Let's Make a $(3^6)_D$ $(3^6)_L$ Chiral Tessellation Dance

Joseph D. Clinton
Retired
334 Mackenzie Drive
West Chester, PA, 19380-3816, USA
E-mail: Jclinton@kennett.net

Abstract

Mathematics is an abstract language that often attempts to explain real world observations. Generalizations often emerge that become useful in explaining the physical world around us. Too often the abstract concepts are assumed to give precise solutions to physical realities. Hopefully, this workshop will give educators and their students' hands-on experience to understand the differences between theoretical abstraction and the reality of applying physical restraints. Each participant in the workshop will receive a modeling kit to assemble and take with her/him. The model will illustrate the physical realities of applying the mathematical principles of rotation and translation transformations of a linkage of triangles from one symmetry form to another. Each participant will also receive an animated film that will illustrate the theoretical application of the same mathematical abstractions.

Introduction

Mathematics is one of the many areas of study that bridge all other areas of study and discovery. At one time it had become a universal language crossing the boundaries between the arts, sciences, humanities, technologies and other areas of specialization. However, in recent years, the study of mathematics has separated itself into many dialects becoming specialized languages of their own. Is the universal bridge of mathematics crumbling?

In the abstract world of mathematics, a point may be described as having zero dimensions. A line may be described as having one dimension only. A plane may be described as having two dimensions only. A solid may be described as having three dimensions. In the physical world, these abstractions cannot fully explain the behavior of the physical world. These definitions must account for the three dimensionality of physical reality to be able to fully describe its observed behavior.

The goal of the workshop is to give each participant hands-on experience to the differences between mathematical abstraction and the physical reality of applying rules of restraint to common materials.

The Workshop

Each participant in the workshop will receive a modeling kit to assemble and take with her/him. The model will illustrate the physical realities of applying the mathematical principles of rotation and translation transformations of a linkage of triangles from one symmetry form to another. Each participant will also receive an animated film that will illustrate the theoretical application of the same mathematical abstractions.

Let's Make a $(3^6)_D$ $(3^6)_L$ Chiral Tessellation Dance, the Physical Model

Required materials. The materials contained in the kit are: twelve - laser cut wood triangles, six - nylon push rivets, one - 8" length of 0.90mm diameter monofilament line, one - finger nail clipper and one - lighter.

Figure 1: *Material required for model.*

Step one. Choose two triangles. Push a rivet through one side of one of the triangles. Attach the other triangle with the rivet. Complete a total of six triangle assembly sets.

Figure 2: *Step one* **-** *Two triangles & one rivet assembly.*

NOTE: <u>The rivet should be placed on the same side of the assembled set for the desired symmetry of the final assembly.</u>

Step Two. Choose two triangle assembly sets and a monofilament line. Push the line through one outside triangle and one inside triangle vertex. Heat the end of the line forming a ball larger than the hole – let cool. Pull the line tight and cut the other end of the line about 1/8th inch long. Heat the end of the line forming a ball larger than the hole – let cool.

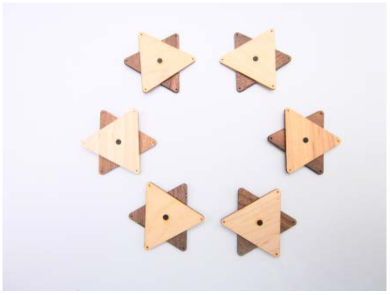

Figure 3: *Step one* **-** *Six triangle assembly sets.*

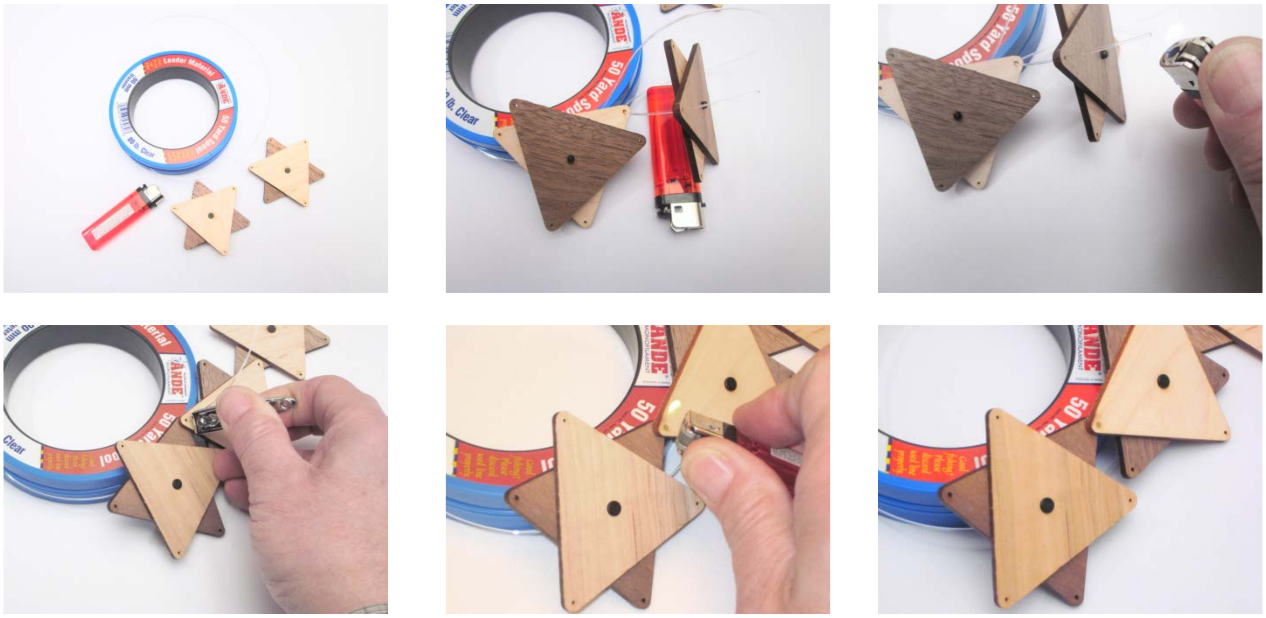

Figure 4: *Step two – Attach one inside triangle to one outside triangle of two sets.*

Step Three. Choose an adjacent vertex on the triangle assembly set and monofilament line. Push the line through one outside triangle and one inside triangle vertex. Repeat step two for all triangle sets and their matching holes.

NOTE: <u>The choice of matching vertex holes will determine the direction of allowable rotation for all remaining sets.</u>

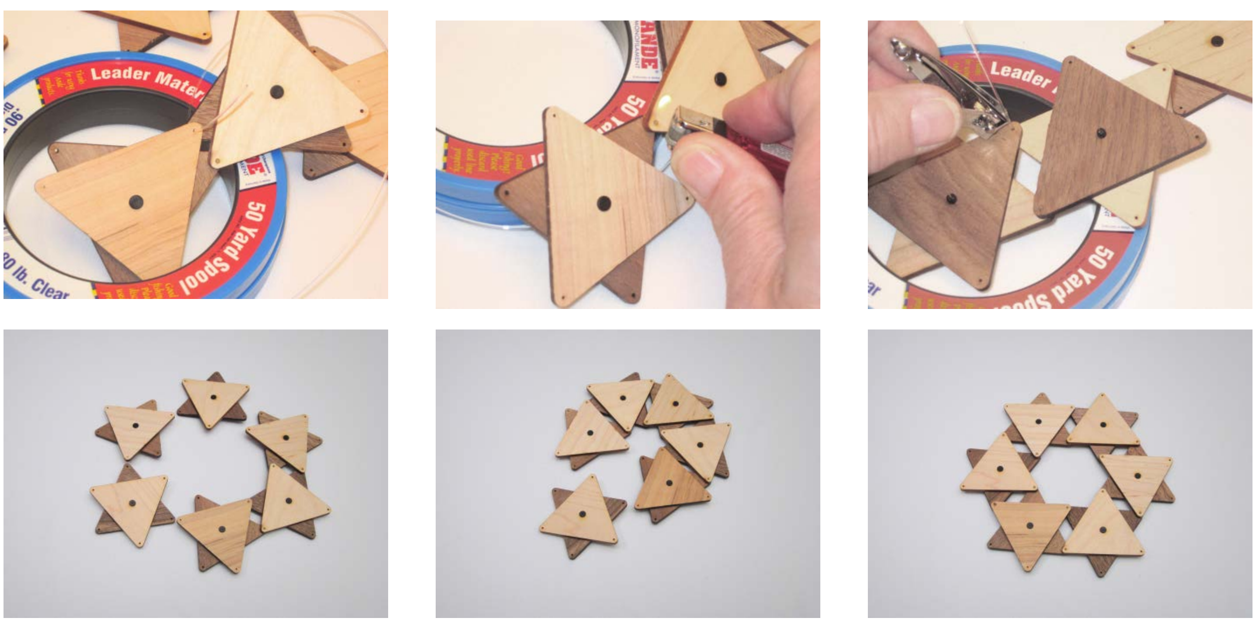

Figure 5: *Step three – Attach an adjacent inside triangle to an adjacent outside triangle of two sets.*

Step Four. Enjoy the transformation of a $(3^6)_D$ $(3^6)_L$ *chiral tessellation* to a $(3.6.3.6)_D$ $(3.6.3.6)_L$ *chiral tessellation.*

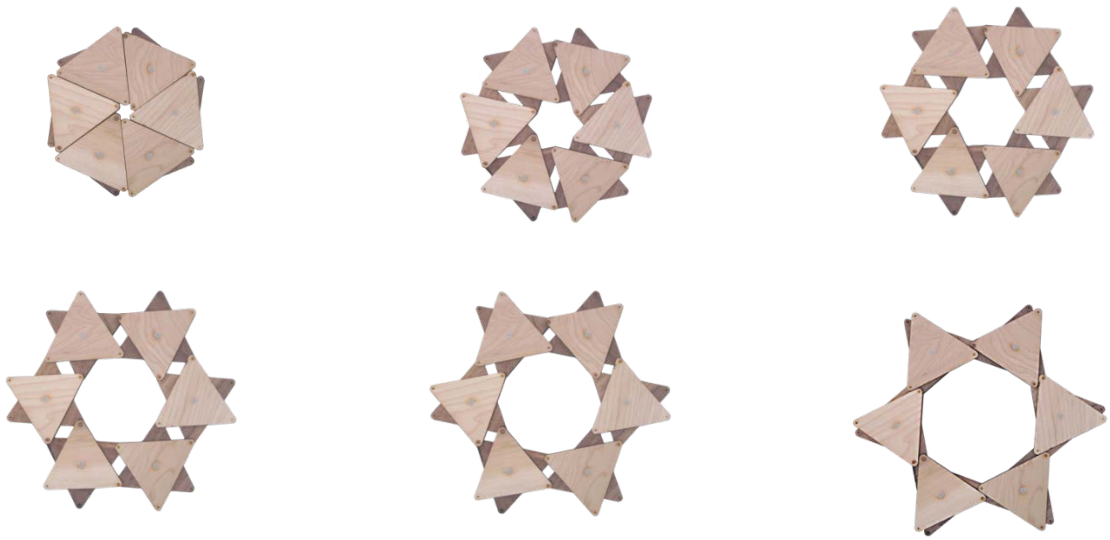

Figure 6: *Step four –* $(3^6)_D$ $(3^6)_L$ *chiral tessellation* to $(3.6.3.6)_D$ $(3.6.3.6)_L$ *chiral tessellation.*

Let's Make a $(3^6)_D$ $(3^6)_L$ Chiral Tessellation Dance, the Analytical Model

Description of Model. The analytical model is in the form of an animated film of a $(3^6)_D$ $(3^6)_L$ *Chiral Tessellation*. The center of rotation and translation for each set of triangles is located at the center of the six set tessellation. As with the physical model, the inside triangle of a set is rotated about its common outside triangle center. The two centers stay in common with each other throughout a full 360^O rotation of the respective triangle sharing the common center. The vertex of an inside triangle is connected to an adjacent outside triangle. A vertex connection is not allowed to be broken. The center of each set is translated along an axis through the center of the six set tessellation. In this model all triangles are equilateral.

Software Used. The animation and mathematical modeling used was *version 3.6 POVRAY*. [1] POVRAY is a freeware animation and graphics software package. The final animation was generated with *version 5.2.2.173 AVS Video Editor*. [2]

The POVRAY INI File. The *POVRAY Chiral_Poly.ini* file is an initialization file giving the command line code used to set the parameters for generating the animation frame files, in this case .jpg files.

```
; _______ Chiral_Poly.ini ______ Animation File____ Joseph D. Clinton ____ 2012 ________
  #version 3.62;
Antialias=On

Antialias_Threshold=0.1
Antialias_Depth=2
Input_File_Name=Chiral_Polygon.pov
Output_File_Name= FST36_.jpg
Initial_Frame=1
Final_Frame=360
```

```
// Subset_Start_Frame=1
//  Subset_End_Frame=361
Initial_Clock=0
Final_Clock=1

Cyclic_Animation=on
Pause_when_Done=off
```

The POVRAY POV File. The *POVRAY Chiral_Polygon.pov* file provides the code used to set the scene description for generating the animation.

```
// _________ Chiral_Polygon.pov ____ Joseph D. Clinton ___________ 2012 ___________ //
_________ POVRAY 3.62 Software ____________________________________________
#version 3.62;
global_settings {assumed_gamma 2.1}
#include "colors.inc"
#include "shapes.inc"
#include "textures.inc"
#include "metals.inc"
#include "skies.inc"
#include "Transformation_36.inc"
// #include "TET_ELLIPSE_01.inc"
light_source {<      0.0,      0.0,      0.0 > color rgb <1.000,1.000,1.000>}
light_source {<  100.0,  100.0,  100.0 > color rgb <1.000,1.000,1.000>}
light_source {<    00.0,    00.0,  100.0 > color rgb <1.000,1.000,1.000>}
light_source {< -100.0, -100.0, -100.0 > color rgb <0.500,0.500,0.500>}
camera {location  (< 0.0, 0.0, 20.0 >) sky < 0.0, 0.0, 1.0 >
look_at < 0.000,0.000,0.000 > angle 50}
// background { rgb < 1.000, 1.000, 1.000 > }   // White
background { rgb < 0.000, 0.000, 0.000 > }   // Black
#declare CPTex1 =  texture { pigment {color rgbt < 0.000 , 0.000 , 0.000 , .3 >   } }
#declare X1 = 0.0;
#declare Y1 = 0.0;
#declare Z1 = 0.0;
# declare Axis = union {
//    sphere    { < 0.0, 0.0, 0.0 > 0.5  texture { CPTex1 } }    // center of polygon set
      cylinder  { < 5.0, 0.0, 0.0 > , < -0.0,  0.0,  0.0 > 0.02 texture { pigment { Red } } }
      cylinder  { < 0.0, 5.0, 0.0 > , <  0.0, -0.0,  0.0 > 0.02 texture { pigment { Green } } }
      cylinder  { < 0.0, 0.0, 4.0 > , <  0.0,  0.0, -0.0 > 0.02 texture { pigment { Blue } } } }
// object { Axis no_shadow rotate < X1, Y1, Z1 > }
// }
object { TJ no_shadow  translate < X1, Y1, Z1 > scale 0.13 }    // Chiral polygon
```

The POVRAY INC File. The *POVRAY Transformation_36.inc* file is an include file giving the mathematical operations and logic code used for generating objects to be animated.

```
// ______ Transformation_36.inc ______ Joseph D. Clinton __________ 2012 ____________
// ______ Rotation & Translation equations for chiral (3^6)  ____________________________
// _________ POVRAY 3.62 Software ____________________________________________
#version 3.62;
```

```
// ________________ Rotation angle _______________________________________________
#declare RATetB  = clock*360; // Triangle rotation angle _ USE WITH Clock ______________
#declare EndRATetB = RATetB+0.000001;  // End Rotation cycles _ USE WITH clock _____
// ________________ Rotation angle _______________________________________________
  #declare RATetA  = 60/2; // One half triangle face angle
  #declare EL     = 10.000000000;                    // Edge length of triangle
  #declare AL     =  8.660254038;                    // Altitude of triangle  (sqrt( EL^2-(EL/2)^2)
  #declare AL23    = 2*(8.660254038/3);              // two thirds Altitude of triangle
  #declare RTet   = 5.773502692; // (sqrt(3)/3)*10; // Center of triangle to vertex of triangle
  #declare XT0   = 0.0 ;
  #declare YT0   = 0.0 ;
  #declare XT1   = 0.0 ;
  #declare YT1   = RTet;
  #declare XT2   = EL/2;
  #declare YT2   = RTet-AL;
  #declare XT3   = -EL/2;
  #declare YT3   = RTet-AL;
  #declare InT = 0.999;
  #declare OnT = 1.001;
// ___________________________
  #declare PETex1 =  texture { pigment{color rgbt < 0.000 , 0.000 , 0.000 , .3 >   } }
  #declare PETex2 =  texture { pigment{color rgbt < 0.000 , 0.000 , 0.000 , .3 >   } }
  #declare PETex3 =  texture { pigment{color rgbt < 0.000 , 0.000 , 0.000 , .3 >   } }
  #declare PETex4 =  texture { pigment{color rgbt < 0.000 , 0.000 , 0.000 , .3 >   } }
  #declare PETex11 = texture {pigment{color rgbt < 1.000 , 0.000 , 0.000 , .8 > }} // { pigment{ Red } }
Botton
  #declare PETex22 = texture {pigment{color rgbt < 1.000 , 0.500 , 0.200 , .5 > }} // { pigment{ Orange }
} Top
  #declare PErad1 = 0.15 ;   // Radius of vertex
  #declare PErad2 = 0.1 ;
  #declare PEradc = 0.15;    // Triangle center
  #declare Rrad   = 0.05;    // Radius of edge
  #declare RETex1 = texture { pigment{color rgbt < 1.000 , 1.000 , 1.000 , .3 > } } // White
  #declare RETex2 = texture { pigment{color rgbt < 0.000 , 0.000 , 0.000 , .3 > } } // Black
// ____________________________
// _______ Transformation of a 3.3.3.3.3.3 tessellation ___________________________________
  #declare TJ = union {
//// ___________________ Not used with Clock __________________________________________
////// #declare RATetB = 0.000001;                   // start degree of rotation  // not used with clock
// not used with clock
////// #declare EndRATetB = 120.000001;          // end degree of rotation    // not used with clock
// not used with clock
////// #while (RATetB < EndRATetB+0.000002)  // operation                      // not used with clock
//// ___________________ Not used with Clock __________________________________________
  #declare RATetC = 180 -( RATetA+RATetB);                          // Translation triangle angle C
  #declare CL = (RTet*sin(RATetC*pi/180))/sin(RATetA*pi/180);  // Translation triangle length C

//    #declare CD = (RTet*cos(RATetB*pi/180));               // Center of triangle long  leg along C
//    #declare TD = AL23-CD;                                  // Center of triangle short leg along C
#declare InT =  0.0001;  //  Top
#declare OnT = -0.0001;  //  Bottom
```

```
// ____________ Basic triangles (Top & Bottom) ____________________________________
#declare X1   = -RTet*(sin(RATetB*pi/180) ) ;
#declare Y1   =  RTet*(cos(RATetB*pi/180) ) ;
#declare Z1  = 0.0; // X1*tan(HDA*pi/180) ;
#declare TC1 = union {  // Basic triangles (Top & Bottom) at center
//  sphere { < XT0, YT0, Z1 >  PEradc texture { RETex1} rotate < 0.0, 0, 0.0 >}  // Triangle Center
sphere   { < XT1, YT1, Z1 >  PErad1 texture { RETex1}  rotate < 0.0, 0, 0.0 >} // Triangle vertex
sphere   { < XT2, YT2, Z1 >  PErad1 texture { RETex1}  rotate < 0.0, 0, 0.0 >} // Triangle vertex
sphere   { < XT3, YT3, Z1 >  PErad1 texture { RETex1}  rotate < 0.0, 0, 0.0 >} // Triangle vertex
cylinder { < XT1, YT1, Z1 >, < XT2, YT2, Z1 > Rrad texture { RETex1 } }        // triangle edge
cylinder { < XT2, YT2, Z1 >, < XT3, YT3, Z1 > Rrad texture { RETex1 } }        // triangle edge
cylinder { < XT3, YT3, Z1 >, < XT1, YT1, Z1 > Rrad texture { RETex1 } }        // triangle edge
mesh {triangle {< XT1, YT1, Z1 > < XT2, YT2, Z1 > < XT3, YT3, Z1 > }          // Triangle face
  }   rotate < 0.0, 0.0, 0.0 >                                                // Triangle face
  }
// _____________ Basic triangles (Top & Bottom) ___________________________________
#declare LD_Triangle = union {
object { TC1 translate < 0.0, 0.0, OnT > rotate < 0.0,  0.0, 180+RATetB >  translate < 0.0, CL, 0.0 >
texture  { PETex11 } }  //  Bottom
object { TC1 translate < 0.0, 0.0, InT > rotate < 0.0,  0.0, 180-RATetB >  translate < 0.0, CL, 0.0 >
texture  { PETex22 } }  //  Top
      }
#declare FST_Triangle = union {
// First set triangles
     object { LD_Triangle  rotate < 0.0,  0.0,      0.0 > }                  // First set triangles
     object { LD_Triangle  rotate < 0.0,  0.0,    60.0 > }                   // First set triangles
     object { LD_Triangle  rotate < 0.0,  0.0,  120.0 > }                    // First set triangles
     object { LD_Triangle  rotate < 0.0,  0.0,  180.0 > }                    // First set triangles
     object { LD_Triangle  rotate < 0.0,  0.0,  240.0 > }                    // First set triangles
     object { LD_Triangle  rotate < 0.0,  0.0,  300.0 > }                    // First set triangles
       }
#declare SST_Triangle = union {
     object { FST_Triangle  translate < 0.0, (3*CL), 0.0 > rotate < 0.0,  0.0,      0.0 > }
// Second set triangles
     object { FST_Triangle  translate < 0.0, (3*CL), 0.0 > rotate < 0.0,  0.0,    60.0 > }
// Second set triangles
     object { FST_Triangle  translate < 0.0, (3*CL), 0.0 > rotate < 0.0,  0.0,  120.0 > }
// Second set triangles
     object { FST_Triangle  translate < 0.0, (3*CL), 0.0 > rotate < 0.0,  0.0,  180.0 > }
// Second set triangles
     object { FST_Triangle  translate < 0.0, (3*CL), 0.0 > rotate < 0.0,  0.0,  240.0 > }
// Second set triangles
     object { FST_Triangle  translate < 0.0, (3*CL), 0.0 > rotate < 0.0,  0.0,  300.0 > }
// Second set triangles
        }
object { FST_Triangle }  // First set triangles
//    object { SST_Triangle }  // Second set triangles
//     cylinder { < XT1, YT1, Z1 >, < XT2, YT2, Z1 > Rrad rotate < 0.0, 0.0, 180.0 > translate < 0.0,
AL23, 0.0 > texture { RETex2 } } // triangle edges
//     cylinder { < XT2, YT2, Z1 >, < XT3, YT3, Z1 > Rrad rotate < 0.0, 0.0, 180.0 > translate < 0.0,
AL23, 0.0 > texture { RETex2 } } // triangle edges
```

```
//      cylinder { < XT3, YT3, Z1 >, < XT1, YT1, Z1 > Rrad rotate < 0.0, 0.0, 180.0 > translate < 0.0,
AL23, 0.0 > texture { RETex2 } } // triangle edges
//// __________ Not used with Clock ____________________________________________________
///// #declare RATetB = RATetB + 5;  // increment degree of rotation    // not used with clock
/////    #end // ------------- end of loop ----                         // not used with clock
//// __________ Not used with Clock ____________________________________________________
}
// ____________ End of Transformation of
```

Animation Frames. The animation rotates the vertices of the triangle sets through a complete 360^{O} cycle.

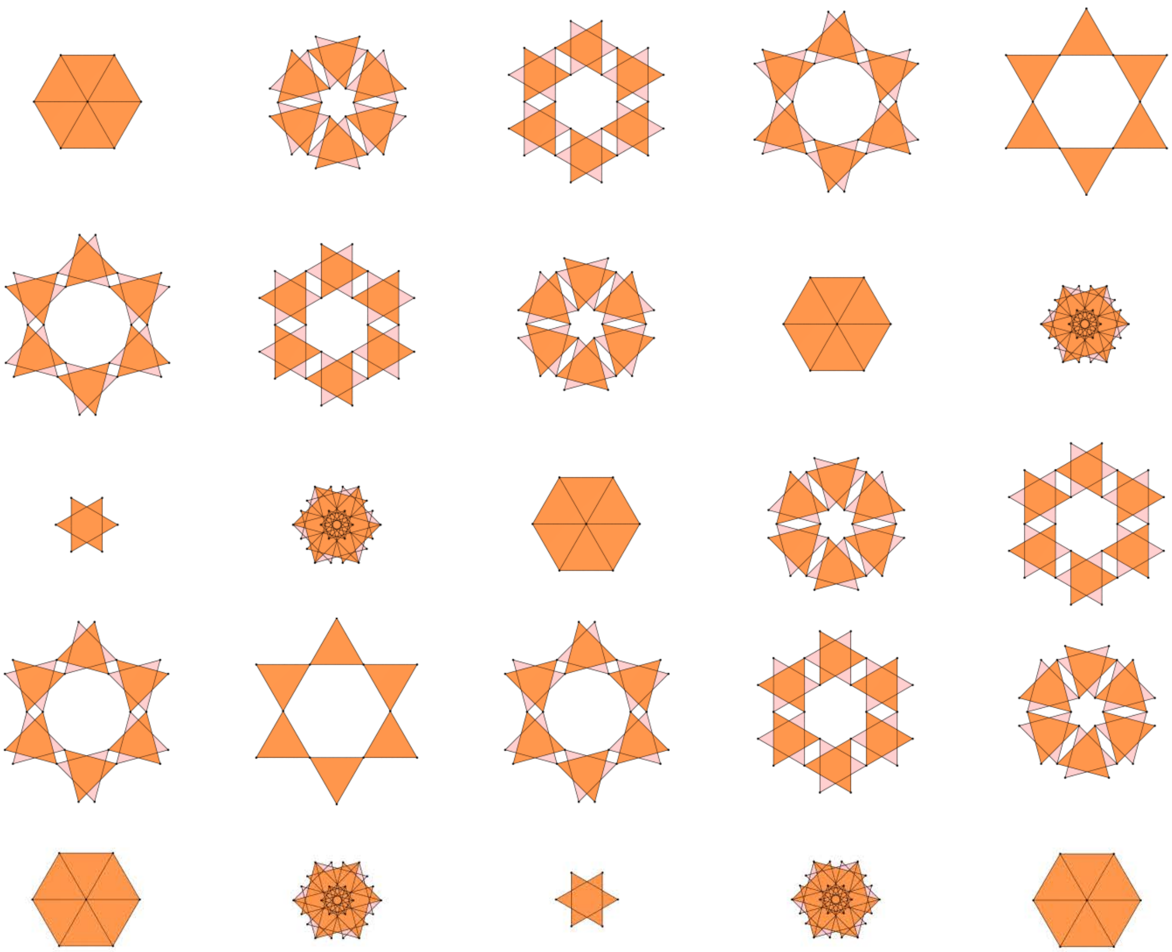

Figure 7: *Animation frames* $(3^6)_D$ $(3^6)_L$ *chiral tessellation,* 0^{O} *to* 360^{O} *at* 15^{O} *intervals.*

Discussion. An open discussion will be held after viewing the analytical model and constructing the physical model. The discussion will focus on limitations between the two models.

[1] *POVRAY*, POV-Team, 2002. Persistence of Vision (TM) Ray Tracer. POV-Team, Williamston, Australia. http://www.povray.org/

[2] *AVS Video Editor*, Online Media Technologies Ltd. 2011. http://www.AVS4You.com