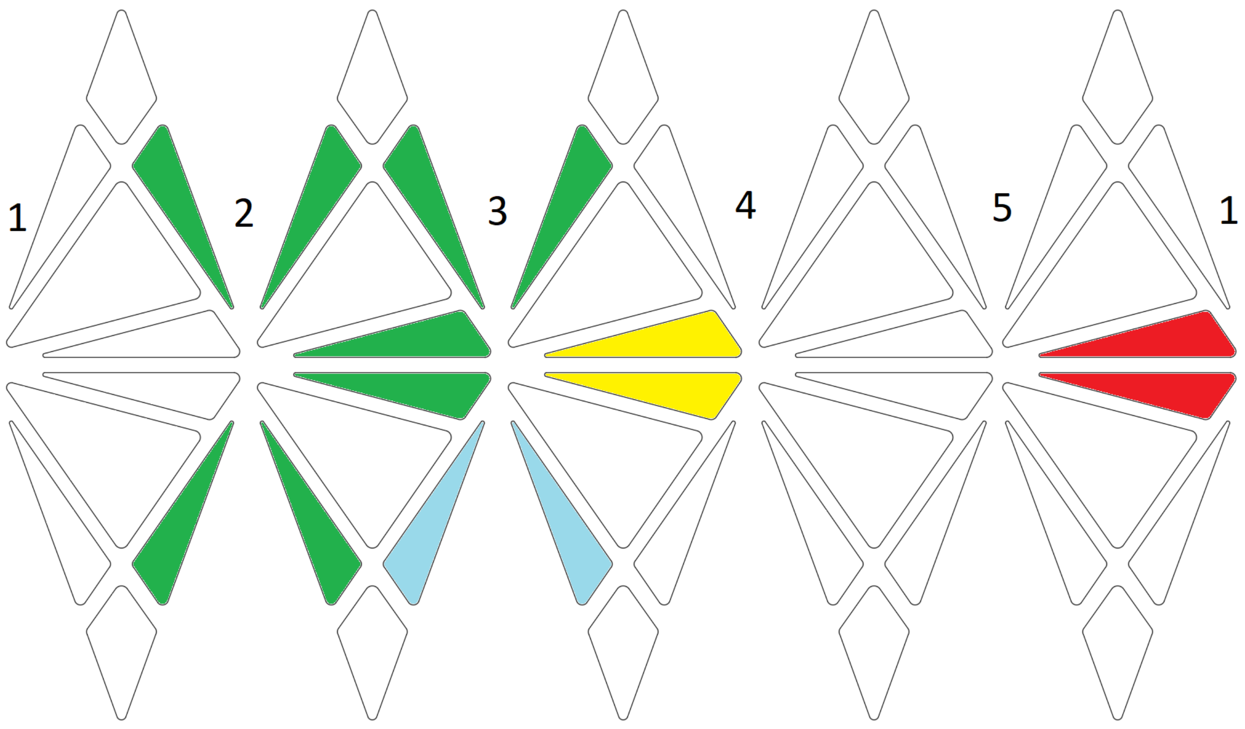
$G4_3^1$ $2''32''$



Permuting the Fifth Edge

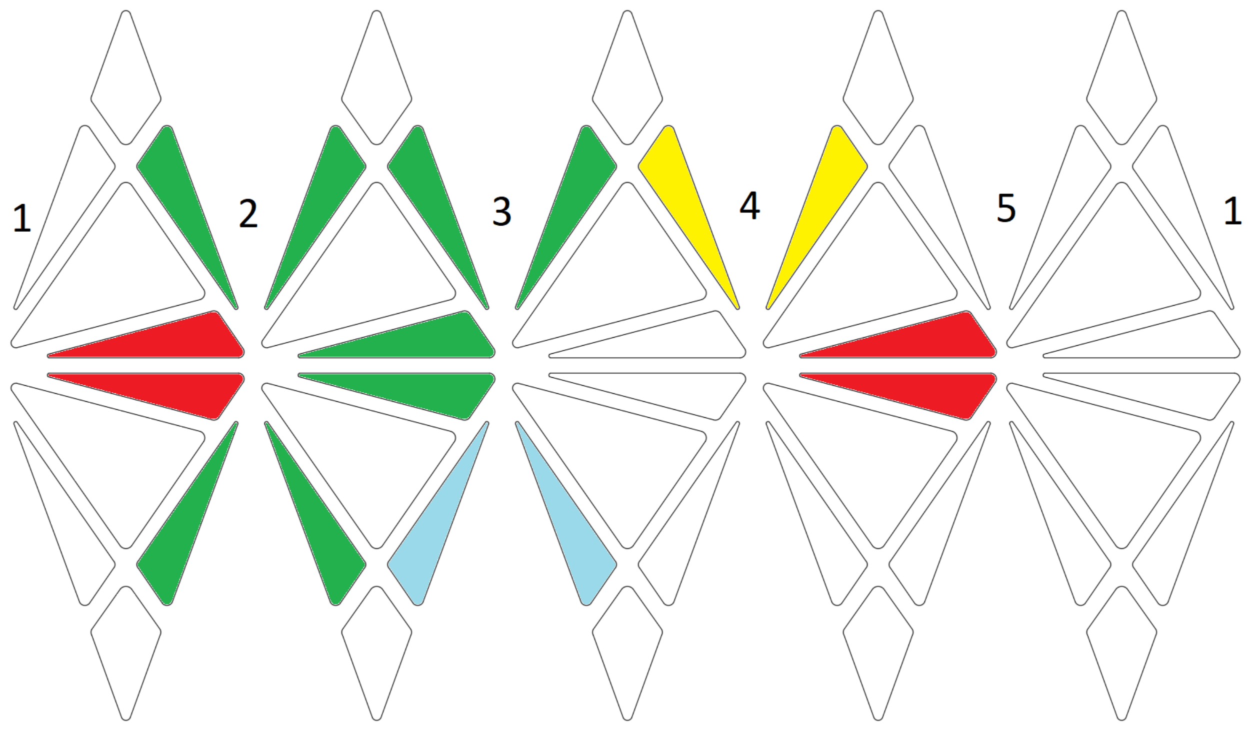
Location 6

$E3_3^{1,1}$ $2''5''4''3'4''35''2''$



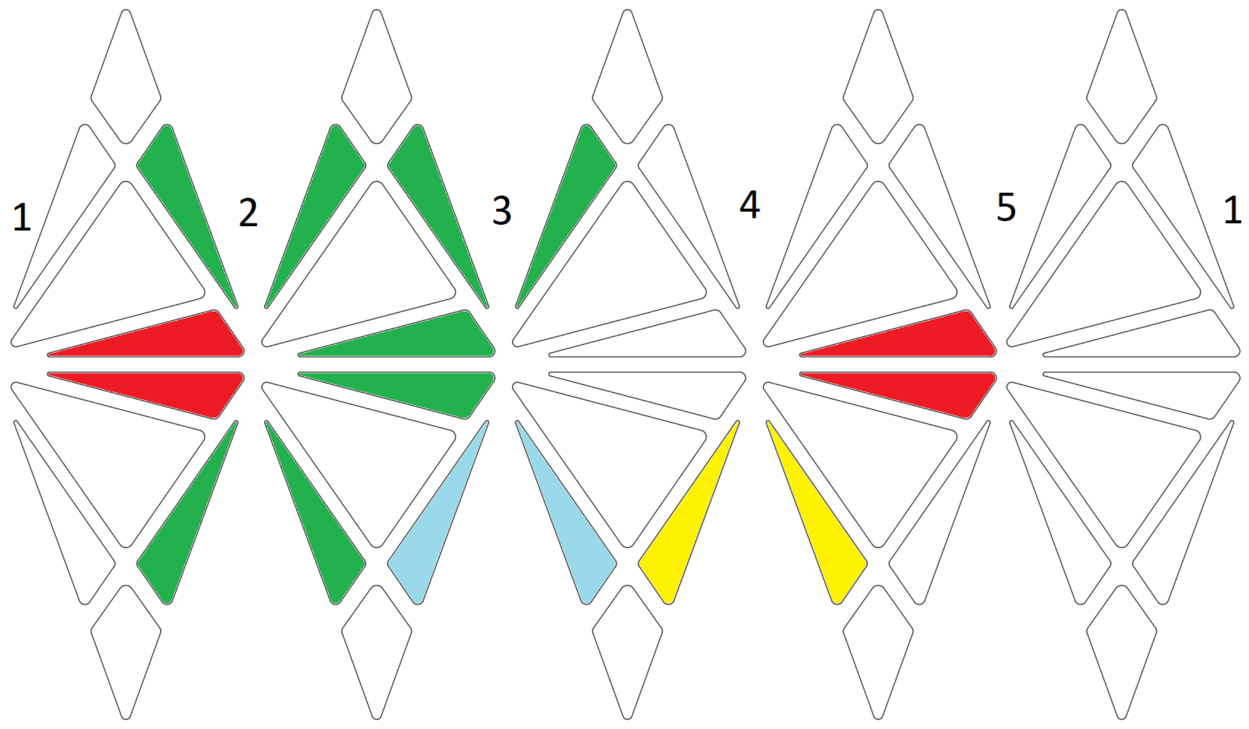
Location 7

$G4_3^1 E3_4^{-1,1} G4_3^{-1}$ $2''32''3''1''5''45''4'1''3''2''3'2''$



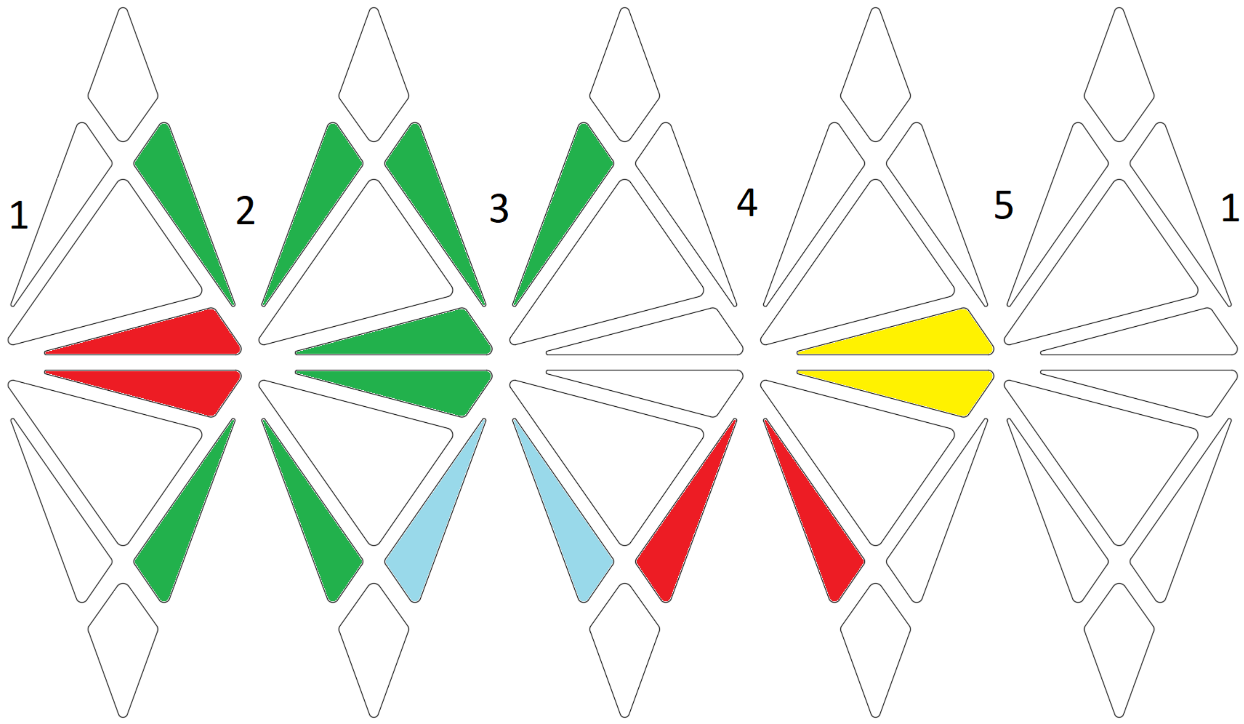
Location 8

$G4_3^1 E3_4^{1,1} G4_3^{-1}$ $2''32''3''1''5''4'5''41''3''2''3'2''$



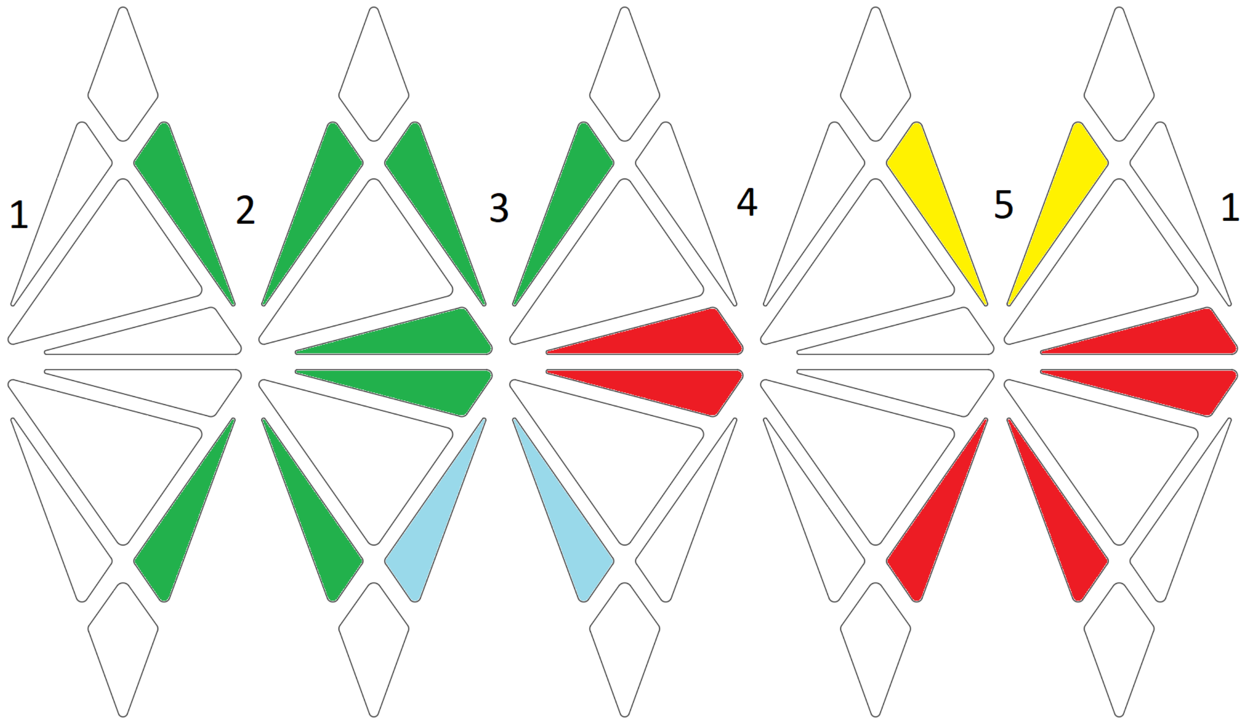
Location 9

$G4_3^1 E3_4^{1,-1} G4_3^{-1}$ $2''32''3''1''4'5''45''1''3''2''3'2''$



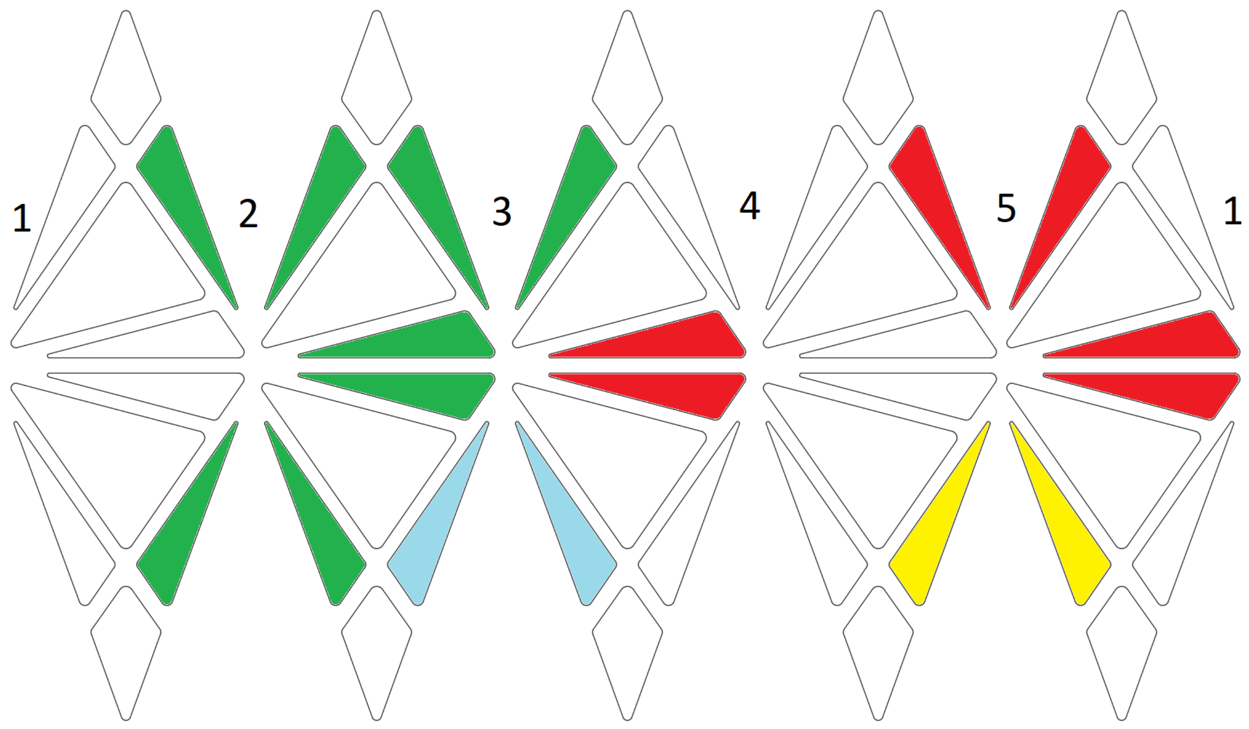
Location 10

$G4_5^1 E3_3^{1,-1}$ $4''54''2''5''3'4''34''5''2''$



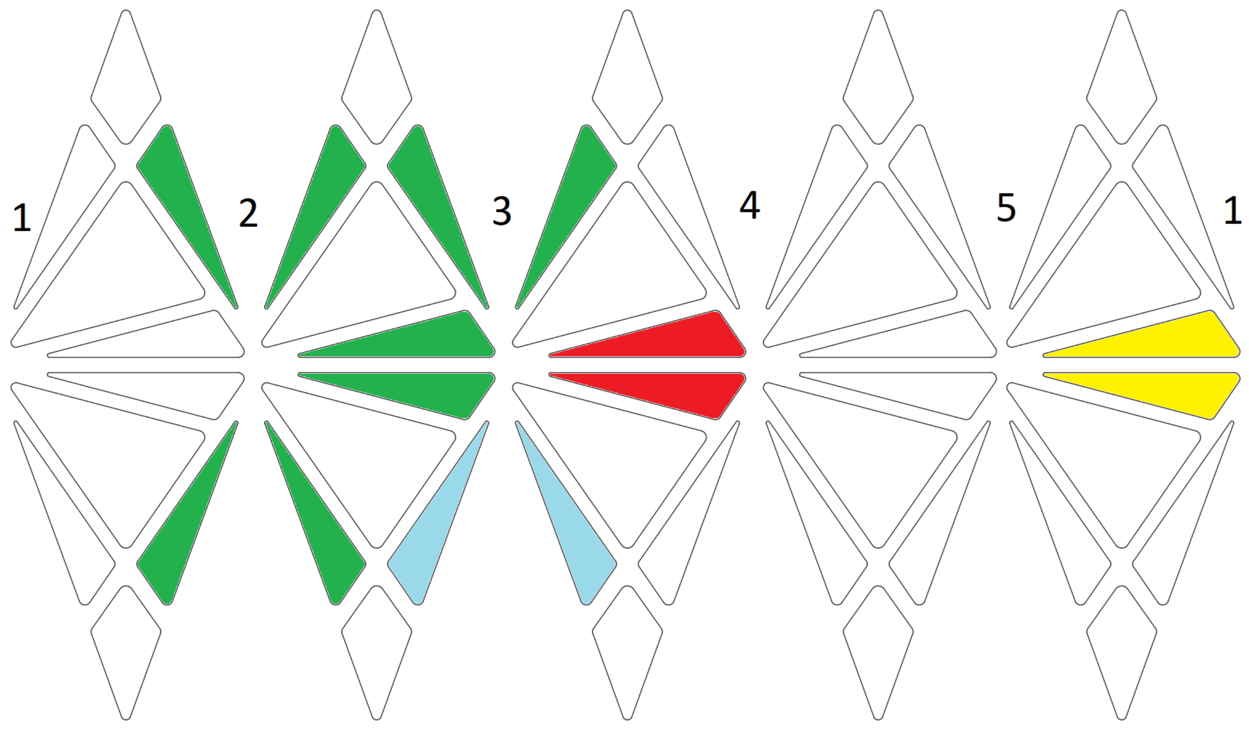
Location 11

$G4_5^{-1}E3_3^{1,-1}$ $4''5'4''2''5''3'4''34''5''2''$



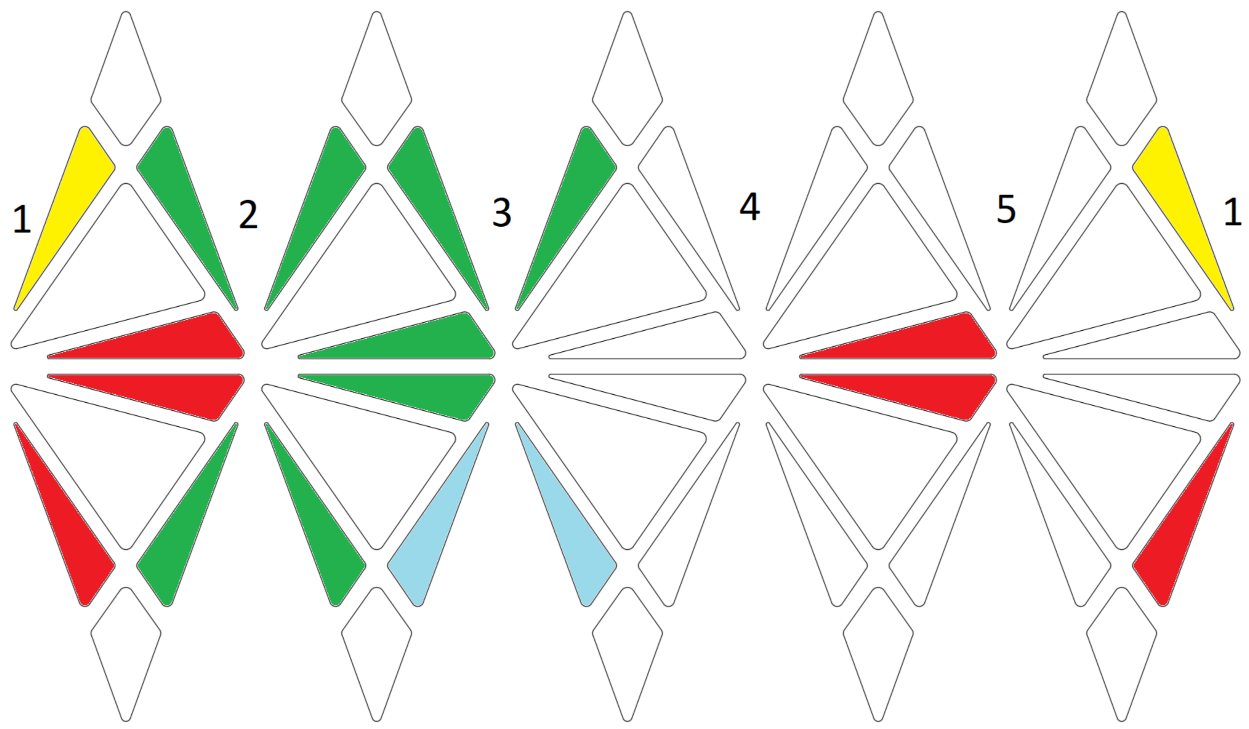
Location 12

$E3_3^{1,-1}$ $2''5''3'4''34''5''2''$



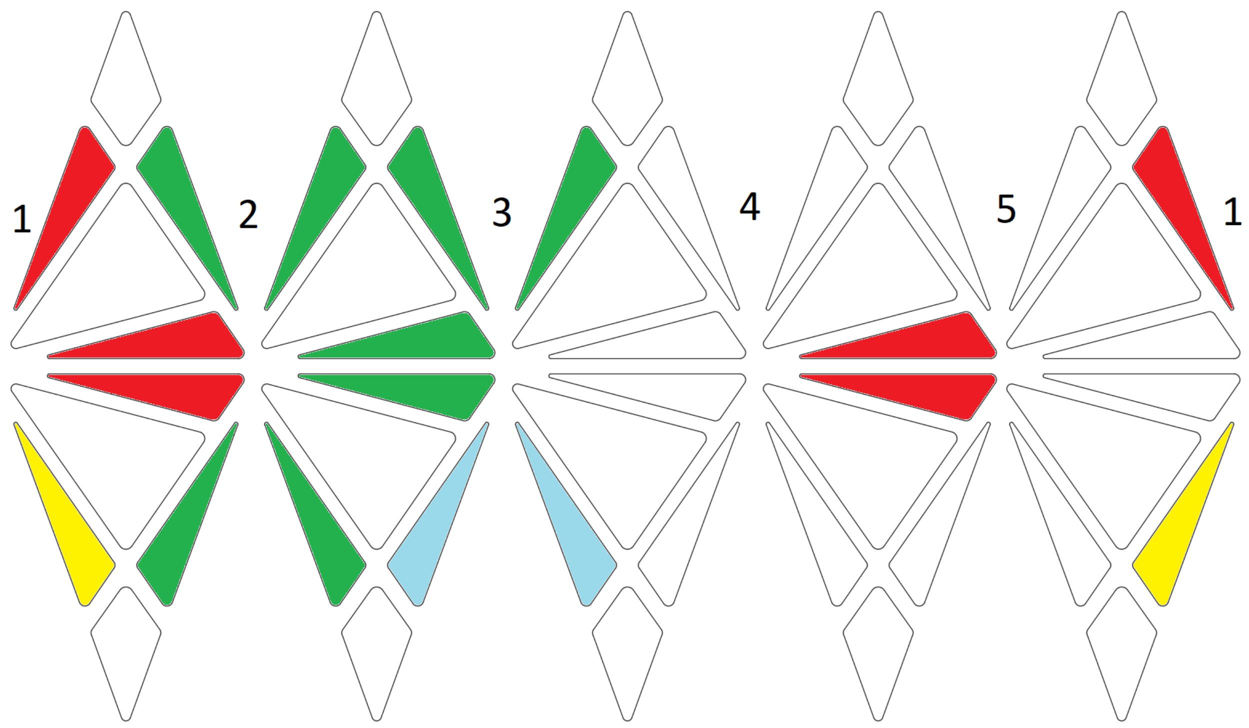
Location 13

$G4_3^1 G4_1^1 G4_3^{-1}$ $2''32''5''15''2''3'2''$



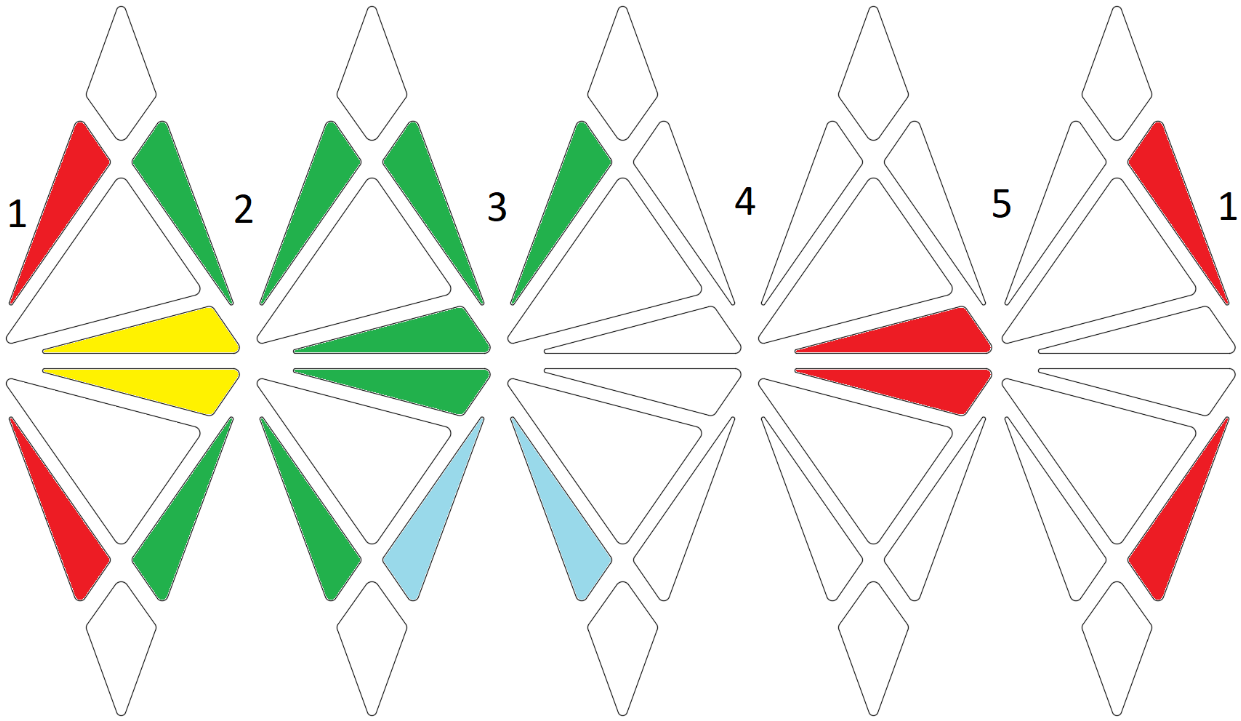
Location 14

$G4_3^1 G4_1^{-1} G4_3^{-1}$ $2''32''5''1'5''2''3'2''$



Location 15

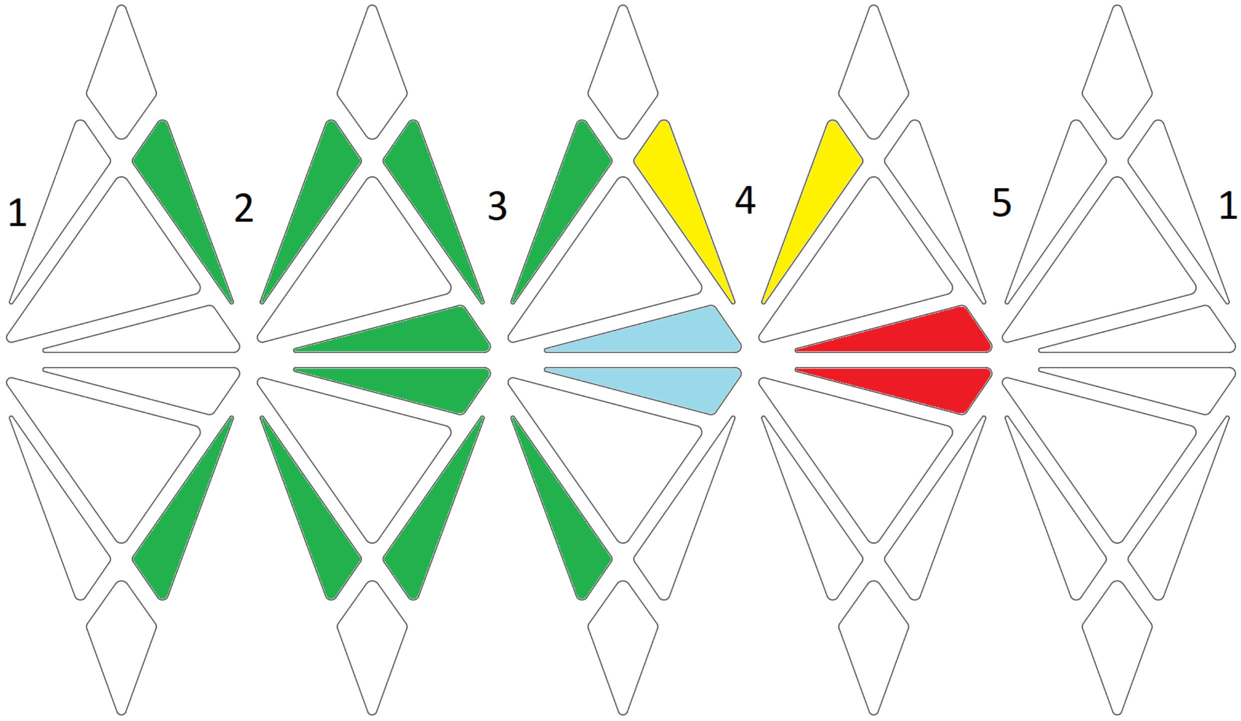
$G4_1^{-1}G4_3^1G4_1^1G4_3^{-1}$ 5″1′5″2″32″5″15″2″3′2″



Permuting the Sixth Edge

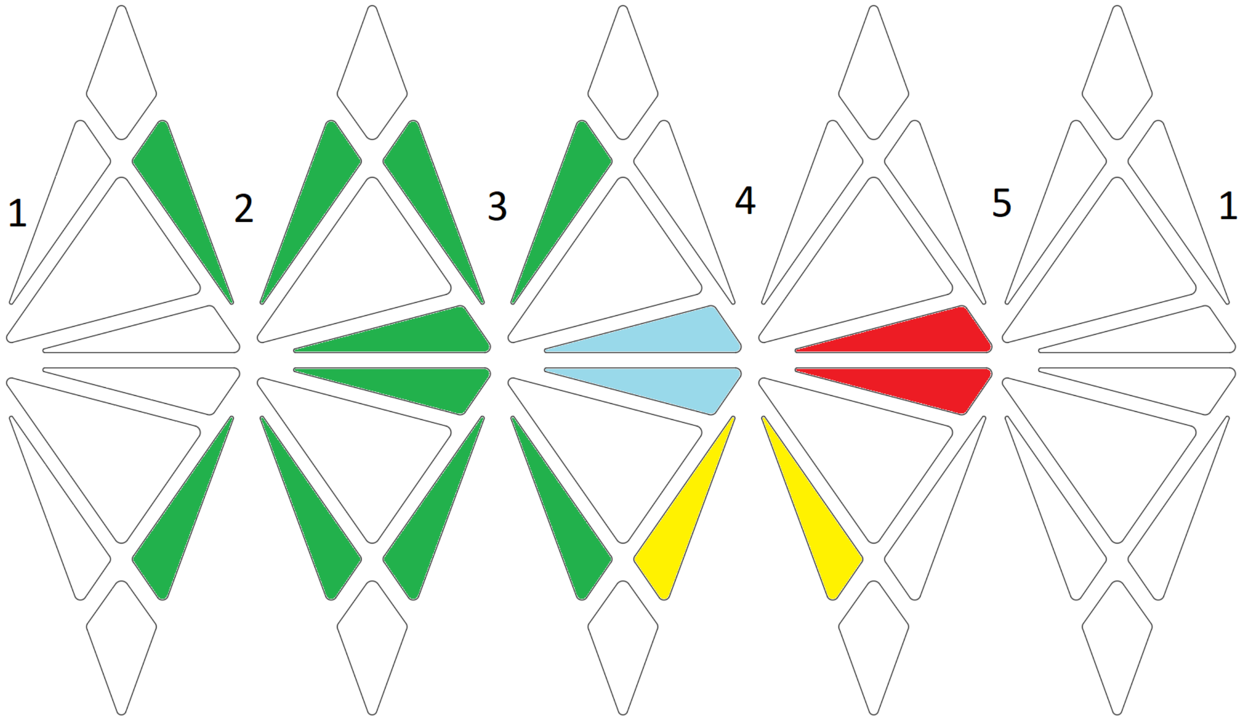
Location 7

$G2_2^3E3_4^{-1,1}G2_2^3$ 3″2″1″5″45″4′1″2″3″



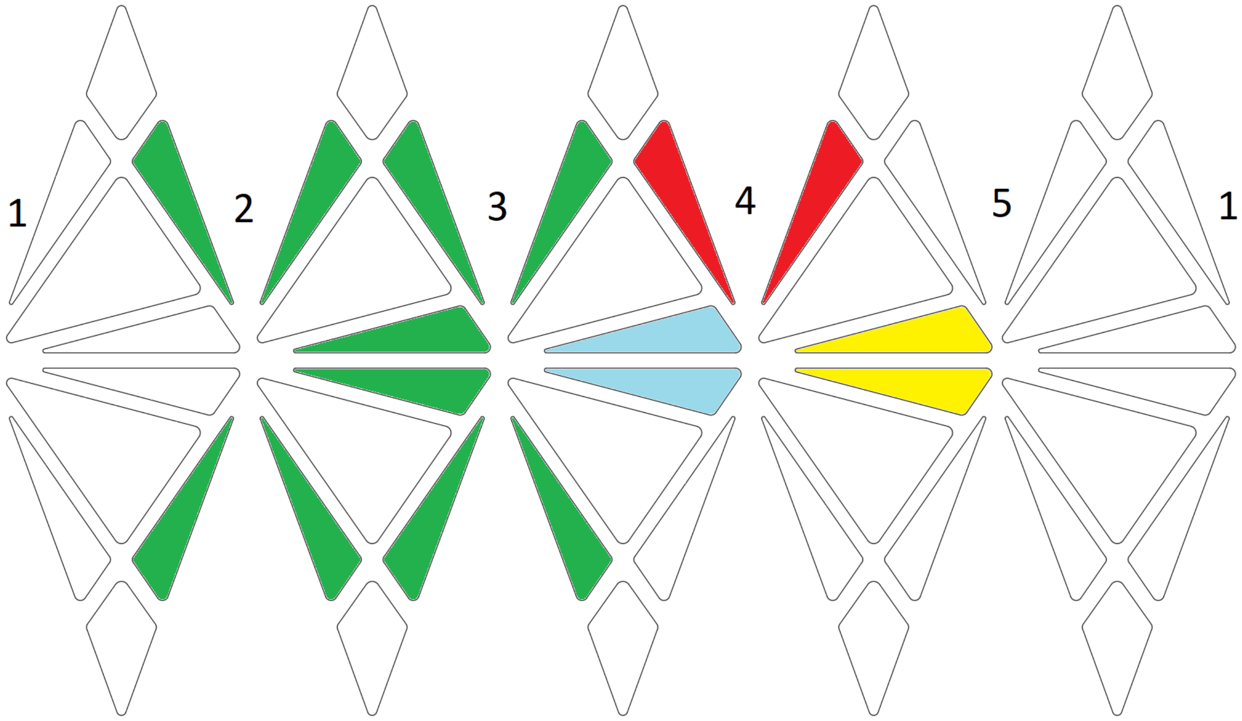
Location 8

$G2_2^3E3_4^{1,1}G2_2^3$ $3''2''1''5''4'5''41''2''3''$



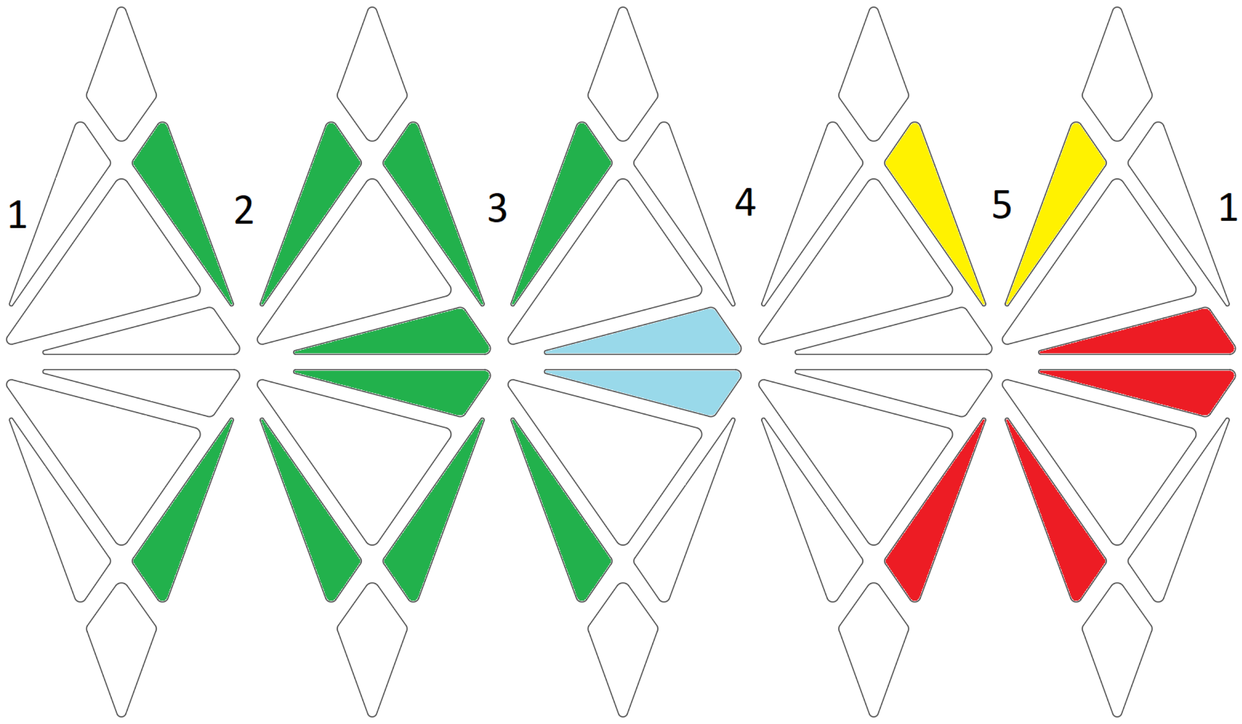
Location 9

$G2_2^3E3_4^{-1,-1}G2_2^3$ $3''2''1''45''4'5''1''2''3''$



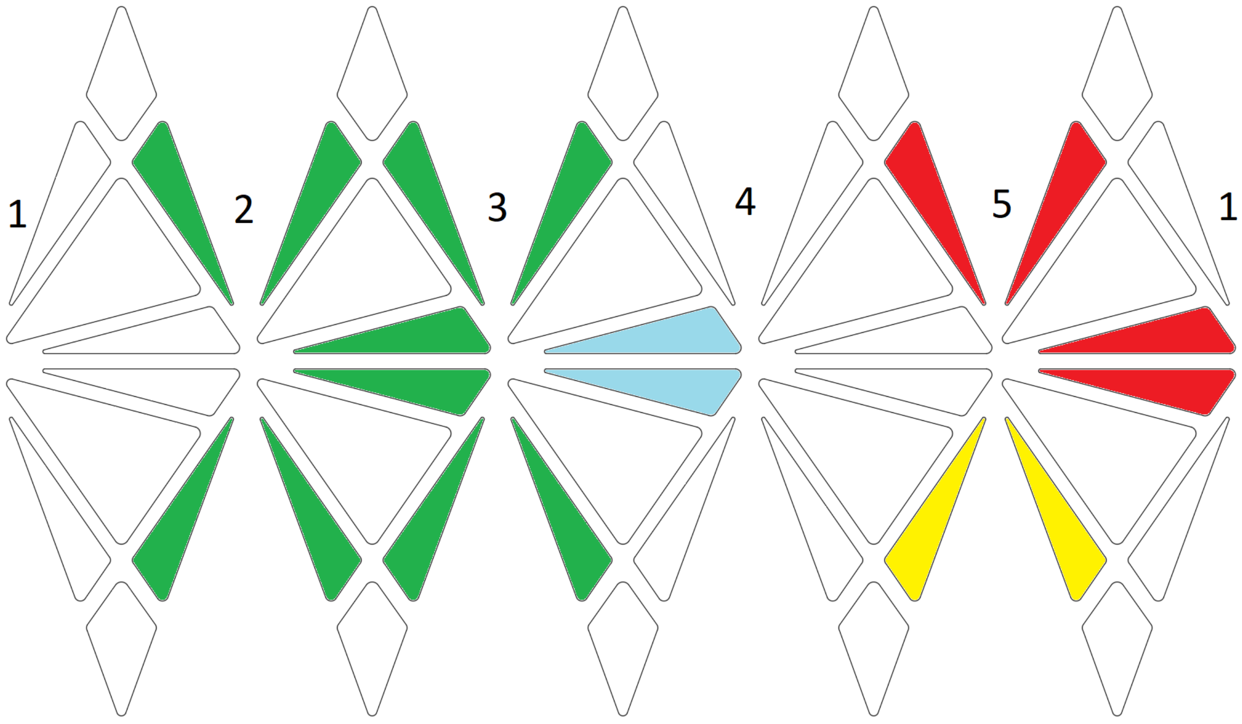
Location 10

$G4_5^{-1}$ $4''5'4''$



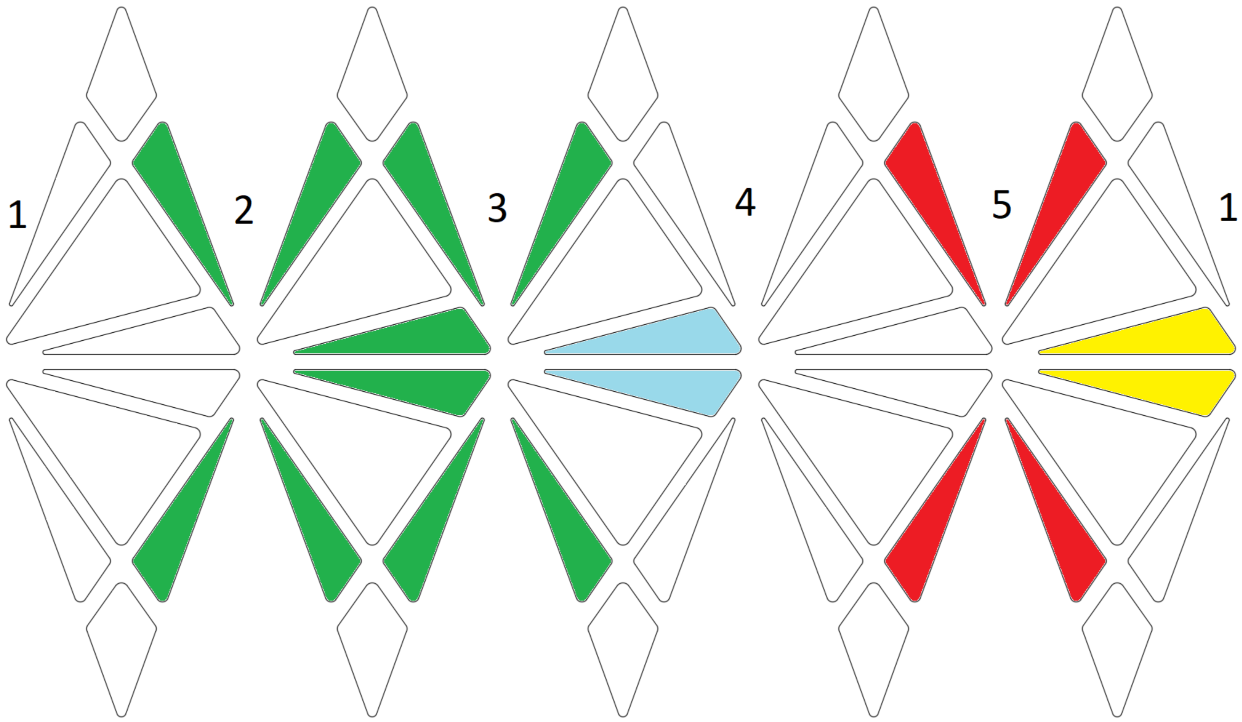
Location 11

$G4_5^{1}$ $4''54''$



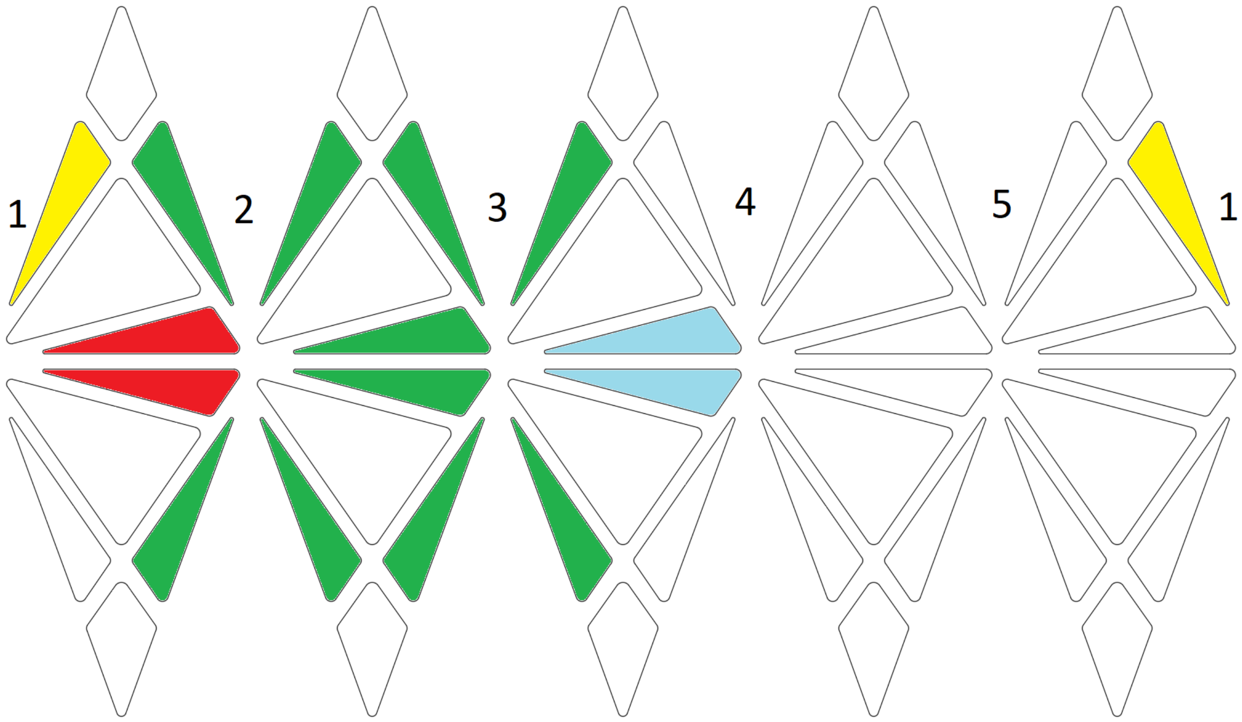
Location 12

$G4_5^2$ $4''5''4''$



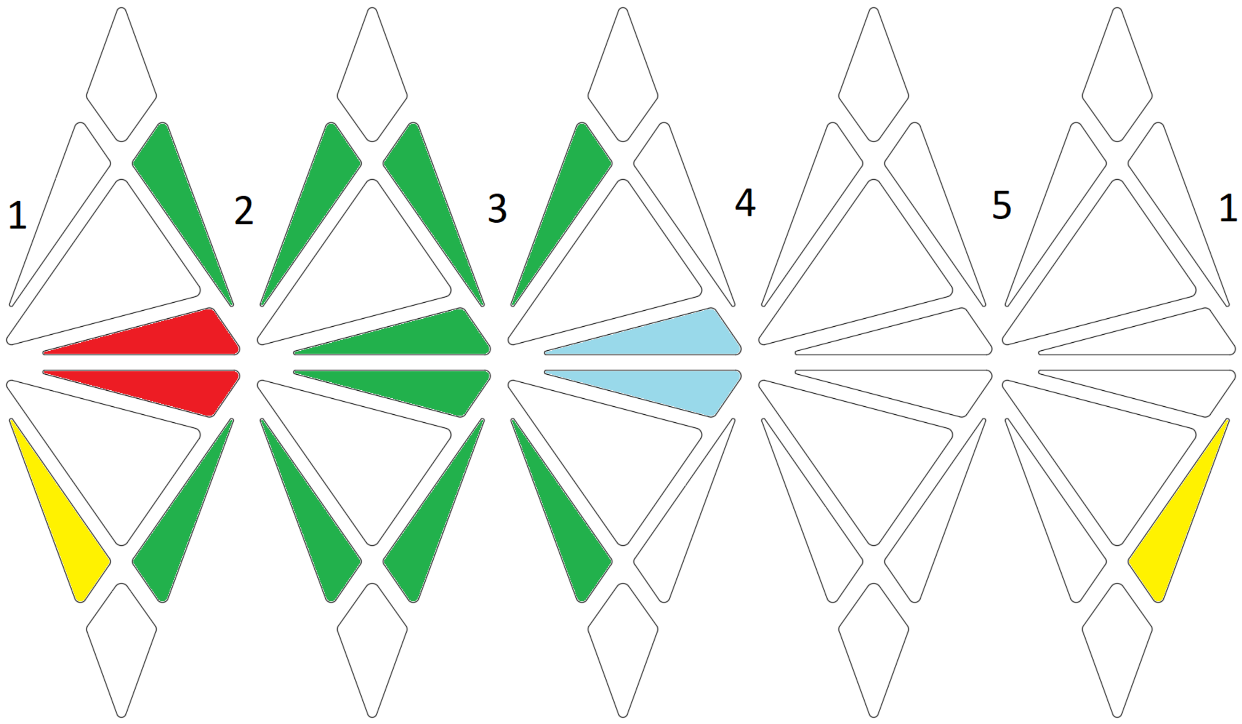
Location 13

$E3_1^{-1,1}$ $5''3''2''12''1'3''5''$



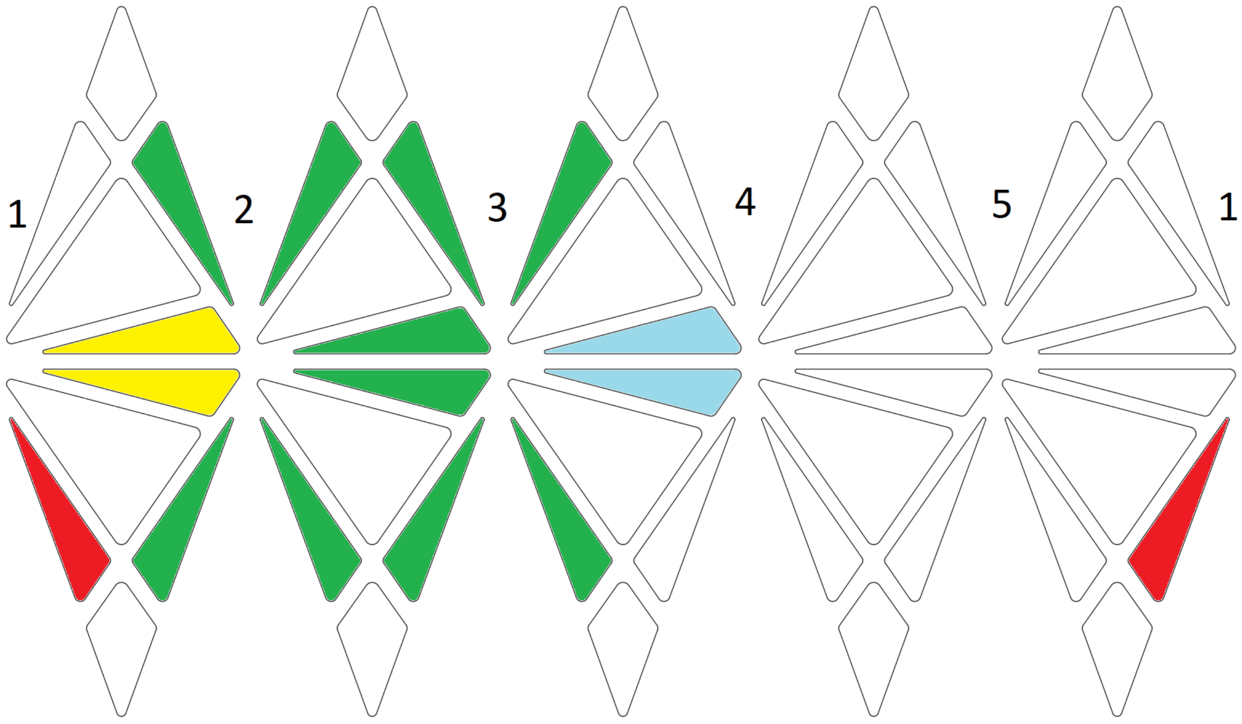
Location 14

$E3_1^{1,1}$ $5''3''2''1'2''13''5''$



Location 15

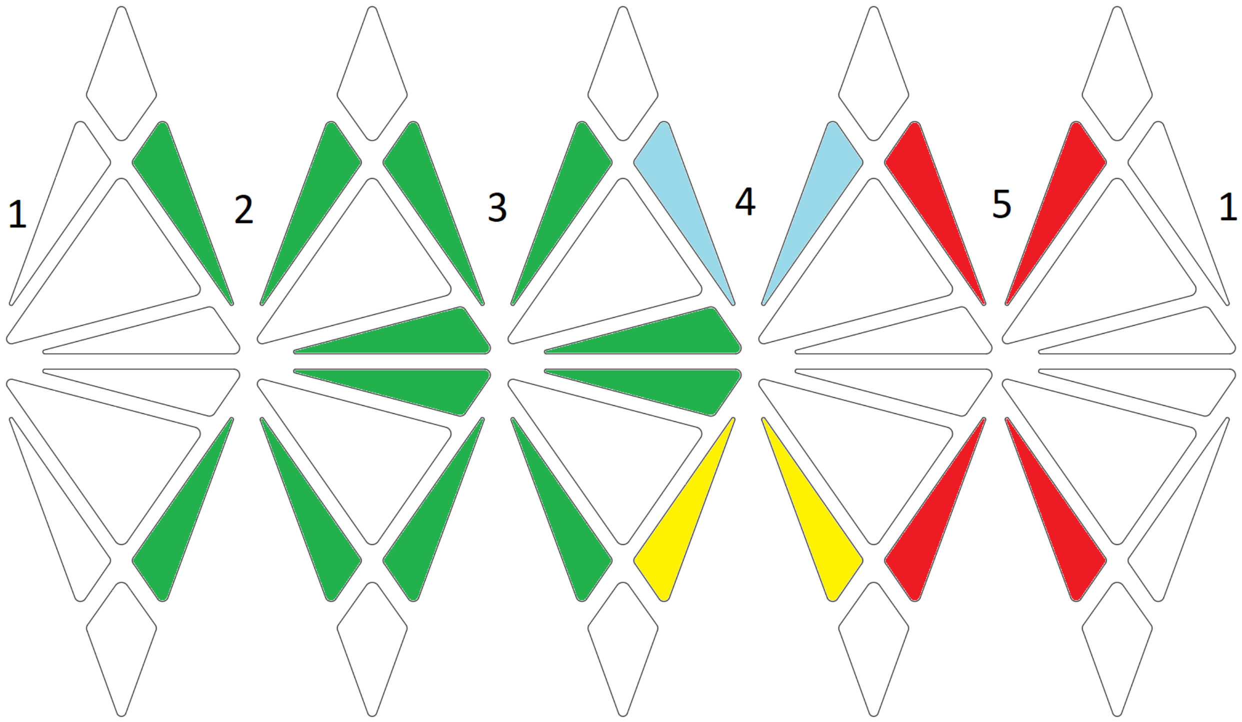
$E3_1^{1,1}$ $5''3''1'2''12''3''5''$



Permuting the Seventh Edge

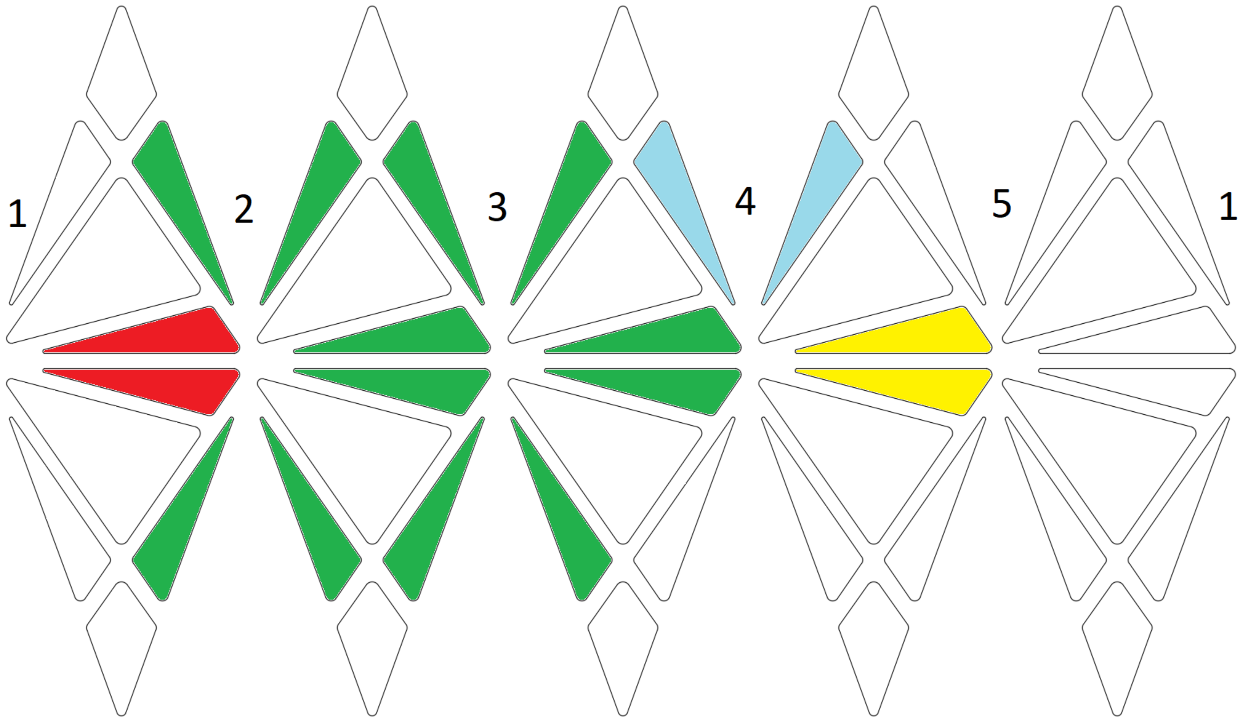
Location 8

$G2_4^5G2_5^4$ $5''4''5''4''5''4''$



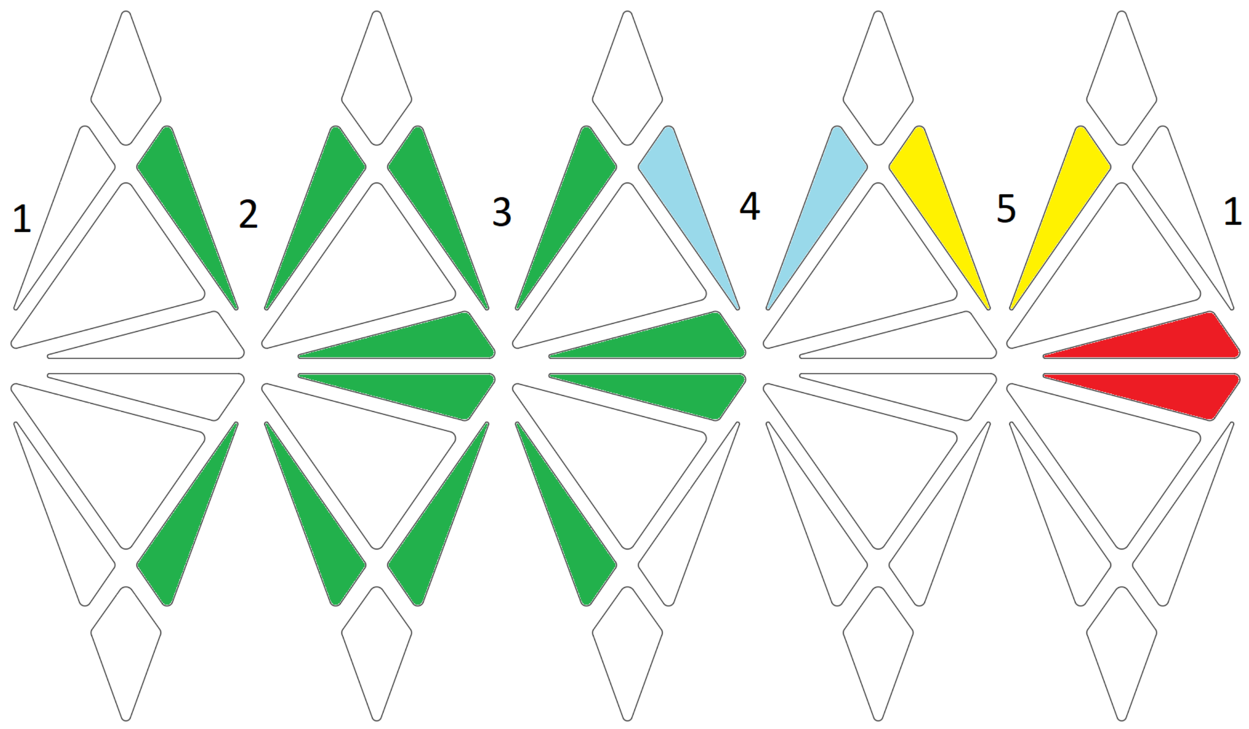
Location 9

$E3_4^{-1,1}$ $3''1''5''45''4'1''3''$



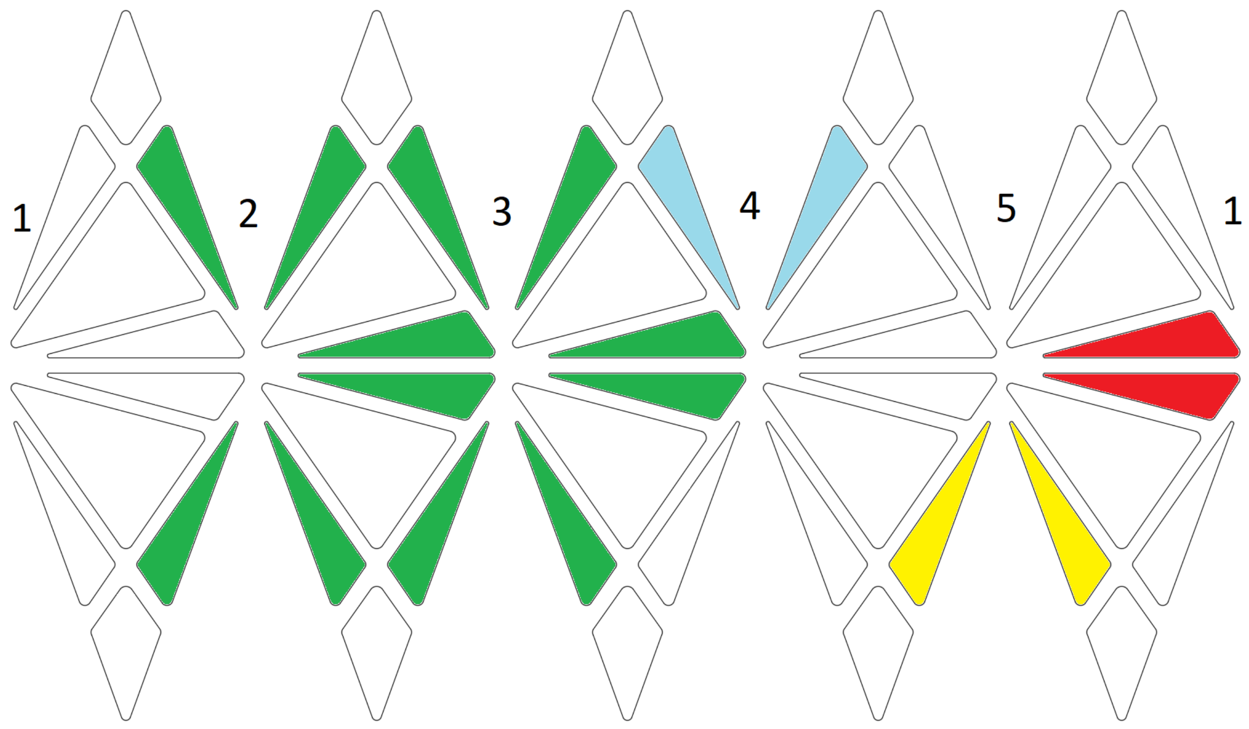
Location 10

$G4_4^{-1}E3_5^{-1,1}G4_4^{1}$ $3''4'3''4''2''1''51''5'2''4''3''43''$



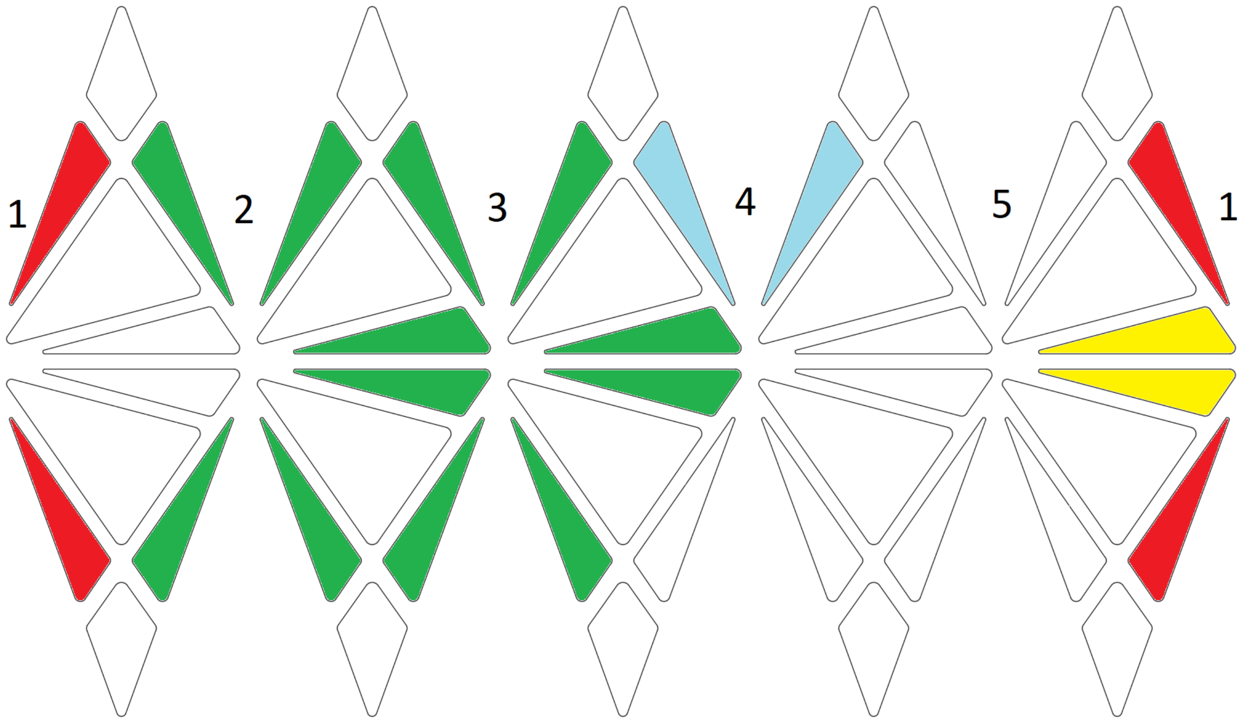
Location 11

$G4_4^{-1}E3_5^{1,1}G4_4^{1}$ $3''4'3''4''2''1''5'1''52''4''3''43''$



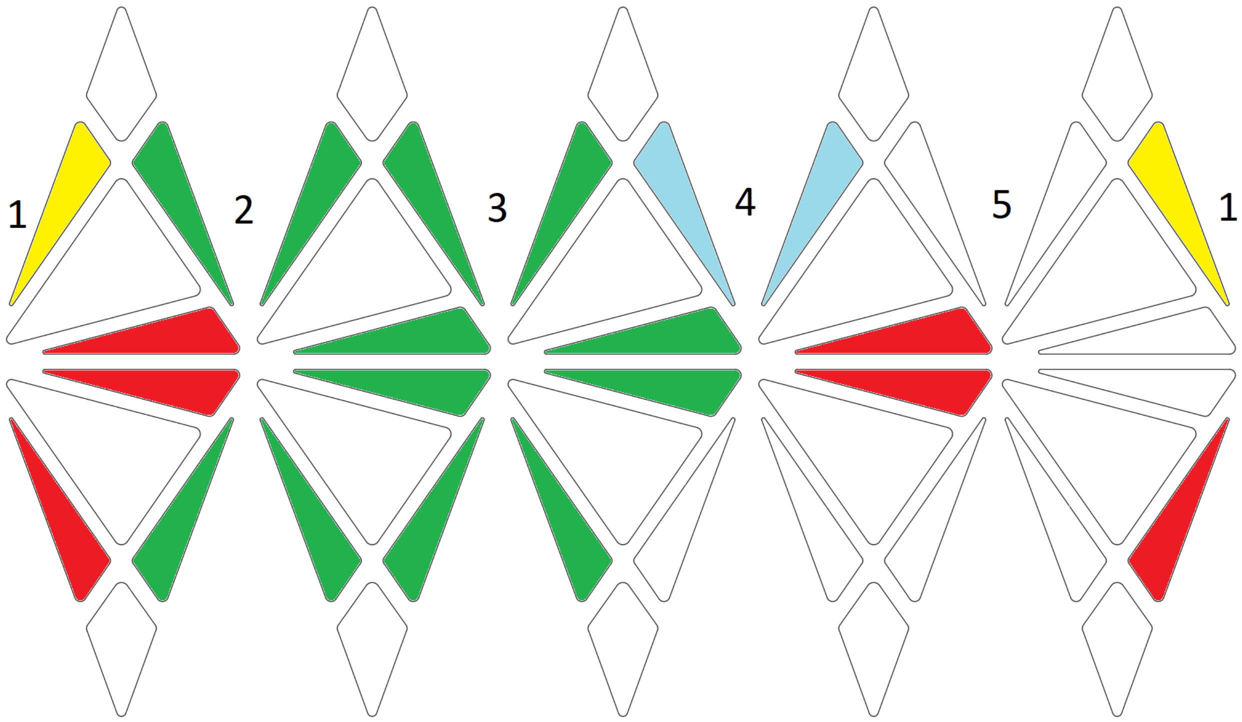
Location 12

$G4_4^{-1}G2_1^2G4_4^1$ $3''4'3''2''1''2''3''43''$



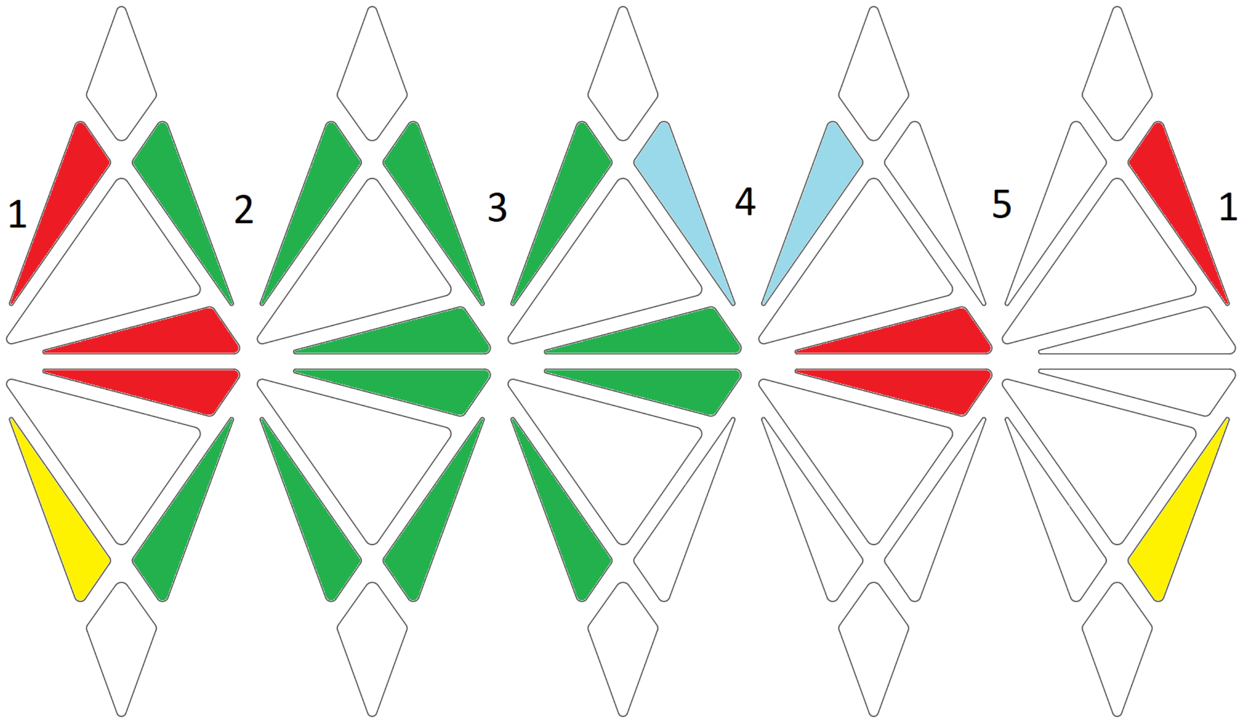
Location 13

$G4_4^1G4_1^{-1}G4_4^{-1}$ $3''43''5''1'5''3''4'3''$



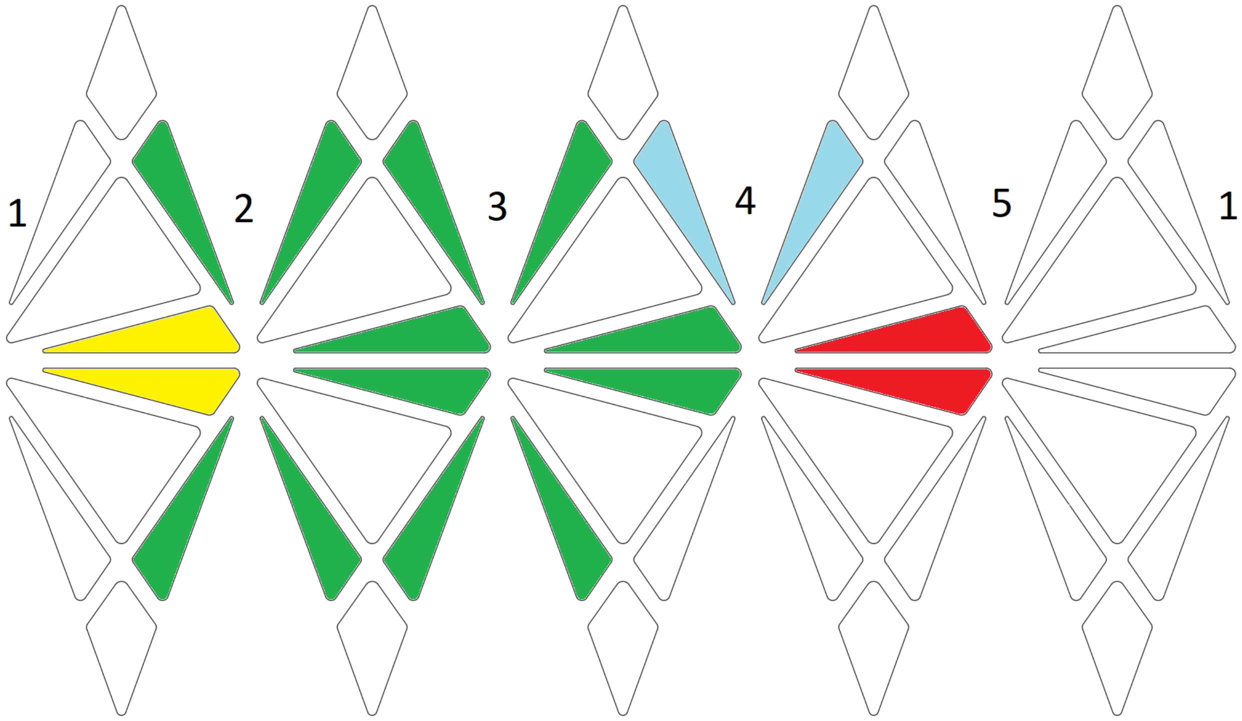
Location 14

$G4_4^1G4_1^1G4_4^{-1}$ $3''43''5''15''3''4'3''$



Location 15

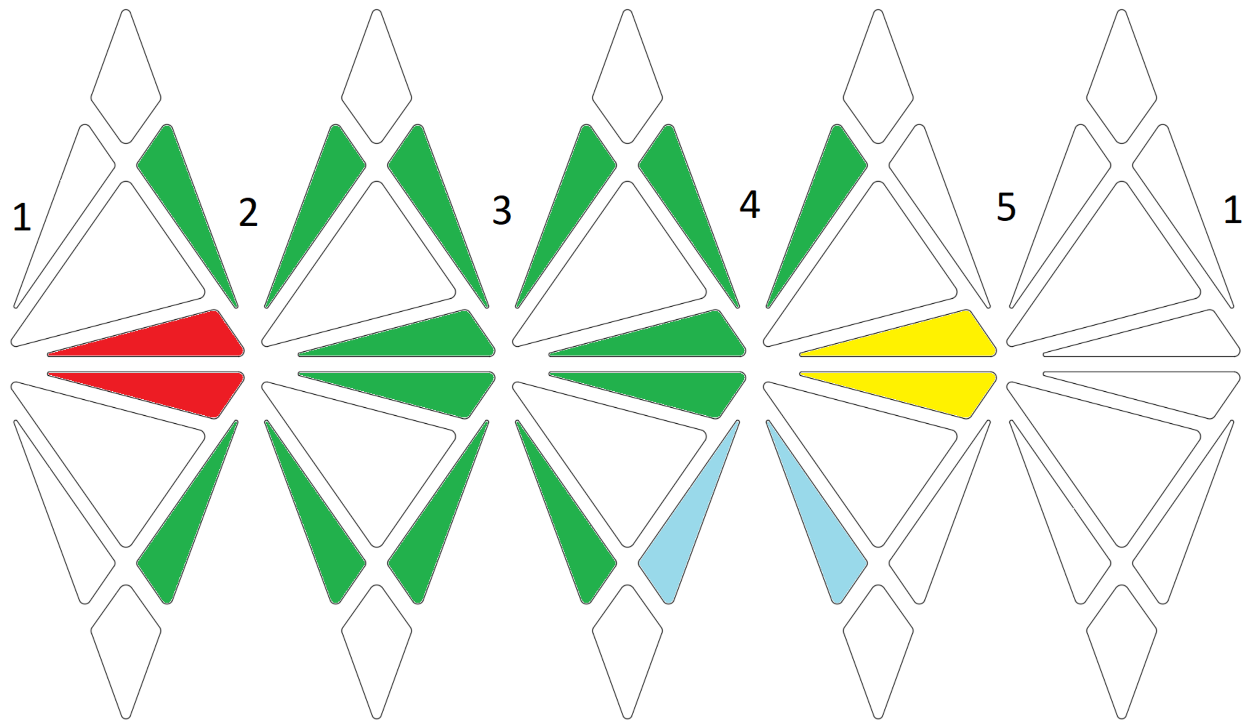
$E3_4^{-1,-1}$ $3''1''45''4'5''1''3''$



Permuting the Eighth Edge

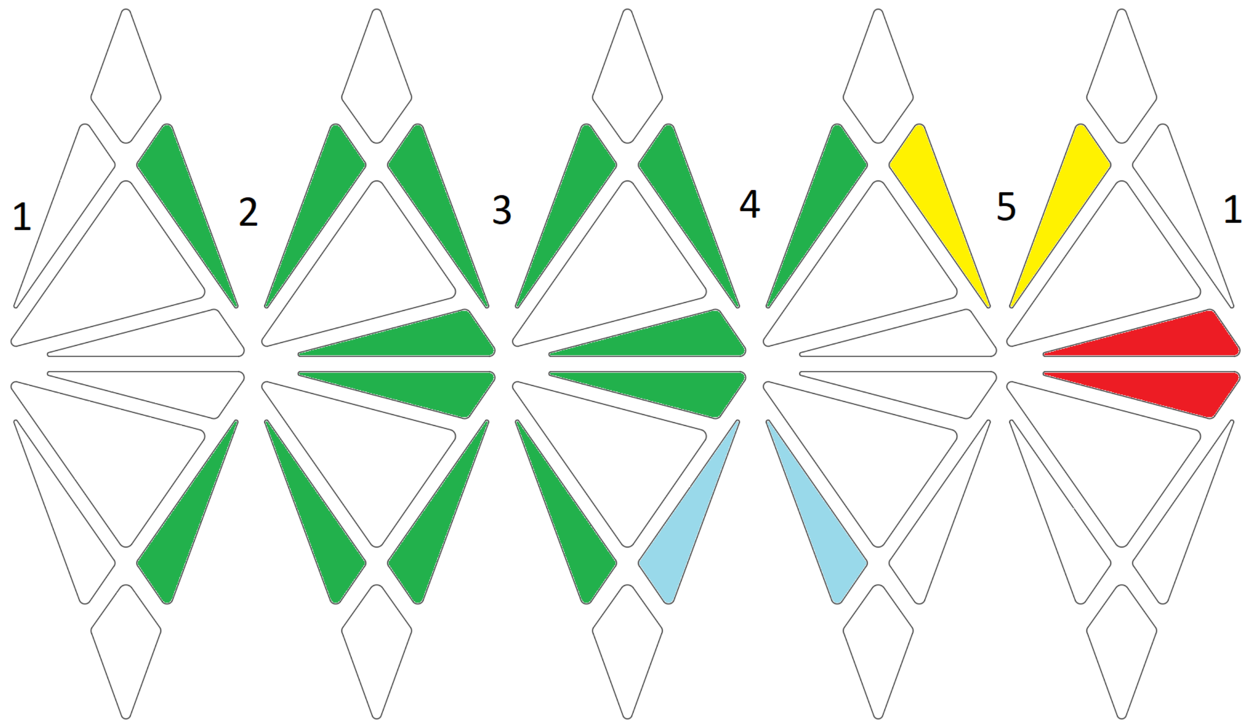
Location 9

$E3_4^{1,1}$ $3''1''5''4'5''41''3''$



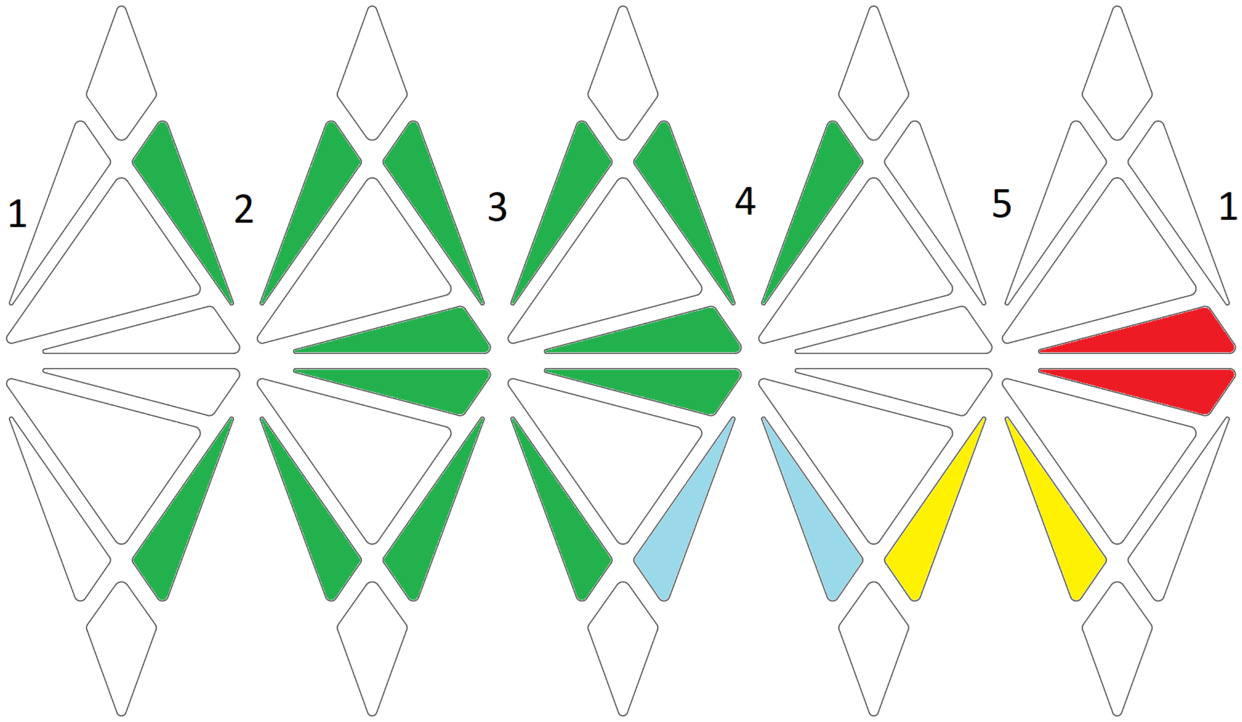
Location 10

$G4_4^1E3_5^{-1,1}G4_4^{-1}$ $3''43''4''2''1''51''5'2''4''3''4'3''$



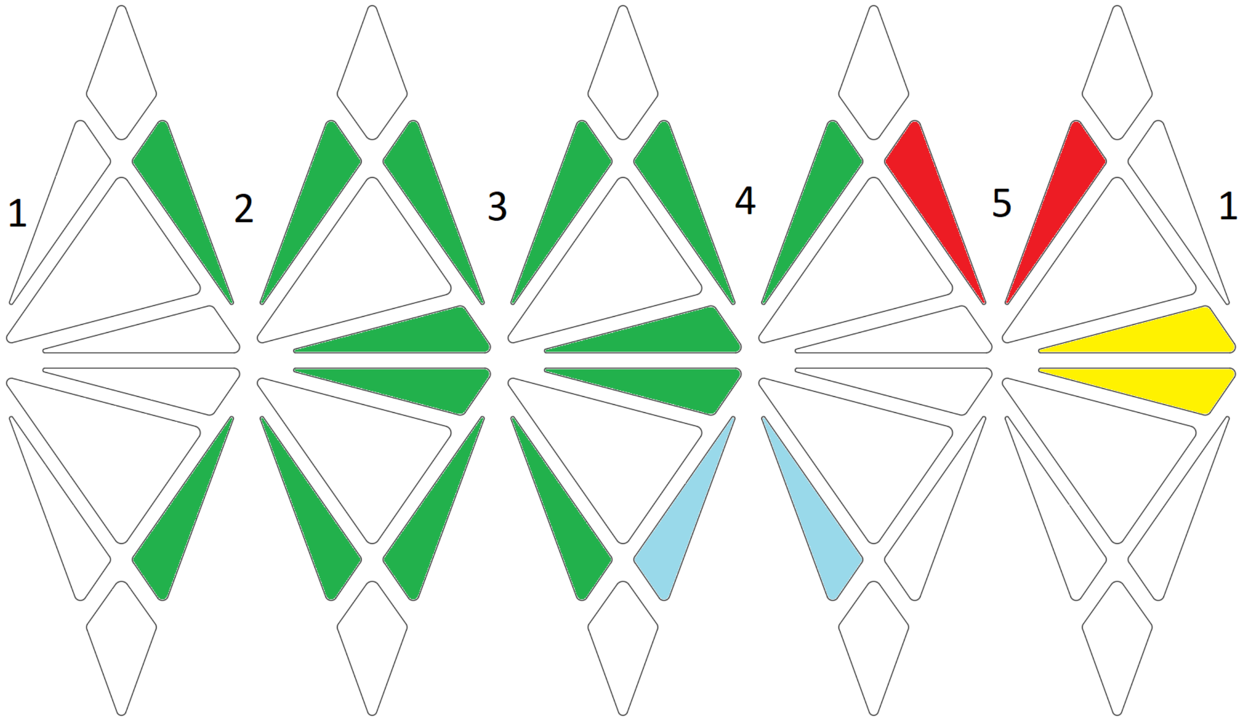
Location 11

$G4_4^1 E3_5^{1,1} G4_4^{-1}$ $3''43''4''2''1''5'1''52''4''3''4'3''$



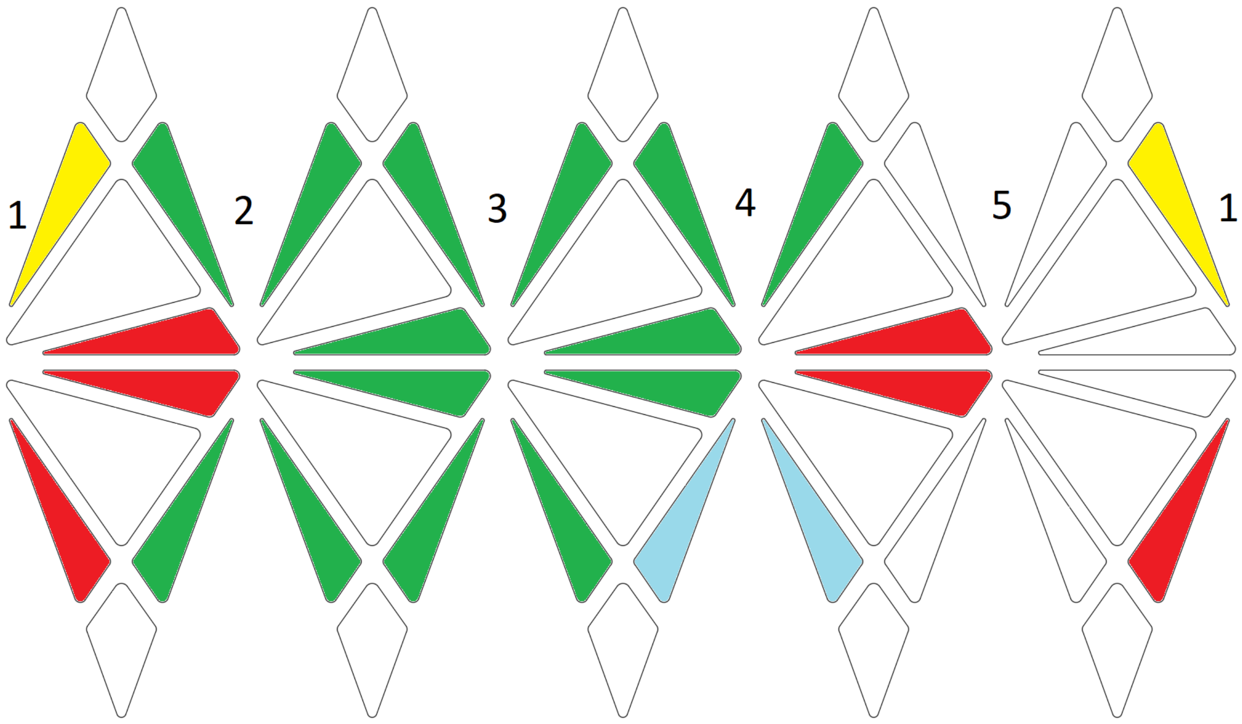
Location 12

$G4_4^1 E3_5^{-1,-1} G4_4^{-1}$ $3''43''4''2''51''5'1''2''4''3''4'3''$



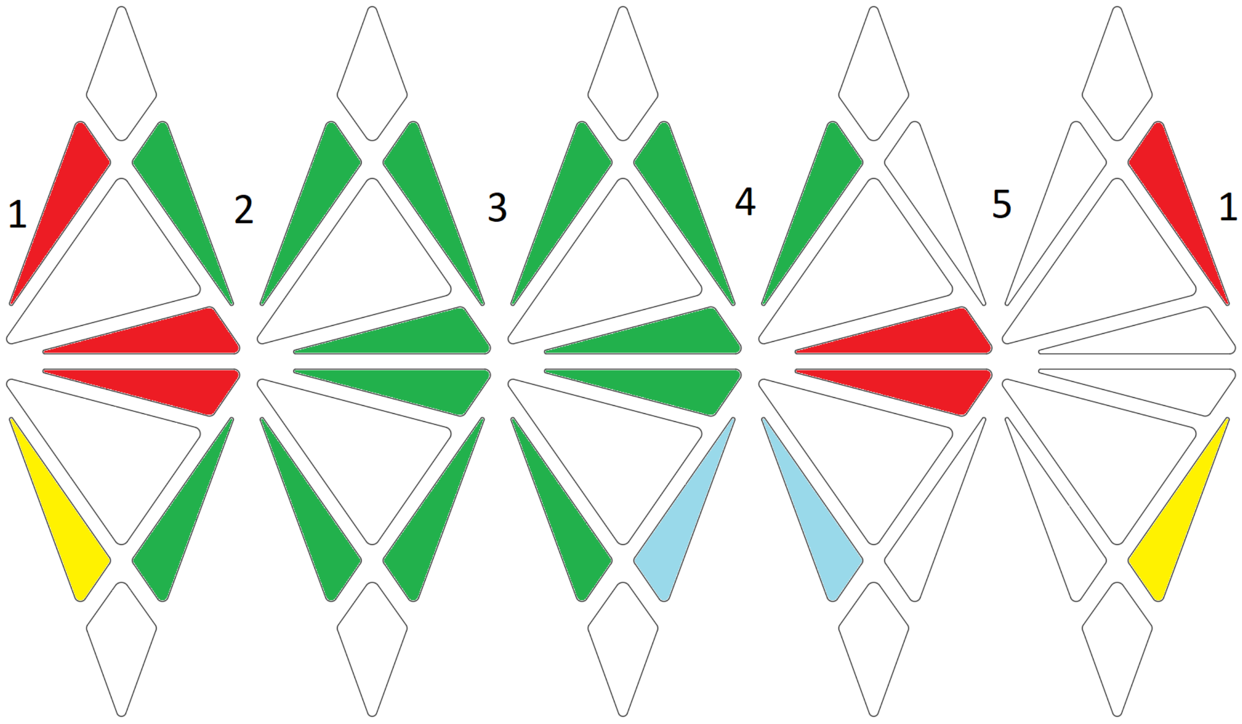
Location 13

$G4_1^{-1}E3_4^{1,1}$ $5''1'5''3''1''5''4'5''41''3''$



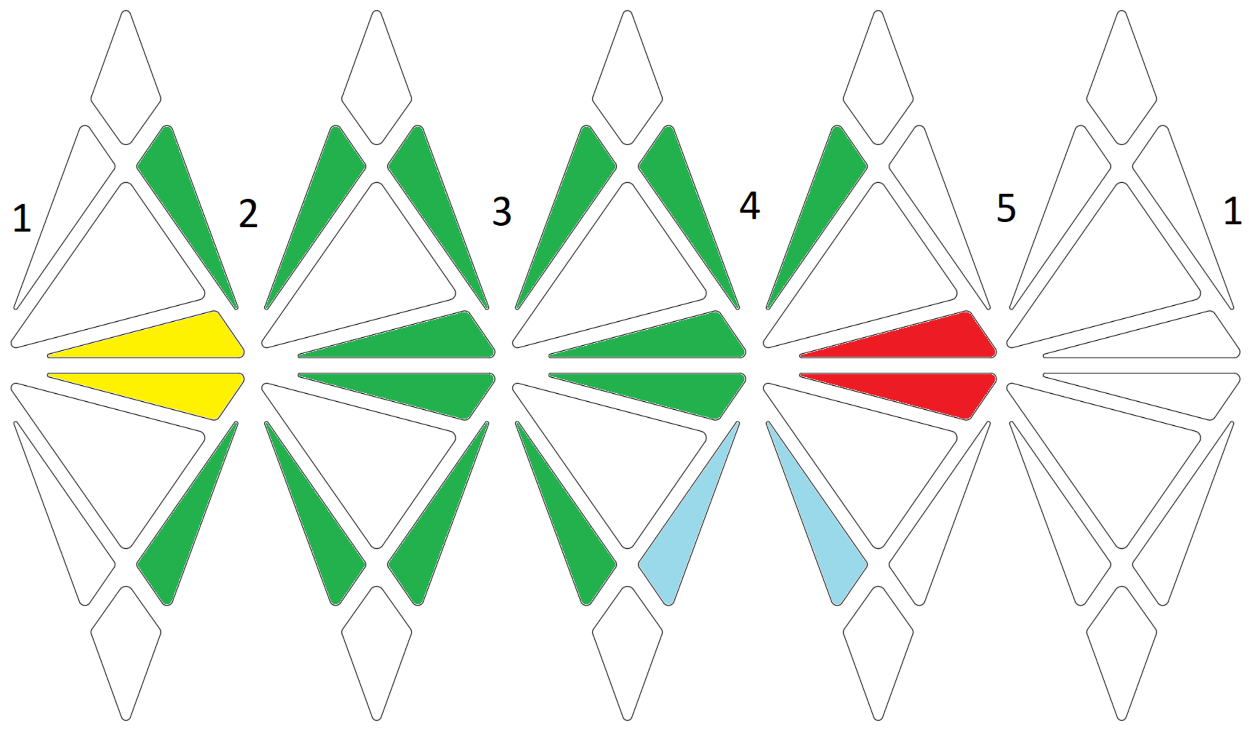
Location 14

$G4_1^{1}E3_4^{1,1}$ $5''15''3''1''5''4'5''41''3''$



Location 15

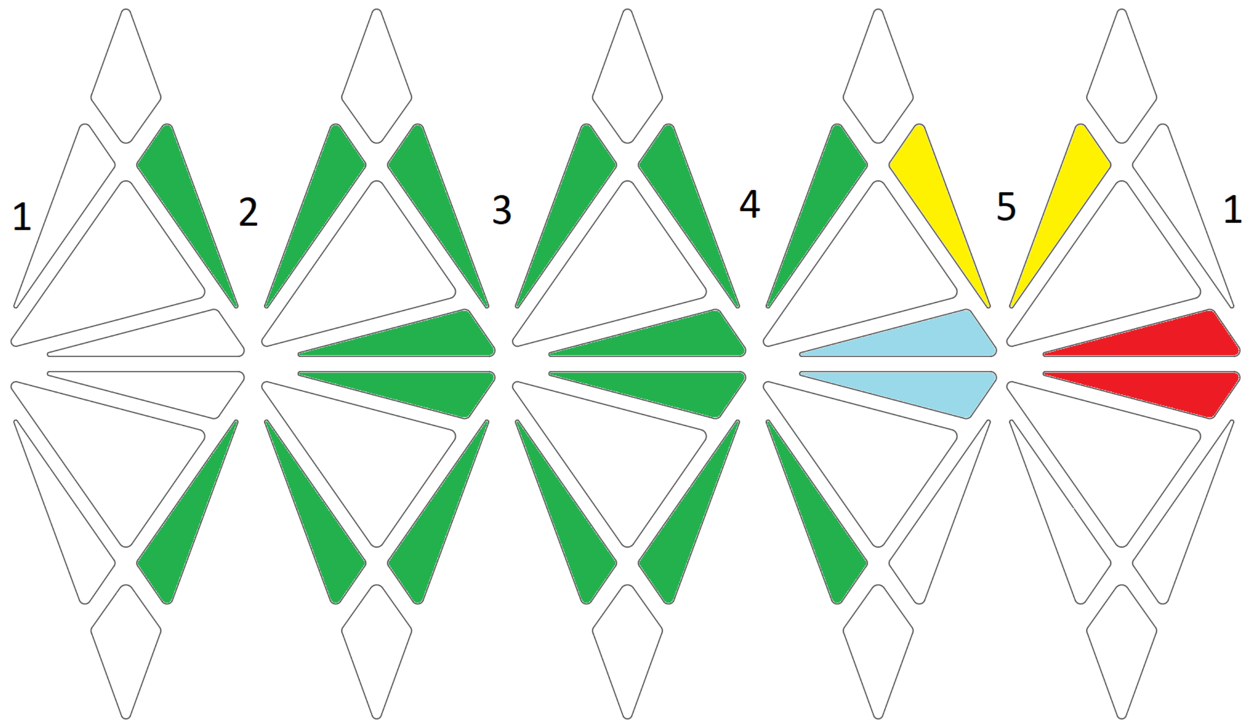
$E3_4^{1,-1}$ $3''1''4'5''45''1''3''$



Permuting the Nineth Edge

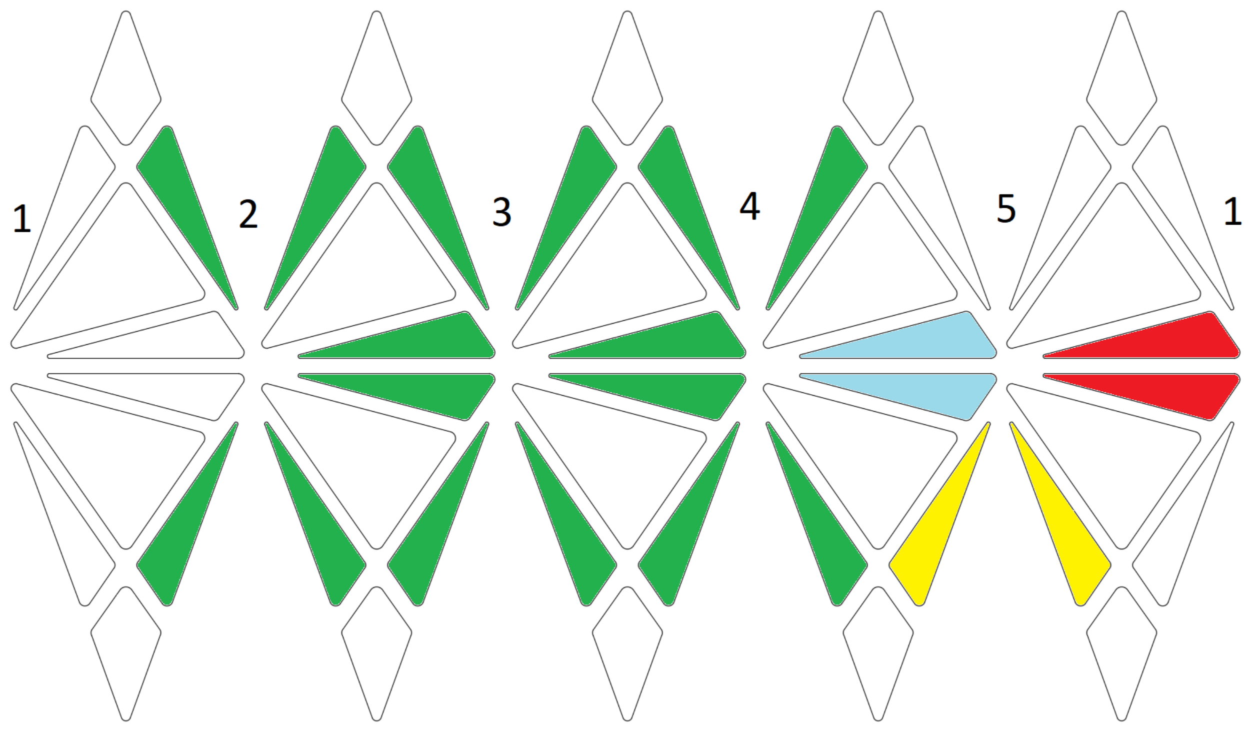
Location 10

$G2_3^4E3_5^{-1,1}G2_3^4$ $4''3''2''1''51''5'2''3''4''$



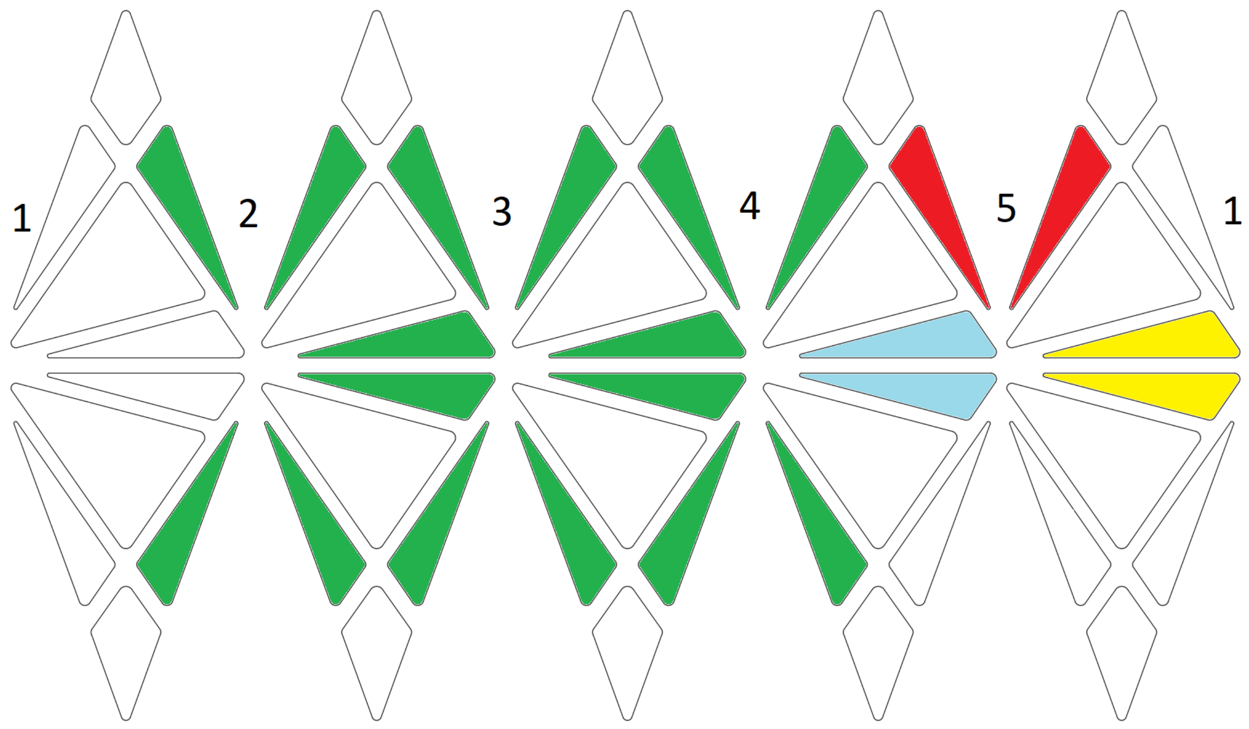
Location 11

$G2_3^4 E3_5^{1,1} G2_3^4$ $4''3''2''1''5'1''52''3''4''$



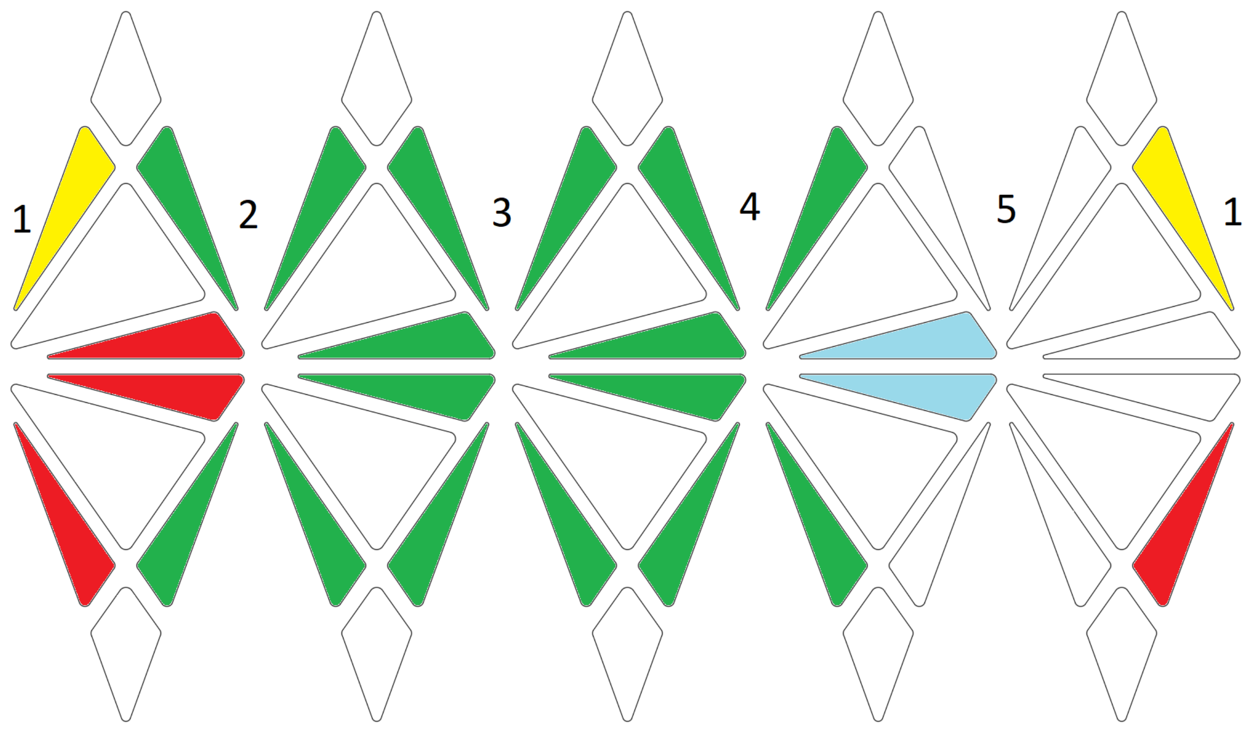
Location 12

$G2_3^4 E3_5^{-1,-1} G2_3^4$ $4''3''2''51''5'1''2''3''4''$



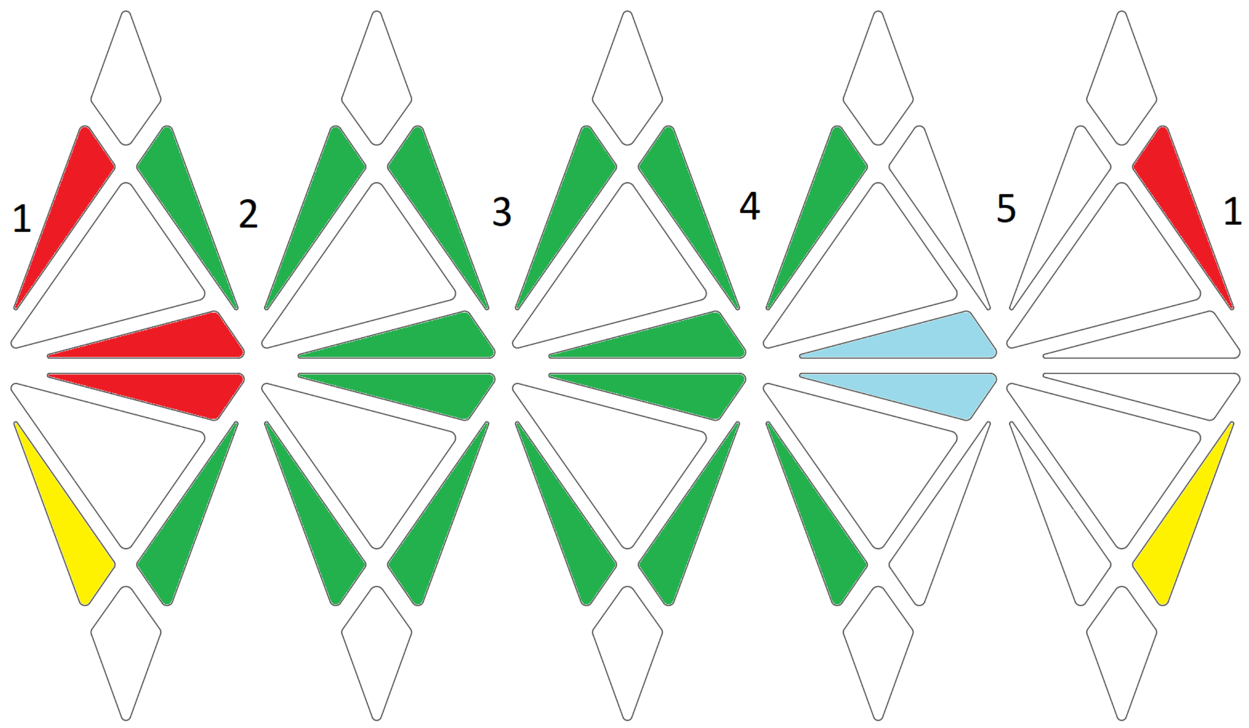
Location 13

$G4_1^{-1}$ $5''1'5''$



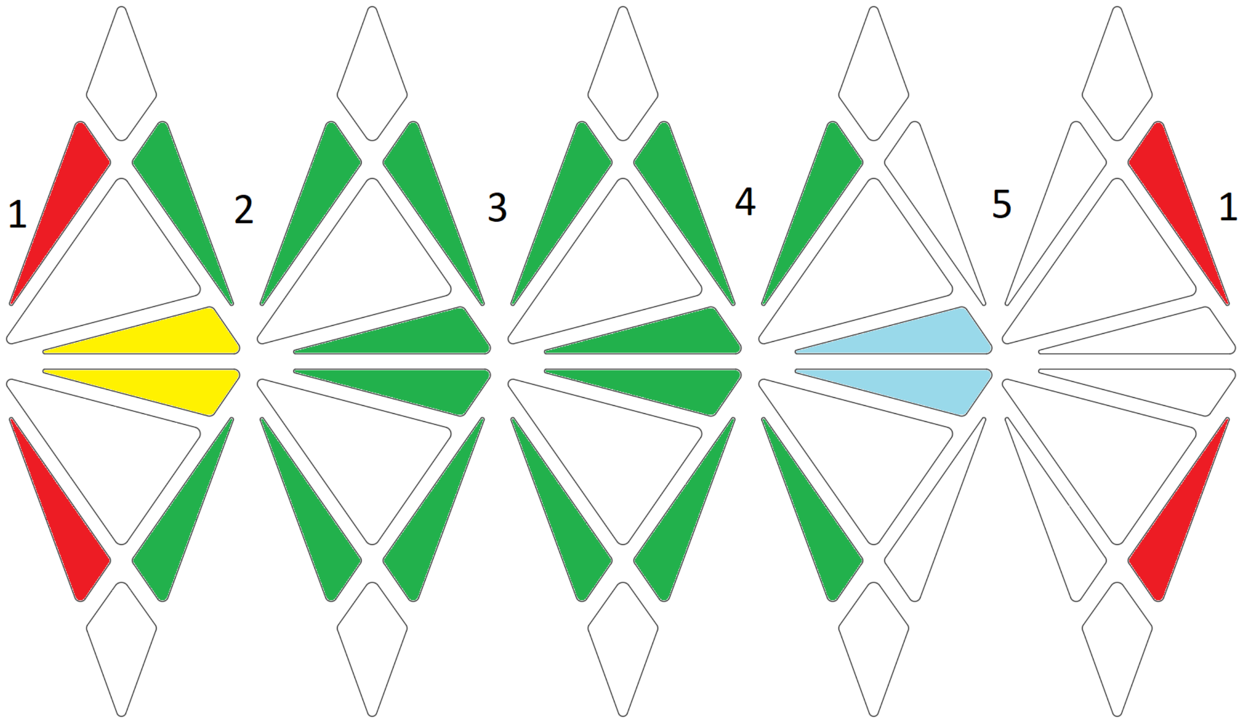
Location 14

$G4_1^{1}$ $5''15''$



Location 15

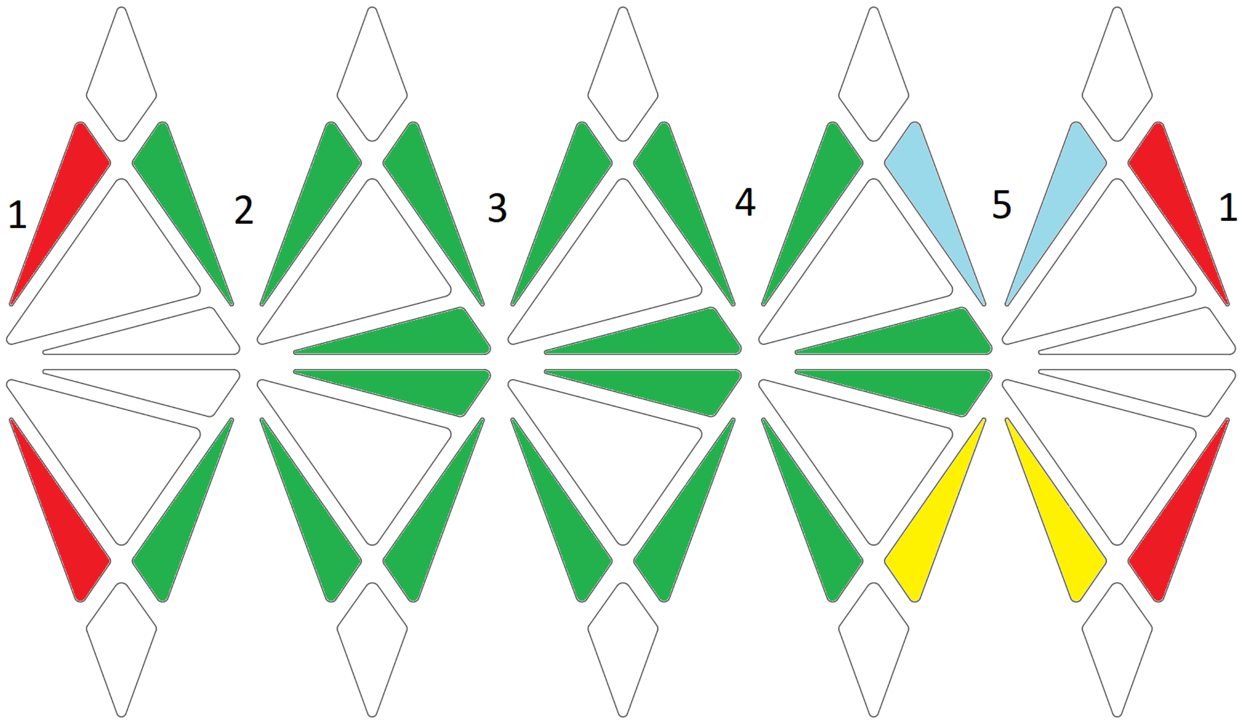
$G4_1^2$ $5''1''5''$



Permuting the Tenth Edge

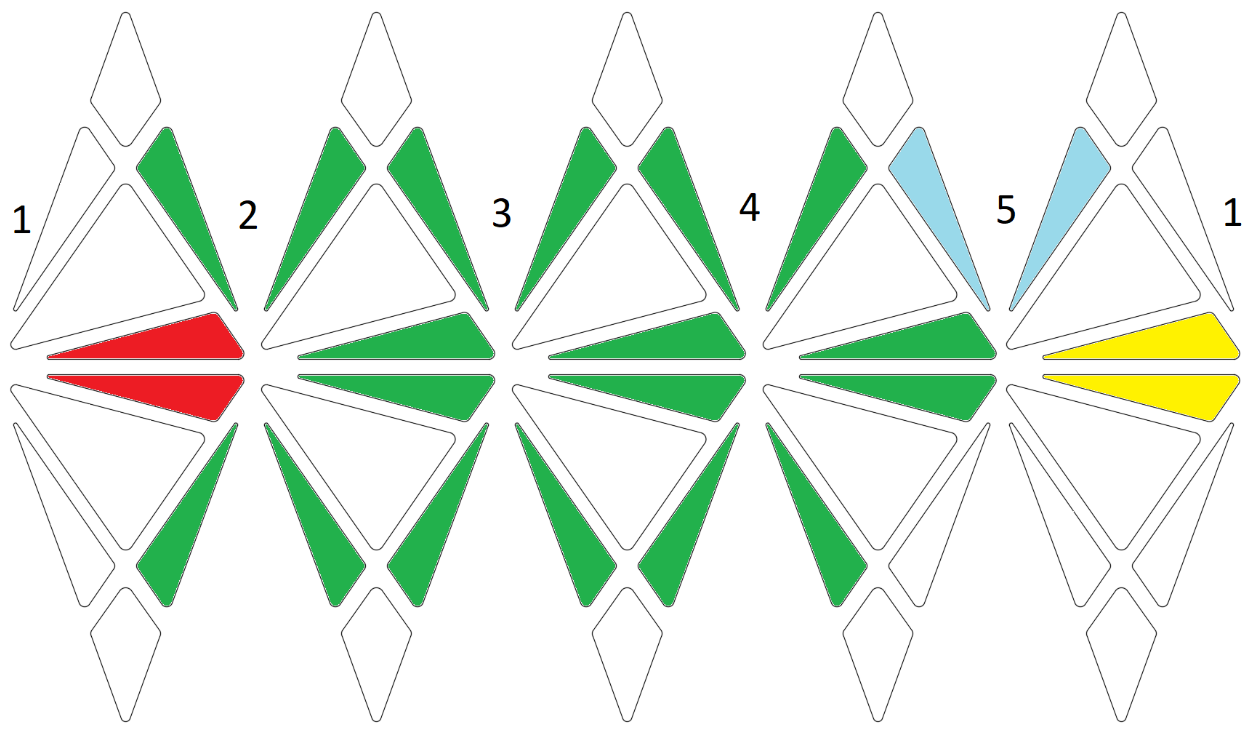
Location 11

$G2_1^5G2_5^1$ $5''1''5''1''5''1''$



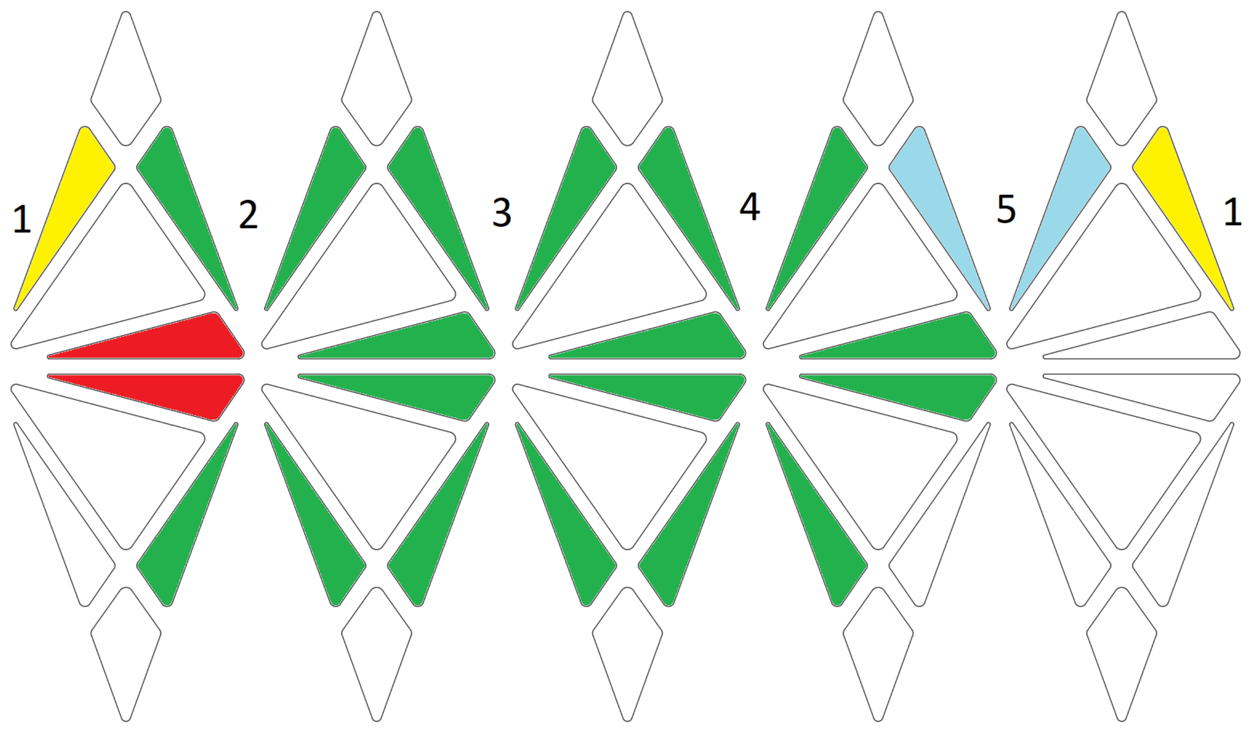
Location 12

$G4_3^2 G4_5^{-1} G4_3^2$ $2''3''2''4''5'4''2''3''2''$



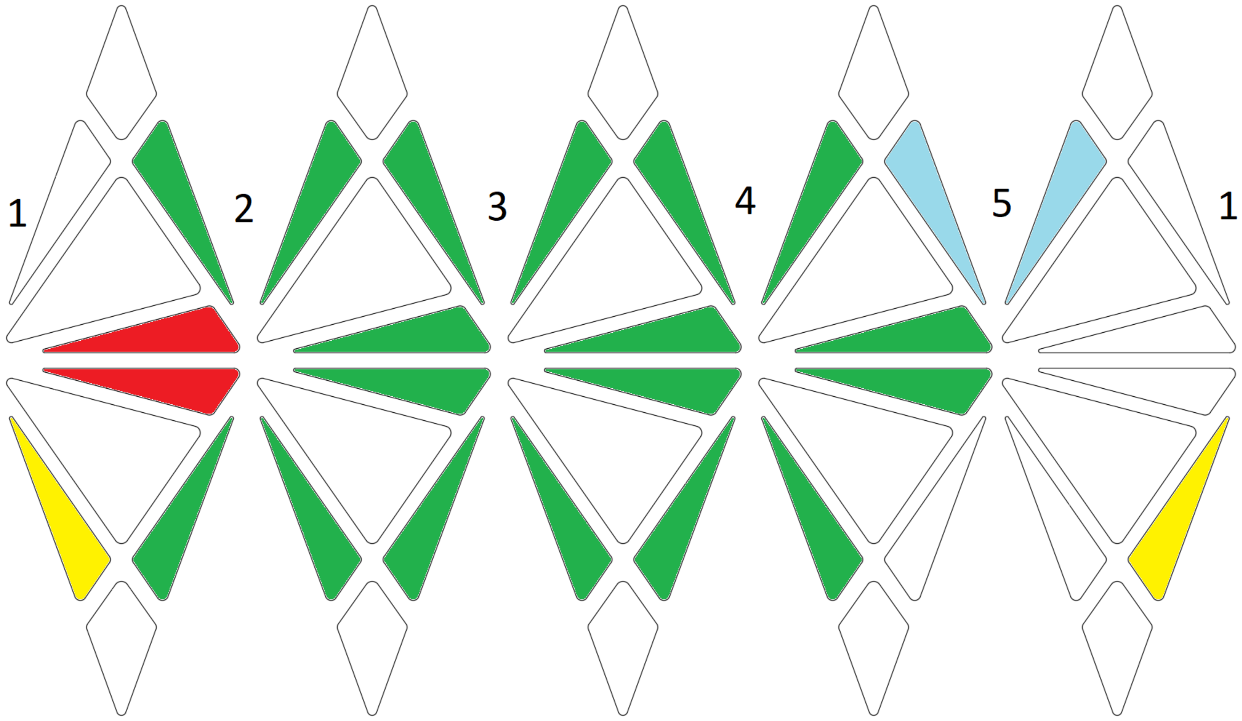
Location 13

$G4_5^{-1} E3_1^{-1,1} G4_5^1$ $4''5'4''5''3''2''12''1'3''5''4''54''$



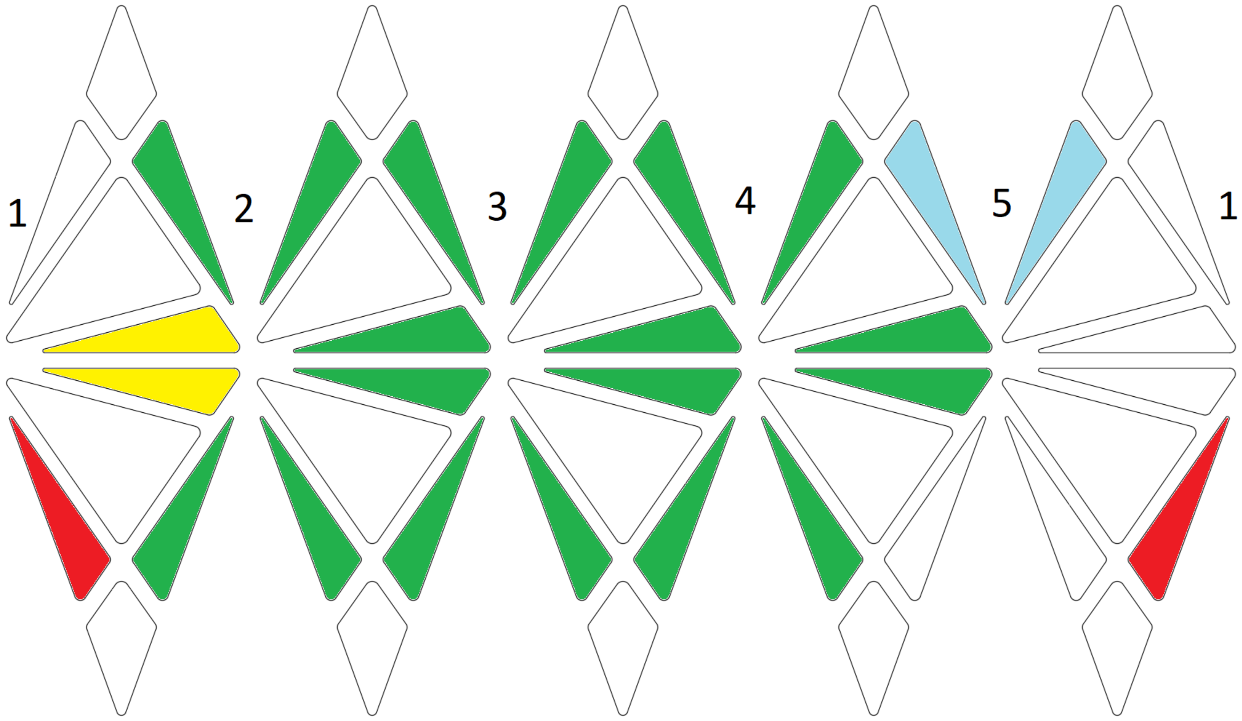
Location 14

$G4_5^{-1}E3_1^{1,1}G4_5^1$ $4''5'4''5''3''2''1'2''13''5''4''54''$



Location 15

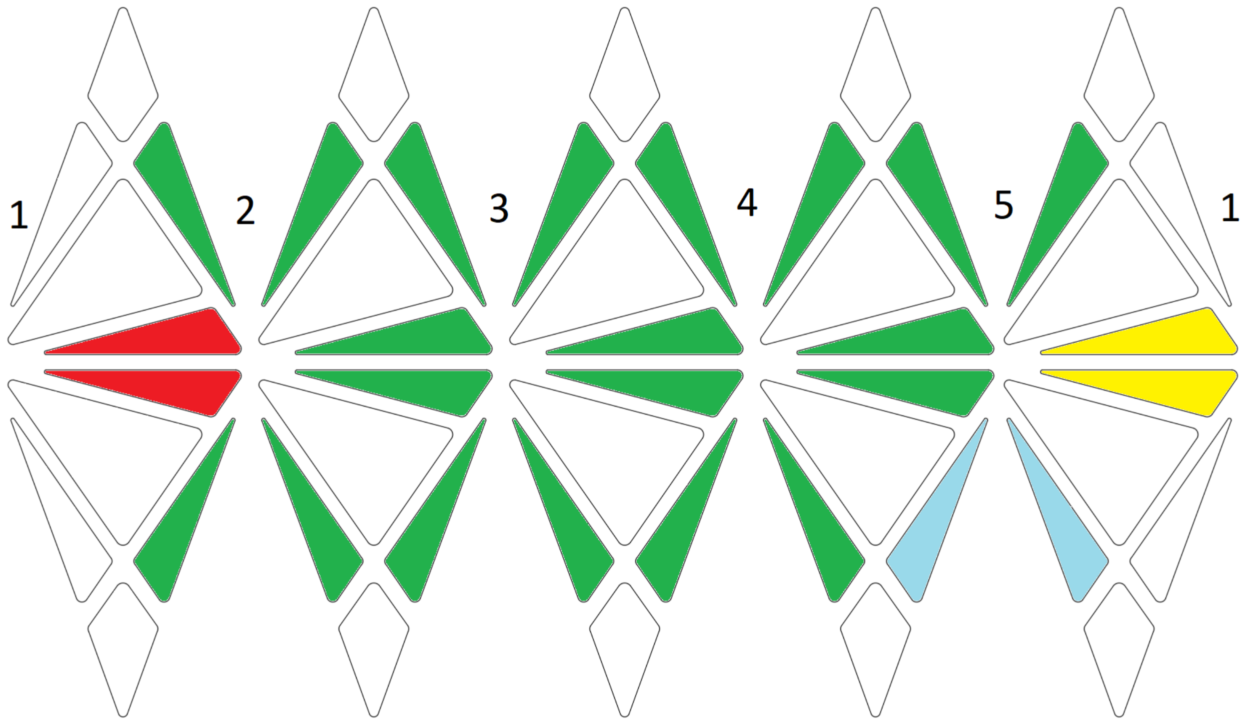
$G4_5^{-1}E3_1^{1,-1}G4_5^1$ $4''5'4''5''3''1'2''12''3''5''4''54''$



Permuting the Eleventh Edge

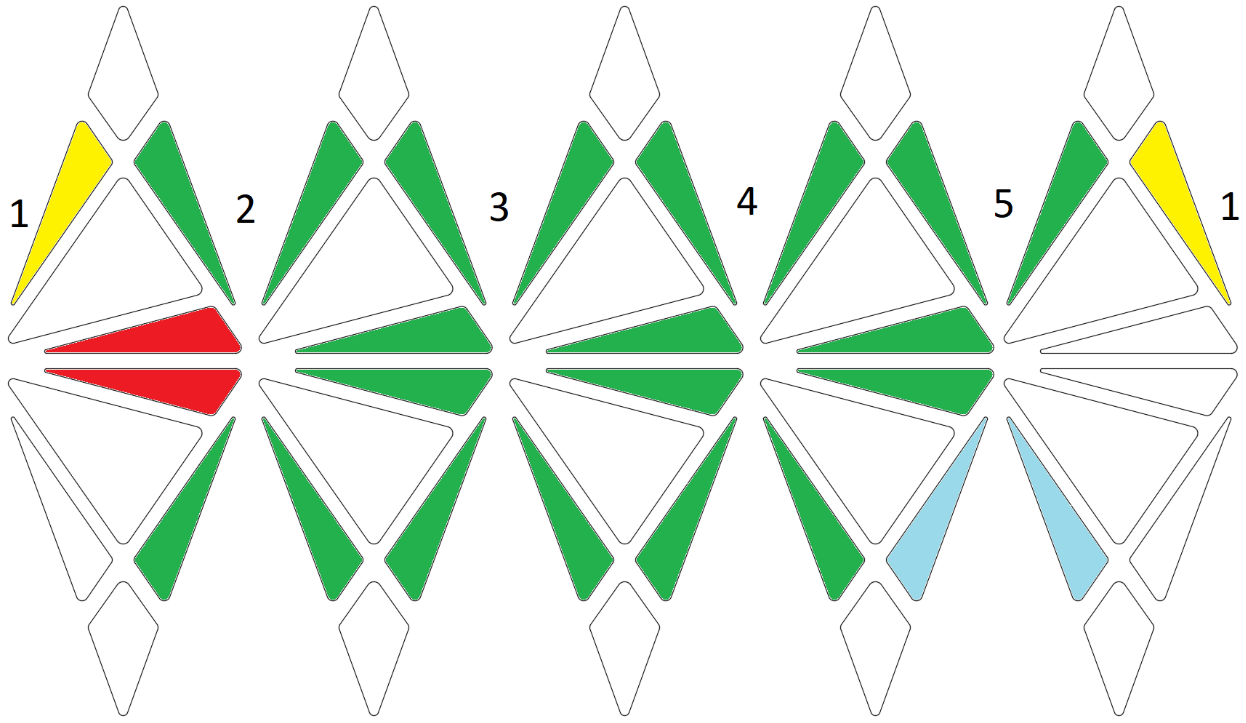
Location 12

$G2_1^5 G2_3^4 E3_5^{1,1} G2_3^4 G2_1^5$ $5''1''5''4''3''2''1''5'1''52''3''4''5''1''5''$



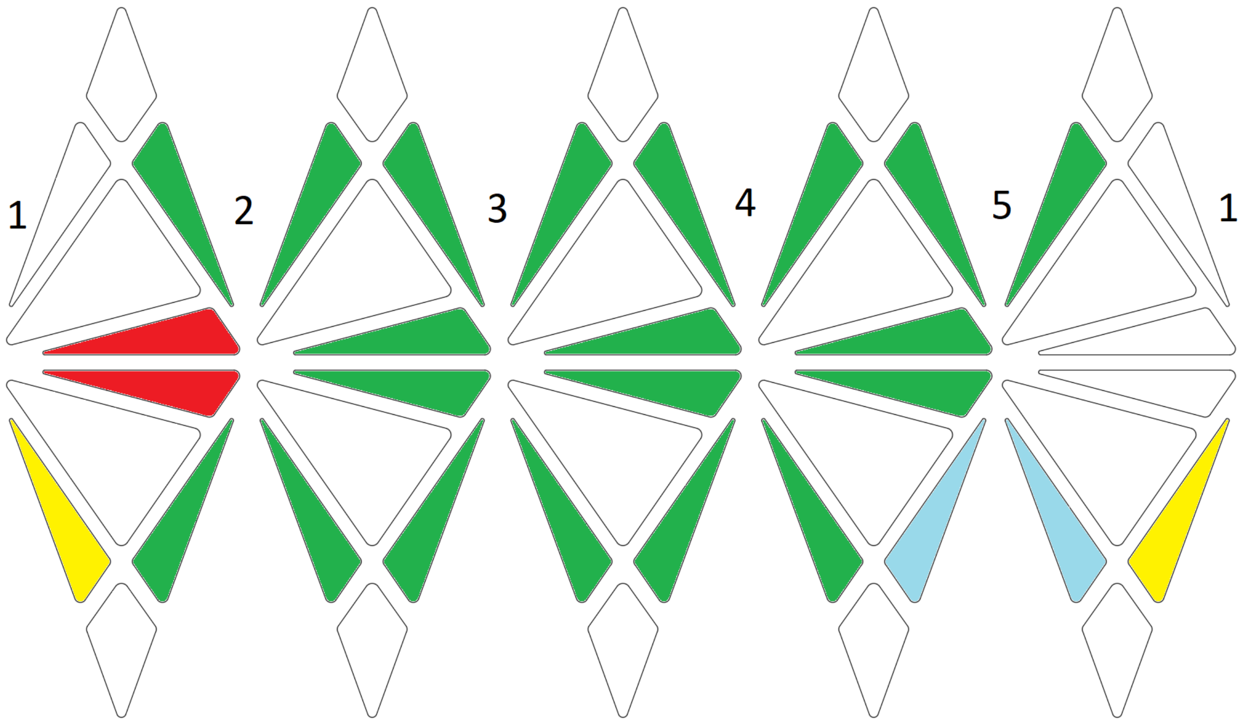
Location 13

$G4_5^1 E3_1^{-1,1} G4_5^{-1}$ $4''54''5''3''2''12''1'3''5''4''5'4''$



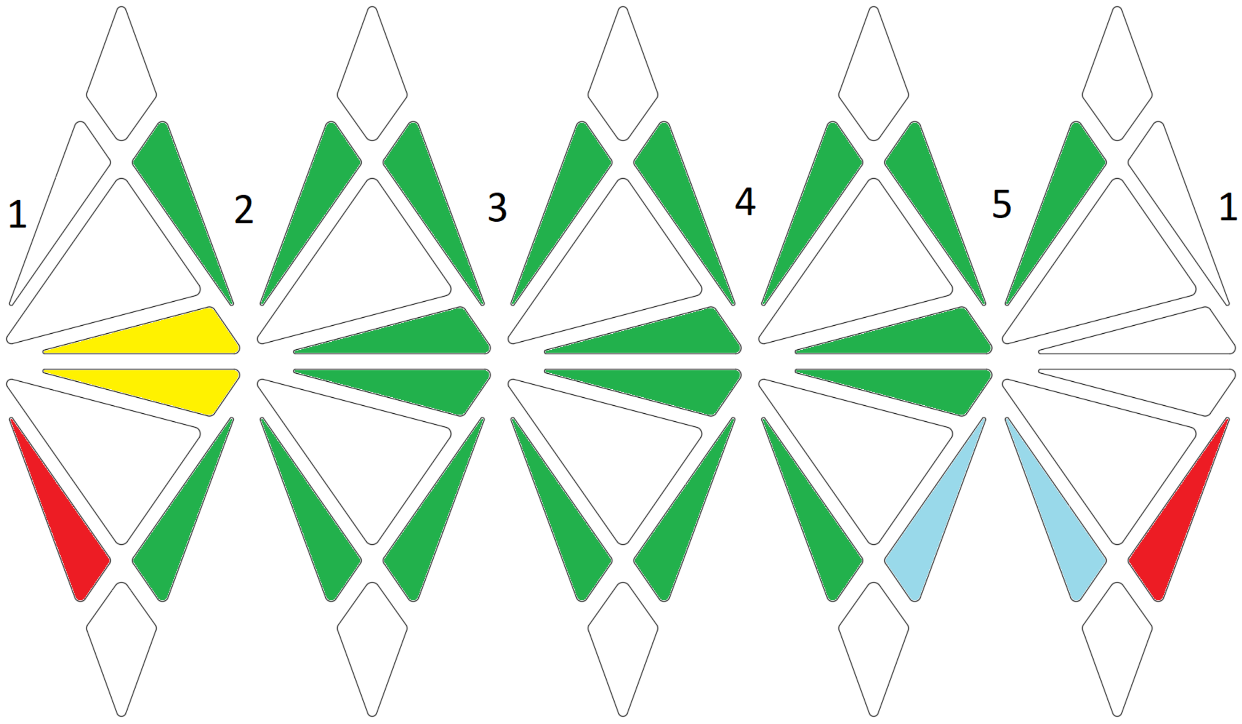
Location 14

$G4_5^1 E3_1^{1,1} G4_5^{-1}$ $4''54''5''3''2''1'2''13''5''4''5'4''$



Location 15

$G4_5^1 E3_1^{1,-1} G4_5^{-1}$ $4''54''5''3''1'2''12''3''5''4''5'4''$



Appendix 2: Vocabulary

Graph Theory

Graph
https://en.wikipedia.org/wiki/Graph_(discrete_mathematics)

Loop
https://en.wikipedia.org/wiki/Loop_(graph_theory)

Radius
https://en.wikipedia.org/wiki/Distance_(graph_theory)#Related_concepts

Vertex
https://en.wikipedia.org/wiki/Vertex_(graph_theory)

Degree
https://en.wikipedia.org/wiki/Degree_(graph_theory)

Statistics

Combination
https://en.wikipedia.org/wiki/Combination

! - Factorial
https://en.wikipedia.org/wiki/Factorial

Set Theory

ℤ - Set of Integers
https://en.wikipedia.org/wiki/Integer

| - Such That
https://en.wikipedia.org/wiki/Set-builder_notation

∈ *- Is a Member of*
https://en.wikipedia.org/wiki/Element_(mathematics)#Notation_and_terminology

≡ *- Congruent*
https://en.wikipedia.org/wiki/Modular_arithmetic

Arithmetic Underflow/Overflow
https://en.wikipedia.org/wiki/Arithmetic_underflow

"Twisty Puzzle" Terminology

God's Algorithm
https://en.wikipedia.org/wiki/God%27s_algorithm

Devil's Algorithm
https://getgocube.com/play/devils-number/

God's Number
https://en.wikipedia.org/wiki/God%27s_algorithm#Solution

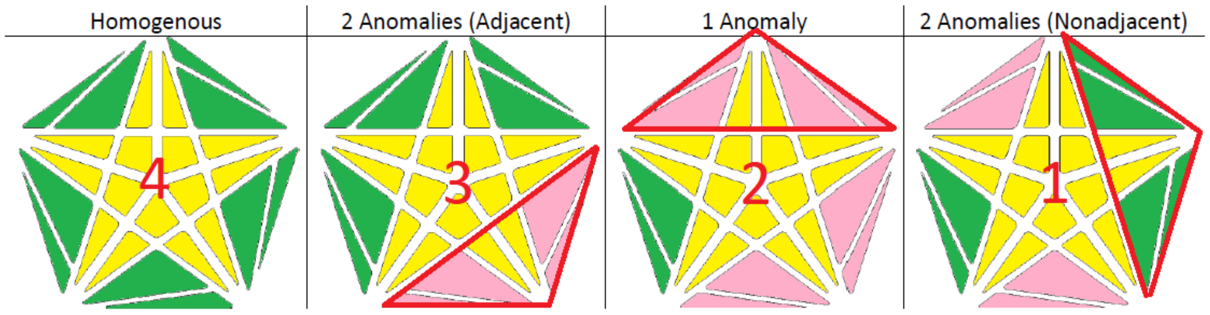
Appendix 3: Condensed Solution Guide

Step 1: Polar Orientation

1. Identify how many of the 5 axes can be turned.
    a) If 5 axes can be turned, proceed to Step 2: Equatorial Orientation.
    b) If only 3 axes can be turned, identify the axis located between the two axes that can't turn. Rotate the Identified axis by 90-degrees (either direction).
    c) If only 2 axes can be turned, rotate the two axes that can be turned by 90-degrees each (either direction).

Step 2: Equatorial Orientation

2. Without spinning any axes, reorient the puzzle as a whole until it looks like one of the following 4 projections from Figure 14 The Four Unique Equatorial Orientations:



    a) If your puzzle matches projection #1, rotate the right most axis by 180-degrees to reorient your puzzle to projection #2
    b) If your puzzle matches projection #2, rotate the top axis by 180-degrees to reorient your puzzle to projection #3
    c) If your puzzle matches projection #3, rotate the bottom right axis by 180-degrees to reorient your puzzle to projection #4
    d) If your puzzle matches projection #4, proceed to Step 3: Edge Permutation

Step 3: Edge Permutation

The following steps use the previously defined Notation.

3. Iteratively permute the edges using the graphical references provided in Appendix 1: Edge Permutations - Graphical.

Step 4: Face Permutation

4. Iteratively permute the faces using the graphical references provided in Figure 43 Labeling of All Faces and the corresponding algorithms in Table 15 All Face 3-Cycles, Efficient and with Inverses.

Appendix 4: Revision History

This paper is intended to be a living document. Updates, corrections, and expansions are expected. If you would like to help, please submit suggestions and contributions to:

Email:	ArtisanPuzzle@gmail.com
Subject Line:	Fracture-10 Manual v0.4 - Suggestion

Version History

Version 0.4 – 12/21

This is the initial public release. While incomplete, this paper has been released to coincide with the 13th day of the 2021 Puzzle Advent Calendar.

Errata

As this is the first version released, no errata has been discovered yet.

Contributors

Tanner Frisby	Primary Author