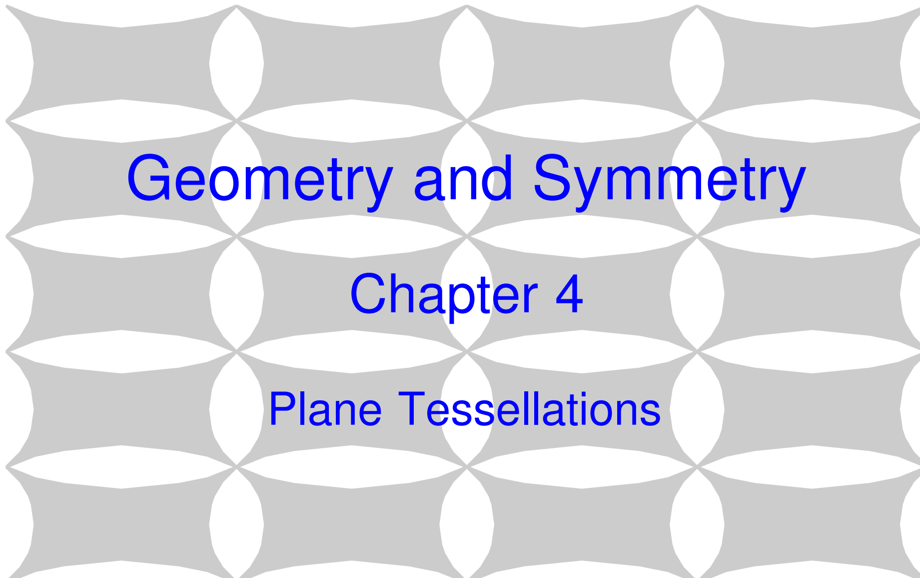


Chapter 4: Plane Tessellations

In this chapter I will return to plane isometries and tessellations. We will see that this is actually harder than working on the sphere which is compact. Not only will we need to use infinite symmetry groups but we will be working with all 4 isometry types. Our symmetry groups will be generated by actual isometries which makes our discussion different than than the classical discussion. There are two uses for symmetry groups the c*onstruction groups* and *full symmetry groups.*

In this book I will use stone pavers and floor tiles rather than wallpaper patterns to avoid problems caused by the artistry. The pavers and tiles will assumed to be polygons for simplicity but may have many sides to approximate smooth curves as in the graphic above.

I expect the reader to review Chapter 2 of my *Transformation and Symmetry* book. A pdf version is available on my website, a **Mathematica notebook** is available from my page in the Wolfram Community. In particular the reader should be familiar with Transformation Functions as a data type in **Mathematica.**

To review, the 4 isometry types are Translations and Rotations which are given by Mathematica's built-in functions, Reflections and Glide reflections also have built-in functions but I prefer my own functions. First a hidden subroutine `getTF2D` from Chapter 2.

In[ ]:=
```
reflectionTF2D [{v_, w_}] := Module[{p, pp},
p = N[v - w];
pp = {p⟦2⟧, -p⟦1⟧};
getTF2D[{v, w, v + pp}, {v, w, v - pp}]]
```

In[ ]:=
```
glideReflectionTF2D [{v_, w_}] := reflectionTF2D [{v, w}]@* TranslationTransform [w - v]
```

I also note that to invert TransformationFunctions one should use my function because Mathematica’s Inverse[] does not work well for transformation functions.

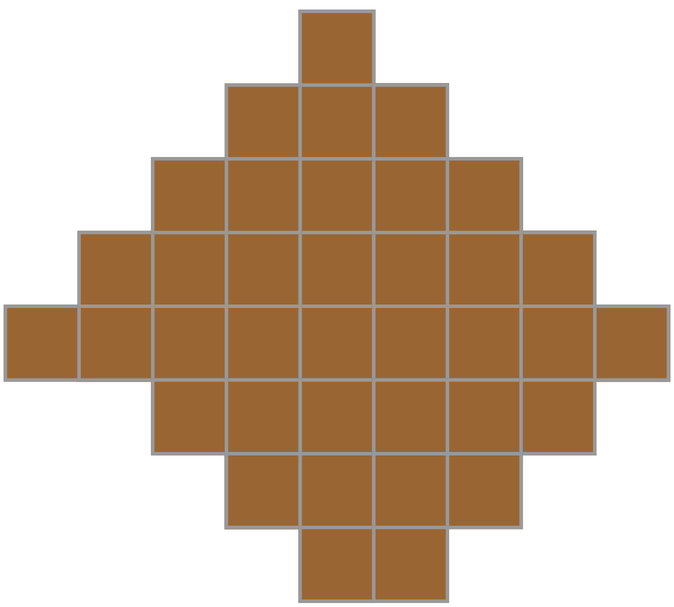
In[ ]:=
```
InverseTF [λ_] := TransformationFunction [Inverse[TransformationMatrix [λ]]]
```

In this chapter we will name translations starting with τ, reflections with ρ, glide reflections with γ and rotations with σ except for half turns, rotations in π (180°) will be considered a special category and denoted by names beginning in η.

4.1 An Example

I will start with an example: the square tiling. This is one of the most common paving or tiling you will see and in one sense the simplest. On the the other hand we will see its symmetry group is one of the largest. This tessellation theoretically tiles the entire real plane, a small piece looks like.

In[ ]:=

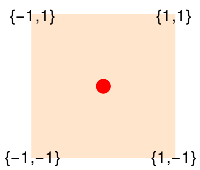


Our base or home tile is given by the square

In[ ]:=
```
S = {{1, 1}, {1, -1}, {-1, -1}, {-1, 1}}
```

Out[ ]=
```
{{1, 1}, {1, -1}, {-1, -1}, {-1, 1}}
```

In[ ]:=
```
Graphics[{{LightOrange, Polygon[S]},
  {Black, Text["{1,1}", {1, 1}], Text["{1,-1}", {1, -1}], Text["{-1,-1}", {-1, -1}],
   Text["{-1,1}", {-1, 1}]}, {Red, PointSize[.1], Point[{0, 0}]}}, ImageSize → Tiny]
```

Out[ ]=


The centroid is the origin {0, 0} .

The simplest symmetries are the translations we will call

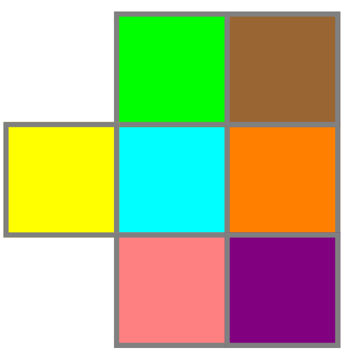
In[ ]:=
```
τv2 = TranslationTransform[{2, 0}]
τh2 = TranslationTransform[{0, 2}]
```

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 1 & 0 & 2 \\ 0 & 1 & 0 \\ \hline 0 & 0 & 1 \end{array}\right)\right]$

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 1 & 0 & 0 \\ 0 & 1 & 2 \\ \hline 0 & 0 & 1 \end{array}\right)\right]$

To see how this works, here are 6 transforms of our base polygon in the center.

```
Graphics[{EdgeForm[{Gray, Thickness[.015]}], {Cyan, Polygon[S]}, {Orange, Polygon[τv2@S]},
  {Green, Polygon[τh2@S]}, {Yellow, Polygon[InverseTF[τv2]@S]},
  {Pink, Polygon[InverseTF[τh2]@S]}, {Brown, Polygon[τv2@* τh2@S]},
  {Purple, Polygon[τv2@* InverseTF[τh2]@S]}}, ImageSize → Small]
```

Out[ ]=


Continuing this way it is easy to see that each tile can be constructed by applying a element of the group generated by `{τv2, τh2}` we can get any tile in the tessellation.

Even though we will not get a finite group of transformations we can still use our finite transformation group code in Chapter 2 to approximate the infinite groups. Later in this chapter I will give a better alternative.

In[ ]:=
```
G1 =
 finiteTransGroup [<| 1 → τv2, 2 → InverseTF [τv2], 3 → τh2, 4 → InverseTF [τh2] |> , {0, 0}, 4]
```

» number of group elements calculated 41

Out[ ]=
```
{{1}, {2}, {3}, {4}, {1, 1}, {1, 2}, {1, 3}, {1, 4}, {2, 2}, {2, 3}, {2, 4}, {3, 3}, {4, 4}, {1, 1, 1},
 {1, 1, 3}, {1, 1, 4}, {1, 3, 3}, {1, 4, 4}, {2, 2, 2}, {2, 2, 3}, {2, 2, 4}, {2, 3, 3},
 {2, 4, 4}, {3, 3, 3}, {4, 4, 4}, {1, 1, 1, 1}, {1, 1, 1, 3}, {1, 1, 1, 4}, {1, 1, 3, 3},
 {1, 1, 4, 4}, {1, 3, 3, 3}, {1, 4, 4, 4}, {2, 2, 2, 2}, {2, 2, 2, 3}, {2, 2, 2, 4},
 {2, 2, 3, 3}, {2, 2, 4, 4}, {2, 3, 3, 3}, {2, 4, 4, 4}, {3, 3, 3, 3}, {4, 4, 4, 4}}
```

A technical comment is that since translations and glide reflections have infinite order this procedure will not find the inverses, therefore we have to add the inverses as generators. We will see later that often if half turns or reflections are also generators then the inverses of translations and glide reflections will be compositions of the generators so specifically adding the inverses may not be necessary.

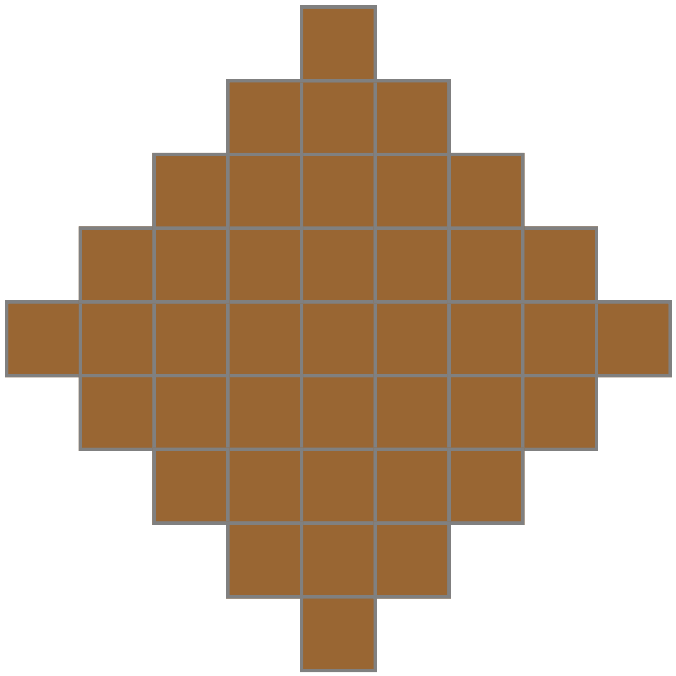
A second technical comment is that the second argument of the finiteTransGroup function is a test point. Using one such as {0,0} above only gets that part of the group that is in the orbit of {0,0}. To get all elements of the group use a pseudo random test point, one such as {.2152 , .1483 } is sufficiently random in our context.

In this chapter a useful routine for constructing tessellations is

In[ ]:=
```
groupTessellate [G_, tas_, P_, col_, bcol_] := Module[{tab},
  tab = Table[TasTF[g, tas]@P, {g, G}];
  Graphics [{col , EdgeForm [{Thickness [.005], bcol}], Polygon [tab]}]]
```

In[ ]:=
```
groupTessellate[
 G1, <|1 → τv2, 2 → InverseTF[τv2], 3 → τh2, 4 → InverseTF[τh2]|>, S, Brown, Gray]
```

Out[ ]=



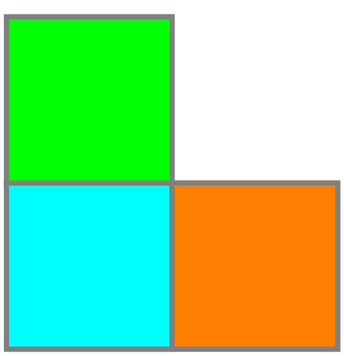
So I say group G1 is a construction group for the square tessellation.

But there are other symmetries . We note the entire section above has rotation symmetries, in particular a 4 fold rotation about the center, {0,0}. If we look more closely we see that the point {1,0}, on the base square is the center of a half turn.

In[ ]:=
```
σ4 := RotationTransform[Pi / 2, {0, 0}]
η1 := RotationTransform[Pi, {1, 0}]
```

In[ ]:=
```
Graphics[{EdgeForm[{Gray, Thickness[.015]}], {Cyan, Polygon[S]},
  {Orange, Polygon[η1@S]}, {Green, Polygon[σ4@* η1@S]}}, ImageSize → Small]
```

Out[ ]=



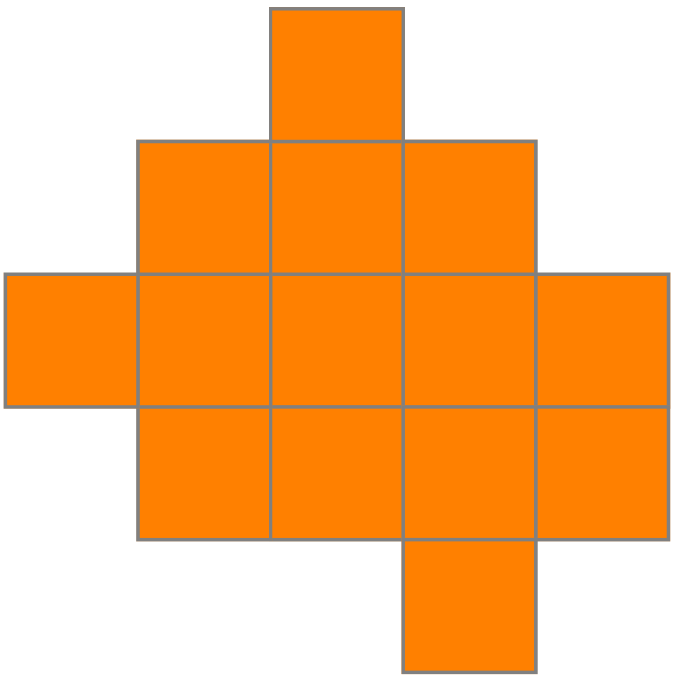
We could guess that these two transformations could also construct the group.

In[ ]:= G2 = finiteTransGroup [<| 1 → σ4, 2 → η1 |> , {2.3214 , 3.571}, 6]

» number of group elements calculated 38

Out[ ]= {{1}, {2}, {1, 1}, {1, 2}, {2, 1}, {2, 2}, {1, 1, 1}, {1, 1, 2}, {1, 2, 1}, {2, 1, 1}, {2, 1, 2},
{1, 1, 1, 2}, {1, 1, 2, 1}, {1, 2, 1, 1}, {1, 2, 1, 2}, {2, 1, 1, 1}, {2, 1, 1, 2}, {2, 1, 2, 1},
{1, 1, 1, 2, 1}, {1, 1, 2, 1, 1}, {1, 1, 2, 1, 2}, {1, 2, 1, 1, 1}, {1, 2, 1, 1, 2},
{1, 2, 1, 2, 1}, {2, 1, 1, 2, 1}, {2, 1, 2, 1, 1}, {2, 1, 2, 1, 2}, {1, 1, 1, 2, 1, 1},
{1, 1, 1, 2, 1, 2}, {1, 1, 2, 1, 1, 1}, {1, 1, 2, 1, 1, 2}, {1, 1, 2, 1, 2, 1}, {1, 2, 1, 1, 2, 1},
{1, 2, 1, 2, 1, 1}, {2, 1, 1, 2, 1, 1}, {2, 1, 1, 2, 1, 2}, {2, 1, 2, 1, 1, 1}, {2, 1, 2, 1, 1, 2}}

In[ ]:= groupTessellate [G2, <| 1 → σ4, 2 → η1 |> , S, Orange , Gray]

Out[ ]=


Again we construct a square tessellation, but the squares shown are somewhat different from our first attempt.

Both of these construction groups contain only *direct* isometries, that is, isometries which are orientation preserving or alternatively, have the upper left 2×2 block in the transformation matrix with determinant 1. But we can see that this tessellation also has reflection symmetry. Three obvious reflections are

In[ ]:= ρh := reflectionTF2D [{{-1, 0}, {1, 0}}]
ρv := reflectionTF2D [{{0, -1}, {0, 1}}]
ρd := reflectionTF2D [{{-1, -1}, {1, 1}}]

These are all reflections of the square. Note that

In[ ]:= **ρd@* ρh**

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 0. & -1. & 0. \\ 1. & 0. & 0. \\ \hline 0. & 0. & 1. \end{array}\right)\right]$

while

In[ ]:= **σ4 = RotationTransform [Pi / 2]**

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 0 & -1 & 0 \\ 1 & 0 & 0 \\ \hline 0 & 0 & 1 \end{array}\right)\right]$

so σ4 will be contained in any group containing ρh and ρd

Also note

In[ ]:= **σ4@* ρh@* InverseTF [σ4]**

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} -1. & 0. & 0. \\ 0. & 1. & 0. \\ \hline 0. & 0. & 1. \end{array}\right)\right]$

is the same as

In[ ]:= **ρv**

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} -1. & 0. & 0. \\ 0. & 1. & 0. \\ \hline 0. & 0. & 1. \end{array}\right)\right]$

so if ρh and ρd are in a group then so is ρv .

Unfortunately all these transformations leave {0,0} fixed so we will only get a finite group. So we need another reflection

In[ ]:= **ρ1 = reflectionTF2D [{{1, -1}, {1, 1}}]**

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} -1. & 0. & 2. \\ 0. & 1. & 0. \\ \hline 0. & 0. & 1. \end{array}\right)\right]$

Now consider

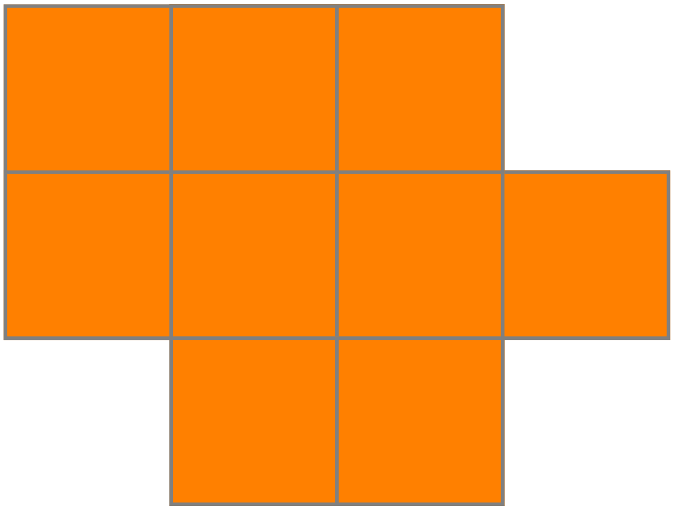
In[ ]:= **G3 = finiteTransGroup [<| 1 → ρv, 2 → ρd, 3 → ρ1 |> , {2.3214 , 3.571}, 5]**

» number of group elements calculated 41

Out[ ]= {{1}, {2}, {3}, {1, 1}, {1, 2}, {1, 3}, {2, 1}, {2, 3}, {3, 2}, {1, 2, 1}, {1, 2, 3}, {1, 3, 2}, {2, 1, 2}, {2, 1, 3}, {2, 3, 2}, {3, 2, 1}, {3, 2, 3}, {1, 2, 1, 2}, {1, 2, 1, 3}, {1, 2, 3, 2}, {1, 3, 2, 1}, {1, 3, 2, 3}, {2, 1, 2, 3}, {2, 1, 3, 2}, {2, 3, 2, 1}, {2, 3, 2, 3}, {3, 2, 1, 2}, {3, 2, 1, 3}, {1, 2, 1, 2, 3}, {1, 2, 1, 3, 2}, {1, 2, 3, 2, 1}, {1, 2, 3, 2, 3}, {1, 3, 2, 1, 2}, {1, 3, 2, 1, 3}, {2, 1, 2, 3, 2}, {2, 1, 3, 2, 1}, {2, 1, 3, 2, 3}, {2, 3, 2, 1, 2}, {2, 3, 2, 1, 3}, {3, 2, 1, 2, 3}, {3, 2, 1, 3, 2}}

In[ ]:=
```
groupTesselate[G3, <|1 → ρv, 2 → ρd, 3 → ρ1|>, S, Orange, Gray]
```

Out[ ]=



which appears to be an additional square tessellation of the plane . These groups are different, certainly G2, G3 are bigger than G1 but since G3 contains reflections it is different from G2 which only has direct isometries. We will see G2 is contained in G3 which we will find is the complete group of of symmetries of the square tessellation. In particular it will contain translations and glide reflections, rotations of order 4 around all vertices and half turns about all edge midpoints.

Given this example we can now look at the theory .

Section 4.2 The Theory

In this long section I give the basic theory of symmetry of plane tessellations

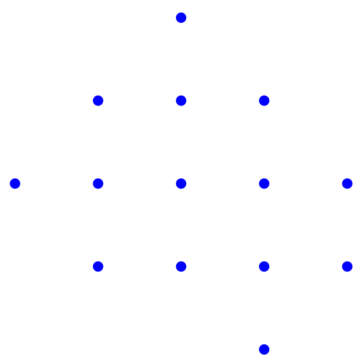
4.2.1 Discrete and complete plane tessellations

Following Yale, a group of plane transformations G is *discrete* if given any bounded set W of the plane and any point p in the plane the G-orbit of p contains only finitely many points in W.

In practice, working with isometries, we can use any circular disc about the origin for W and any convenient point p to demonstrate this. In particular all construction groups of isometries will be discrete, a non-discrete group would just give a mess.

A *complete plane tessellation group* is one in which the tessellation, if continued would fill up the whole plane. Groups G1,G2 and G3 in section 4.1 will be discrete complete tessellation groups if we just follow one point of the square. The group association routine can be used to demonstrate. For example

In[ ]:=
```
G2A = groupAssoc [G2, <|1 → σ4, 2 → η1|>, {0, 0}]
```
Out[ ]=
```
<|{1} → {0, 0}, {2} → {2, 0}, {1, 2} → {0, 2}, {1, 1, 2} → {-2, 0},
 {2, 1, 2} → {2, -2}, {1, 1, 1, 2} → {0, -2}, {1, 2, 1, 2} → {2, 2}, {2, 1, 1, 2} → {4, 0},
 {1, 1, 2, 1, 2} → {-2, 2}, {1, 2, 1, 1, 2} → {0, 4}, {1, 1, 1, 2, 1, 2} → {-2, -2},
 {1, 1, 2, 1, 1, 2} → {-4, 0}, {2, 1, 1, 2, 1, 2} → {4, -2}, {2, 1, 2, 1, 1, 2} → {2, -4}|>
```

In[ ]:=
```
Graphics[{Blue, PointSize[.03], Point[Values[G2A]]}, ImageSize → Small]
```
Out[ ]=


This plot shows no clustering and a definite 2 dimensional picture which if continued would fill up the plane.

On the other hand, consider, again using the transformation functions defined in 4.1 the group

In[ ]:=
```
H1 = finiteTransGroup [<|1 → τv, 2 → ρh|>, {0, 0}, 4]
```
» number of group elements calculated 8

Out[ ]=
```
{{1}, {2}, {1, 1}, {2, 1}, {1, 1, 1}, {2, 1, 1}, {1, 1, 1, 1}, {2, 1, 1, 1}}
```

In[ ]:=
```
H1A = groupAssoc [H1, <|1 → τv, 2 → ρh|>, {0, 0}]
```
Out[ ]=
```
<|{1} → {2, 0}, {2} → {0., 0.}, {1, 1} → {4, 0}, {2, 1} → {-2., 0.}, {1, 1, 1} → {6, 0},
 {2, 1, 1} → {-4., 0.}, {1, 1, 1, 1} → {8, 0}, {2, 1, 1, 1} → {-6., 0.}|>
```

In[ ]:=
```
Graphics[{Blue, PointSize[.03], Point[Values[H1A]]}, ImageSize → Small]
```
Out[ ]=

Here the points give a 1 dimension set so we will not get a complete tessellation group .

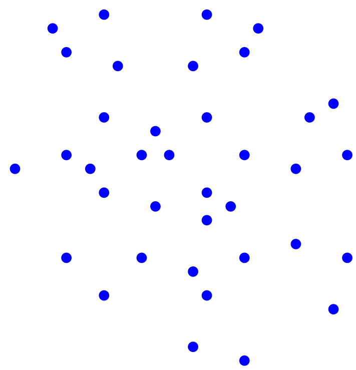
On the other hand consider

In[ ]:=
```
σ3 := N[RotationTransform [2 Pi / 3, {1, 0}]]
```

In[ ]:=
```
H2 = finiteTransGroup [<|1 → σ4, 2 → σ3|>, {0, 0}, 6];
```
» number of group elements calculated 37

In[ ]:=
```
H2A = groupAssoc[H2, <|1 → σ4, 2 → σ3|>, {0, 0}];
Graphics[{Blue, PointSize[.03], Point[Values[H2A]]}, ImageSize → Small]
```

Out[ ]=



We are getting a jumble. This group is not discrete. It would be complete as a 2-dimensional set but we will not get a symmetric tessellation out of this.

4.2.2 An algorithm for discrete groups

So far we have taken the view that if we use our `finiteTransGroup` procedure then the we will get our group elements but unlike the the finite case this will not stabilize as n gets large, but continue to grow. This is actually the correct idea. But in this Chapter we will use an enhanced procedure, based on `finiteTransGroup`, which gives us more information and options.

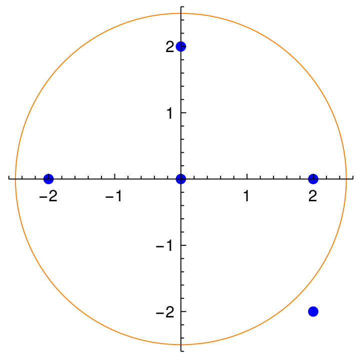
In[ ]:=
```
Options[discreteTransGroup] =
  {plot → True, returnGroup → False, returnAssoc → False};
discreteTransGroup[tas_, tp_, r_, n_, OptionsPattern[]] := Module[{G, A, B, ncpts},
  G = finiteTransGroup[tas, tp, n];
  If[OptionValue[returnGroup], Return[G]];
  A = groupAssoc[G, tas, tp];
  If[OptionValue[returnAssoc], Return[A]];
  B = Reap[Do[If[Norm[Values[A]⟦i⟧] < r, Sow[Values[A]⟦i⟧]], {i, Length[Values[A]]}]]⟦2, 1⟧;
  If[Length[B] == 0, Echo["No c-points"],
   Echo[Length[B], "number c-points"]];
  If[OptionValue[plot], Graphics[{{Orange, Circle[{0, 0}, r]},
     {Blue, PointSize[.03], Point[Values[A]]}}, Axes → True, ImageSize → Small]]]
```

Without specifying a group name or options it returns the pictures in 4.2.1

In[ ]:=
```
discreteTransGroup[<|1 → σ4, 2 → η1|>, {0, 0}, 2.5, 3]
```

» number of group elements calculated 5

» number c–points 4

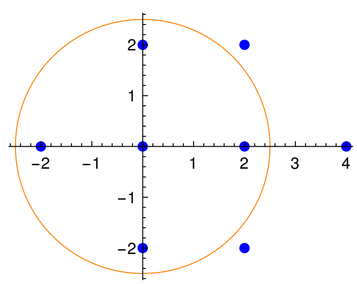


This shows a circle of radius 2.5 about the origin with 4 points inside and one outside .

```
discreteTransGroup [<| 1 → σ4, 2 → η1 |> , {0, 0}, 2.5, 4]
```

» number of group elements calculated 8

» number c–points 5

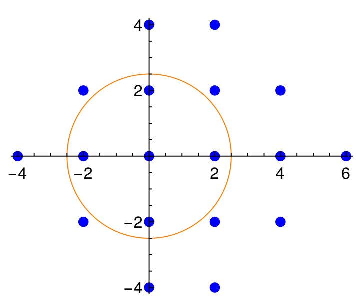


Now there are 5 points inside, c points . But if we increase n to 7

```
discreteTransGroup [<| 1 → σ4, 2 → η1 |> , {0, 0}, 2.5, 7]
```

» number of group elements calculated 18

» number c–points 5



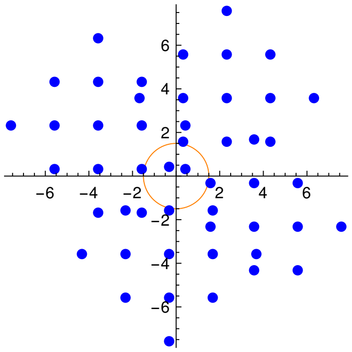
there are still only 5 points inside the circle, now with a larger number of points outside the circle. Thus the number or elements calculated does not stabilize, the number of c-points does so this graphic shows that we have a complete discrete group. Notice also that the points show a nice lattice so could get nice tessellations .

If instead we use a random test point we will get more points

In[ ]:=
```
discreteTransGroup [<| 1 → σ4, 2 → η1 |> , {2.321 , 3.576}, 1.5 , 7]
```

» number of group elements calculated 50

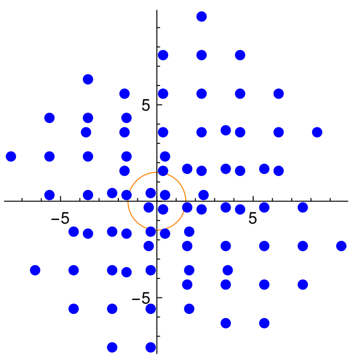
» number c–points 2

Out[ ]=


In[ ]:=
```
discreteTransGroup [<| 1 → σ4, 2 → η1 |> , {2.321 , 3.576}, 1.5 , 9]
```

» number of group elements calculated 81

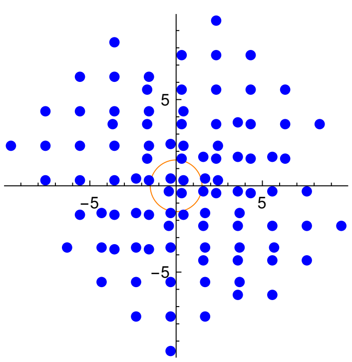
» number c–points 4

Out[ ]=


In[ ]:=
```
discreteTransGroup [<| 1 → σ4, 2 → η1 |> , {2.321 , 3.576}, 1.5 , 10]
```

» number of group elements calculated 98

» number c–points 4

Out[ ]=


We get a messier picture but the c-points are stabilizing at 4. This is a better approximation of the actual group defined with these generators.

In the next subsection we will be using the group Association of a group . This can be found by using **discreteTransGroup** using option **returnAssoc->True** . If you also set returnGroup->True then that will override the return association. If you want both use return Association->True as the group can be recovered as the Keys of the association. With either or both options chosen you will not get the plot. However this can be also be recovered as the values of the association. Here is an example

```
In[ ]:= G2A = discreteTransGroup [<|1 → σ4, 2 → η1|>, {0, 0}, 1.5, 4, returnAssoc → True]
```

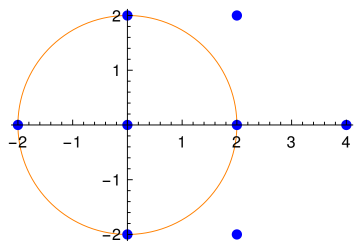
» number of group elements calculated 8

```
Out[ ]= <|{1} → {0, 0}, {2} → {2, 0}, {1, 2} → {0, 2}, {1, 1, 2} → {-2, 0},
 {2, 1, 2} → {2, -2}, {1, 1, 1, 2} → {0, -2}, {1, 2, 1, 2} → {2, 2}, {2, 1, 1, 2} → {4, 0}|>
```

```
In[ ]:= G2 = Keys[G2A]
Out[ ]= {{1}, {2}, {1, 2}, {1, 1, 2}, {2, 1, 2}, {1, 1, 1, 2}, {1, 2, 1, 2}, {2, 1, 1, 2}}
```

```
In[ ]:= Graphics[{{Orange, Circle[{0, 0}, 2]}, {Blue, PointSize[.03], Point[Values[G2A]]}},
 Axes → True, ImageSize → Small]
```

Out[ ]=



4.2.3 Subgroups and equality of groups

Since we are working with actual transformations the term subgroup will only apply when the actual transformations of the smaller group are members of the larger group. Since we are defining groups by non-unique generators it is enough to show that the generators of the smaller group are members of the larger group.

It will not be necessary to actually compute part of the smaller group . But it will be necessary to compute a group association for the larger group at possibly a large size n using a random test point tp. Here random is important is that we want to associate all elements with a test value, not just elements in the smaller orbit belonging to a non-random test point. See the last few examples to see the difference. We use the following algorithm.

```
In[ ]:= Clear[α]
```

```
In[ ]:= Options[AFindKeys] = {tol → 1.*^-5};
AFindKeys[A_, α_, tp_, OptionsPattern[]] := Module[{S, R},
   v = α@tp;
   S = Normal[A];
   R = Reap[Do[If[Norm[s⟦2⟧ - v] < OptionValue[tol], Sow[s⟦1⟧]], {s, S}]];
   If[Length[R⟦2⟧] > 0, Return[R⟦2, 1⟧], Return["Key not found"]]]
```

Given a transformation function α and a group G to show α is an element of G do the following

1. Calculate the group association GA for G with a reasonably large n with random test point tp.
2. Execute AFindKeys with the same test point tp. If it gives an error message α is not found.

Example : Check if the vertical translation of length 2 is contained in the group G2 above and in Section 2.1

In[ ]:=
```
τh
```

Out[ ]= TransformationFunction$\left[\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{pmatrix}\right]$

In[ ]:=
```
G2A =
  discreteTransGroup[<|1 → σ4, 2 → η1|>, {1.7655, 2.0243}, 1.5, 4, returnAssoc → True];
```

» number of group elements calculated 18

In[ ]:=
```
AFindKeys[GA, τh, {1.7655, 2.0243}]
```

Do : Iterator {s, S$21091} does not have appropriate bounds .

Out[ ]= Key not found

Lets test to see if the inverse to τh is in this group.

In[ ]:=
```
AFindKeys[G2A, InverseTF[τh], {1.7655, 2.0243}]
```

Out[ ]= Key not found

This does not mean InverseTF[th] is not in the group. Perhaps we need a larger n.

In[ ]:=
```
G2A =
  discreteTransGroup[<|1 → σ4, 2 → η1|>, {1.7655, 2.0243}, 1.5, 6, returnAssoc → True];
```

» number of group elements calculated 38

In[ ]:=
```
AFindKeys[G2A, InverseTF[τh], {1.7655, 2.0243}]
```

Out[ ]= {{2, 1, 2, 1, 2}}

Of course, by definition of group if τv is in the group so is its inverse, but the key may be much longer.

Note also τv is in the group.

In[ ]:=
```
AFindKeys[G2A, τv, {1.7655, 2.0243}]
```

Out[ ]= {{2, 1, 1}}

Therefore we conclude that the group G1 generated by τv,τh and their inverses is a subgroup of G2.

Of course G2 is not a subgroup of G1 which contains only translations. Suppose we augment G1 by adding the rotation σ4 but not the half turn η1. We call this G11. This will also be contained in G2.

In[ ]:=
```
G11A = discreteTransGroup[<|1 → τv, 2 → τh, 3 → σ4|>,
   {1.7655, 2.0243}, 2, 4, returnAssoc → True];
```

» number of group elements calculated 49

In[ ]:=
```
AFindKeys[G11A, η1, {1.7655, 2.0243}]
```
Out[ ]=
```
{{1, 3, 3}}
```

So $\eta 1$ and $\sigma 4$ are in G11 so G2 is contained in G11 . Hence G11 = G2 since containment goes both ways.

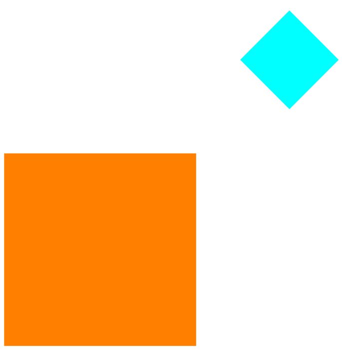
4.2.5 Conjugation in transformation groups

If we take a different square we expect to get essentially the same symmetry group. So suppose

In[ ]:=
```
ζ = TransformationFunction[{{1, -1, 8}, {1, 1, 8}, {0, 0, 4.}}]
```
Out[ ]=
```
TransformationFunction[( 1.  -1. | 8.
                         1.   1. | 8.
                         ---------------
                         0.   0. | 4. )]
```

In[ ]:=
```
ζS = ζ@S
```
Out[ ]=
```
{{2., 2.5}, {2.5, 2.}, {2., 1.5}, {1.5, 2.}}
```

In[ ]:=
```
Graphics[{{Orange, Polygon[S]}, {Cyan, Polygon[ζS]}}, ImageSize → Small]
```
Out[ ]=



This will also give a tessellation, but with different transformation functions.
We want to compare the groups but they are clearly not equal.

In group theory there is the concept of *isomorphic* groups. These are groups with different elements but the same group structure. So to a group theorist isomorphic groups are the “same”. Technically an *isomorphism* is a function f of any sort which has an inverse and preserves group structure, that is $f[a*b] = f[a]*f[b]$ for all a,b in the domain. If an isomorphism exists the groups are isomorphic. But this is too weak for us since the function could be any strange thing, we want the geometry preserved.

Within the set of all invertible transformation functions, there is a special type of function called an inner isomorphism, which is invertible and multiplication preserving. We pick a particular invertible transformation function, say κ, and define

In[ ]:=
```
Clear[α, β, κ]
```
In[ ]:=
```
Ω[α_] := κ@*α@*InverseTF[κ]
```

It is an easy exercise in algebra to show that Ω preserves multiplication and has inverse

In[ ]:=
```
℧[α_] := InverseTF[κ]@*α@*κ
```

where α is any transformation function. So any group G of invertible transformation functions is isomorphic to its image Ω[G]. This operation is called *conjugation* and the images would form *conjugate groups.*

The problem that arises is that if α is an isometry in general Ω[α] may not be. A simple example is

In[ ]:=
```
κ = TransformationFunction[RandomInteger[{-9, 9}, {3, 3}] / 10.]
```

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} -0.3 & -0.7 & 0.1 \\ -0.4 & 0.4 & -0.2 \\ \hline 0.9 & -0.5 & 0.3 \end{array}\right)\right]$

In[ ]:=
```
κ @* τv @* InverseTF[κ]
```

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 5.8 & 23.4 & 12. \\ 6.4 & 32.2 & 16. \\ \hline -14.4 & -70.2 & -35. \end{array}\right)\right]$

This is neither a translation or even an isometry . For our purposes we will stick to similarity transforms of the form

In[ ]:=
```
{{a, b, c}, {d, e, f}, {0, 0, s}} // MatrixForm
```

Out[ ]//MatrixForm=

$$\begin{pmatrix} a & b & c \\ d & e & f \\ 0 & 0 & s \end{pmatrix}$$

where the upper left 2×2 matrix is orthogonal and $s > 0$. The only difference from our isometries is that the lower right entry is s instead of 1. Then polygons are sent to similar polygons, not necessarily congruent. For example in the graphic above the orange square is sent to the cyan square by

In[ ]:=
```
ζ
```

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 1. & -1. & 8. \\ 1. & 1. & 8. \\ \hline 0. & 0. & 4. \end{array}\right)\right]$

Note if $0 < s < 1$ then the image will be larger but if $s > 1$ then the image will be smaller.

Let

In[ ]:=
```
ζi = InverseTF[ζ]
```

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 0.5 & 0.5 & -2. \\ -0.5 & 0.5 & 0. \\ \hline 0. & 0. & 0.25 \end{array}\right)\right]$

Our symmetries in the example of §4.1 then are mapped to
τv ↦ $\tau\zeta$v, τh ↦ $\tau\zeta$hwhere

In[ ]:=
```
τζv = ζ@*τv@*ζi
```

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 1. & 0. & 0.5 \\ 0. & 1. & 0.5 \\ \hline 0. & 0. & 1. \end{array}\right)\right]$

In[ ]:=
```
τζh = ζ@*τh@*ζi
```

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 1. & 0. & -0.5 \\ 0. & 1. & 0.5 \\ \hline 0. & 0. & 1. \end{array}\right)\right]$

Notice that translations stay as translations .

Our reflections ρv, ρh,ρ1 are mapped

In[ ]:=
```
ρζv = ζ@*ρv@*ζi
```

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 0. & 1. & 0. \\ 1. & 0. & 0. \\ \hline 0. & 0. & 1. \end{array}\right)\right]$

Note that this is a reflection, in fact ρd.

In[ ]:=
```
ρζd = ζ@*ρd@*ζi
```

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} -1. & 0. & 4. \\ 0. & 1. & 0. \\ \hline 0. & 0. & 1. \end{array}\right)\right]$

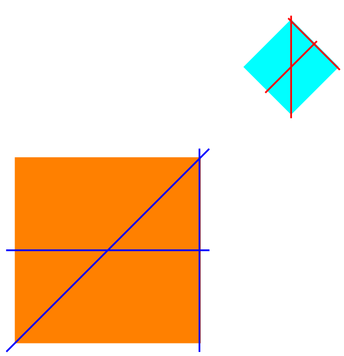
In[ ]:=
```
ρζ1 = ζ@*ρ1@*ζi
```

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 0. & -1. & 4.5 \\ -1. & 0. & 4.5 \\ \hline 0. & 0. & 1. \end{array}\right)\right]$

which are also easily seen as reflections. Here is a picture of the mirror lines

In[ ]:=
```
Graphics[{{Orange, Polygon[S]}, {Cyan, Polygon[ζS]},
  {Blue, Thickness[.005], Line[{{-1.1, 0}, {1.1, 0}}], Line[{{-1.1, -1.1}, {1.1, 1.1}}],
   Line[{{1, -1.1}, {1, 1.1}}]}, {Red, Thickness[.005], Line[ζ@{{-1.1, 0}, {1.1, 0}}],
   Line[ζ@{{-1.1, -1.1}, {1.1, 1.1}}], Line[ζ@{{1, -1.1}, {1, 1.1}}]}}, ImageSize → Small]
```

Out[ ]=



Finally note our rotations

In[ ]:= ηζ1 = ζ @* η1 @* ζi

Out[ ]= TransformationFunction[$\left(\begin{array}{cc|c} -1. & 0. & 4.5 \\ 0. & -1. & 4.5 \\ \hline 0. & 0. & 1. \end{array}\right)$]

is clearly a half turn while.

In[ ]:= σζ4 = ζ @* σ4 @* ζi

Out[ ]= TransformationFunction[$\left(\begin{array}{cc|c} 0. & -1. & 4. \\ 1. & 0. & 0. \\ \hline 0. & 0. & 1. \end{array}\right)$]

Note this last symmetry preserves ζS.

In[ ]:= ζS

Out[ ]= {{2., 2.5}, {2.5, 2.}, {2., 1.5}, {1.5, 2.}}

in fact

In[ ]:= σζ4@ζS

Out[ ]= {{1.5, 2.}, {2., 2.5}, {2.5, 2.}, {2., 1.5}}

is a $\pi/2$ (90°) rotation of ζS.

Note also

In[ ]:= Gζ2 = discreteTransGroup[<|1 → σζ4, 2 → ηζ1|>, {2.3214, 3.571}, 2, 6, returnGroup → True]

» number of group elements calculated 38

Out[ ]= {{1}, {2}, {1, 1}, {1, 2}, {2, 1}, {2, 2}, {1, 1, 1}, {1, 1, 2}, {1, 2, 1}, {2, 1, 1}, {2, 1, 2}, {1, 1, 1, 2}, {1, 1, 2, 1}, {1, 2, 1, 1}, {1, 2, 1, 2}, {2, 1, 1, 1}, {2, 1, 1, 2}, {2, 1, 2, 1}, {1, 1, 1, 2, 1}, {1, 1, 2, 1, 1}, {1, 1, 2, 1, 2}, {1, 2, 1, 1, 1}, {1, 2, 1, 1, 2}, {1, 2, 1, 2, 1}, {2, 1, 1, 2, 1}, {2, 1, 2, 1, 1}, {2, 1, 2, 1, 2}, {1, 1, 1, 2, 1, 1}, {1, 1, 1, 2, 1, 2}, {1, 1, 2, 1, 1, 1}, {1, 1, 2, 1, 1, 2}, {1, 1, 2, 1, 2, 1}, {1, 2, 1, 1, 2, 1}, {1, 2, 1, 2, 1, 1}, {2, 1, 1, 2, 1, 1}, {2, 1, 1, 2, 1, 2}, {2, 1, 2, 1, 1, 1}, {2, 1, 2, 1, 1, 2}}

Compare with

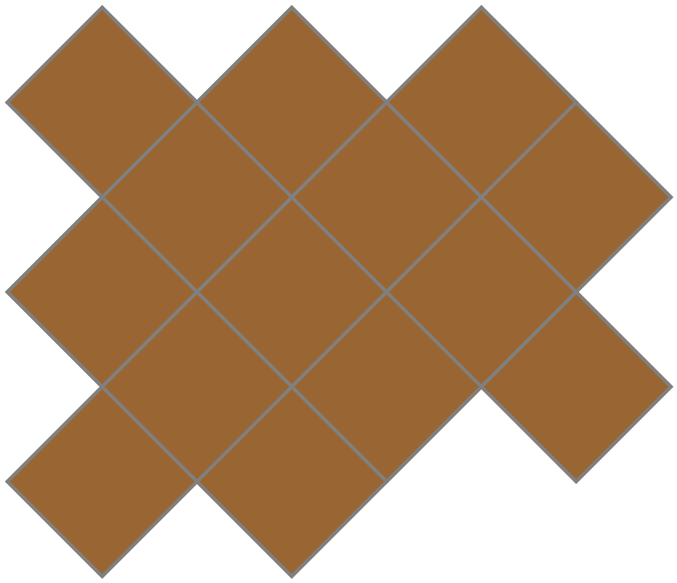
In[ ]:= G2 = finiteTransGroup[<|1 → σ4, 2 → η1|>, {2.3214, 3.571}, 6]

» number of group elements calculated 38

Out[ ]= {{1}, {2}, {1, 1}, {1, 2}, {2, 1}, {2, 2}, {1, 1, 1}, {1, 1, 2}, {1, 2, 1}, {2, 1, 1}, {2, 1, 2}, {1, 1, 1, 2}, {1, 1, 2, 1}, {1, 2, 1, 1}, {1, 2, 1, 2}, {2, 1, 1, 1}, {2, 1, 1, 2}, {2, 1, 2, 1}, {1, 1, 1, 2, 1}, {1, 1, 2, 1, 1}, {1, 1, 2, 1, 2}, {1, 2, 1, 1, 1}, {1, 2, 1, 1, 2}, {1, 2, 1, 2, 1}, {2, 1, 1, 2, 1}, {2, 1, 2, 1, 1}, {2, 1, 2, 1, 2}, {1, 1, 1, 2, 1, 1}, {1, 1, 1, 2, 1, 2}, {1, 1, 2, 1, 1, 1}, {1, 1, 2, 1, 1, 2}, {1, 1, 2, 1, 2, 1}, {1, 2, 1, 1, 2, 1}, {1, 2, 1, 2, 1, 1}, {2, 1, 1, 2, 1, 1}, {2, 1, 1, 2, 1, 2}, {2, 1, 2, 1, 1, 1}, {2, 1, 2, 1, 1, 2}}

We see that this transformation is actually compatible with our group algorithms.

In[ ]:= groupTessellate [Gζ2, <|1 → σζ4, 2 → ηζ1|> , ζS, Brown , Gray]

Out[ ]=


Important Notation: For the rest of this chapter we will use the term *isometry of groups,* or just isometry, to mean an isometry given by conjugation of a similarity transformation.

4.2.6 Finding the type of an isometry

Unfortunately this is too restrictive for mathematicians studying symmetry. For example consider parallelograms

```
In[ ]:= PG1 = {{0, 0}, {1, 1}, {2, 1}, {1, 0}};
PG2 = {{3, 0}, {5, 1}, {7, 1}, {5, 0}};
```

```
In[ ]:= Graphics [{{Orange , Polygon [PG1]}, {Cyan , Polygon [PG2]}}]
```

Out[ ]=


The tiling they would determine would both have translations and half turns but no reflections. The difference is the angles so these are not similar but would give essentially the same groups. But they would not be conjugate by a similarity transform. So as to not have an infinite number of types of tilings then mathematicians rely on describing the types of symmetries, not the actual symmetries.

It is easy to decide the type of an isometry α. If the upper left 2×2 square is the identity, we have a translation. If the determinant of the upper left hand square is -1 we have a reflection if α@∗α is the identity otherwise a glide reflection. The remaining case where the upper left hand square has determinant 1 but is not the identity we have a rotation, the order will be 2,3,4 or 6 for a full paneling. We can find this with orderTF, note we don't need to know the center. Here is an algorithm, it is meant to construct an association so in the rare case it doesn't work it sends answer "O" for "other" rather than an error message. If it works the possible results are "T" for translation, note the identity is classified as a translation, "R" for reflection, "G" for glide reflection, H for half turn and "R3","R4","R6" for rotations of higher order than 2, the number giving the order. Here is the algorithm

```
In[ ]:= IsoClassifierTF2D [α_] := Module[{M, M2, ı, n},
   M = Chop[N[TransformationMatrix [α]]];
   If[M⟦3, 1⟧ ≠ 0 || M⟦3, 2⟧ ≠ 0 || M⟦3, 3⟧ ≠ 1., Return["O"]];
   M2 = Take[M, 2, 2];
   ı = {{1., 0}, {0, 1.}};
   If[Chop[M2.Transpose [M2]] ≠ {{1., 0}, {0, 1.}}, Return["O"]];
   If[M2 == ı, Return["T"]];
   If[Det[M2] == -1.,
    If[Chop[M.M] == {{1., 0, 0}, {0, 1., 0}, {0, 0, 1.}}, Return["RF"], Return["G"]]];
   n = orderTF[α, RandomReal [{-1, 1}, 2], 7];
   Which[n == 2, Return["H"], n == 3, Return["R3"],
    n == 4, Return["R4"], n == 6, Return["R6"], True, Return["O"]];
   Return[
    "O"]]
```

If the algorithm returns "O" the user should look at that case by hand. What we are really after is

```
In[ ]:= isoClassifierA [G_, tas_] := <|Table[k → IsoClassifierTF2D [TasTF[k, tas]], {k, G}]|>
```

Here is an example

```
In[ ]:= G3 = discreteTransGroup [<|1 → ρv, 2 → σ4, 3 → γh|> ,
    {.3214 , .5713}, 1, 4, returnGroup → True];
```

» number of group elements calculated 64

```
In[ ]:= G3TA = isoClassifierA [G3, <|1 → ρv, 2 → σ4, 3 → γh|>]

Out[ ]= <|{1} → RF, {2} → R4, {3} → G, {1, 1} → T, {1, 2} → RF, {1, 3} → H, {2, 1} → RF,
 {2, 2} → H, {2, 3} → G, {3, 1} → H, {3, 2} → G, {3, 3} → T, {1, 2, 1} → R4, {1, 2, 2} → RF,
 {1, 2, 3} → R4, {1, 3, 1} → G, {1, 3, 2} → R4, {1, 3, 3} → RF, {2, 1, 3} → R4,
 {2, 2, 3} → RF, {2, 3, 1} → R4, {2, 3, 2} → RF, {2, 3, 3} → R4, {3, 1, 2} → R4,
 {3, 2, 1} → R4, {3, 2, 2} → RF, {3, 2, 3} → R4, {3, 3, 1} → RF, {3, 3, 2} → R4,
 {3, 3, 3} → G, {1, 2, 1, 3} → G, {1, 2, 2, 3} → T, {1, 2, 3, 1} → G, {1, 2, 3, 2} → H,
 {1, 2, 3, 3} → G, {1, 3, 1, 2} → G, {1, 3, 2, 1} → G, {1, 3, 2, 2} → T, {1, 3, 2, 3} → RF,
 {1, 3, 3, 1} → T, {1, 3, 3, 2} → G, {1, 3, 3, 3} → H, {2, 1, 3, 1} → G, {2, 1, 3, 2} → T,
 {2, 1, 3, 3} → G, {2, 2, 3, 3} → H, {2, 3, 1, 2} → T, {2, 3, 2, 3} → T, {2, 3, 3, 1} → G,
 {2, 3, 3, 2} → H, {2, 3, 3, 3} → G, {3, 1, 2, 1} → G, {3, 1, 2, 3} → G, {3, 2, 1, 3} → G,
 {3, 2, 3, 1} → RF, {3, 2, 3, 2} → T, {3, 2, 3, 3} → G, {3, 3, 1, 2} → G, {3, 3, 2, 1} → G,
 {3, 3, 2, 2} → H, {3, 3, 2, 3} → G, {3, 3, 3, 1} → H, {3, 3, 3, 2} → G, {3, 3, 3, 3} → T|>
```

So this group contains every type except rotation of order 3 or 6.

One thing we can do is select the translation Transforms in this group and find their translation vector.

```
In[ ]:= SG3TA = Select[G3TA, # == "T" &]
Out[ ]= <|{1, 1} → T, {3, 3} → T, {1, 2, 2, 3} → T, {1, 3, 2, 2} → T, {1, 3, 3, 1} → T,
  {2, 1, 3, 2} → T, {2, 3, 1, 2} → T, {2, 3, 2, 3} → T, {3, 2, 3, 2} → T, {3, 3, 3, 3} → T|>
```

```
In[ ]:= Table[TasTF[k, <|1 → ρv, 2 → σ4, 3 → γh|>]@{0, 0}, {k, Keys[SG3TA]}]
Out[ ]= {{0., 0.}, {2., 0.}, {1., 0.}, {-1., 0.},
  {-2., 0.}, {0., -1.}, {0., 1.}, {1., 1.}, {1., -1.}, {4., 0.}}
```

This will be useful information for the classification of groups. The translations of vector {1,0} and {0,1} appear to generate the translation subgroup.

Likewise we can find the locations of the half turns .

```
In[ ]:= SG3R2A = Select[G3TA, # == "H" &]
Out[ ]= <|{1, 3} → H, {2, 2} → H, {3, 1} → H, {1, 2, 3, 2} → H, {1, 3, 3, 3} → H,
  {2, 2, 3, 3} → H, {2, 3, 3, 2} → H, {3, 3, 2, 2} → H, {3, 3, 3, 1} → H|>
```

```
In[ ]:= η = TasTF[{1, 3, 3, 3}, <|1 → ρv, 2 → σ4, 3 → γh|>]
Out[ ]= TransformationFunction[( -1.   0. | -3.
                                  0.  -1. |  0.
                                 ---------+-----
                                  0.   0. |  1. )]
```

```
In[ ]:= SolveValues[η@{x, y} == {x, y}, {x, y}][[1]]
Out[ ]= {-1.5, 0.}
```

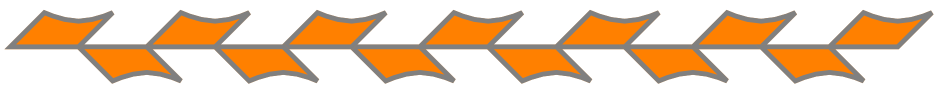
```
In[ ]:= Table[SolveValues[TasTF[k, <|1 → ρv, 2 → σ4, 3 → γh|>]@{x, y} == {x, y}, {x, y}][[1]],
   {k, Keys[SG3R2A]}]
Out[ ]= {{-0.5, 0.}, {0, 0}, {0.5, 0.}, {0., 0.5}, {-1.5, 0.}, {-1., 0.}, {0., 1.}, {1., 0.}, {1.5, 0.}}
```

4.3 Symmetry groups of symmetric strip patterns

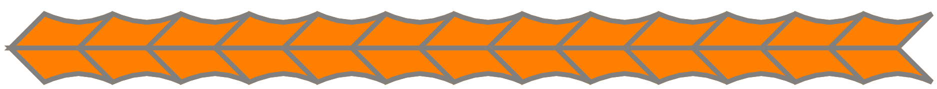
This is the one dimensional case. I don't know what the proper definition of a strip pattern is, but Mathematicians are sure that there are 7 of them. In practice there are probably infinitely many. Unlike the two dimensional case it is not clear how group theory is helpful here but there are seven groups of transformations that have been identified. Essentially these are the subgroups of the finite rectangle group with a translation and maybe glide reflection added. The generators follow

1. Horizontal translation only

2. Horizontal translation and horizontal glide reflection, no horizontal reflection



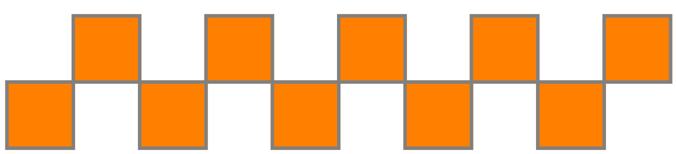
3. Horizontal translation and horizontal reflection with x-axis mirror



4. Horizontal translation and vertical reflection with y-axis mirror

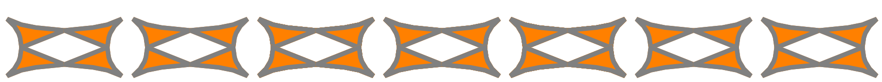
5. Horizontal translation and half turn.

6. Horizontal translation, half turn, vertical reflection, horizontal glide reflection



7. Horizontal translation, half turn, vertical reflection, horizontal reflection.

or, less boring



4.4 Groups of symmetry groups of tilings.

Here we classify only by types of isometries present. In later sections we will go into more details and examples. Note however that certain choices allow additional symmetries present, for example every plane tessellation group contains 2 independent translations, or a translation and glide reflection. Remember, different generator systems often give the same group and, even if not the same group the same kinds of isometries. For example the full triangle group gives rotations of order 6 due to rotations of order 2 and 3, since powers of the rotation of order 6 give rotations of orders 2 and 3 we see that assuming a generator of order 6 gives no further examples than we had with rotations of orders 2and 3. We also not distinguish the types of “cell” giving tessellations. Another distinction made by the crystallographers is whether there is only one family of reflections or more than one. But the composi - tion of two non-parallel reflections is a rotation so that is not a distinct case.

This will be discussed in further subsections. Thus we have only 11 types of groups instead of the usual 17 crystallographic types.

It is important to note that our isometries by similarities preserve type so any group isometric to one of our 11 groups is in that class of groups. In particular if P1, P2 are similar polygons then their symmetry groups will be of the same class.

The following explicit transformations will be standard, others will be defined as needed. The types are {RF, G, H, R3, R4, R6}

```
In[ ]:= ι2 := TransformationFunction [IdentityMatrix [3]]
τh := TranslationTransform [{1, 0}]
τv := TranslationTransform [{0, 1}]
τd := τh@* τv
ρh := reflectionTF2D [{{-1, 0}, {1, 0}}]
ρv := reflectionTF2D [{{0, -1}, {0, 1}}]
γh := glideReflectionTF2D [{{0, 0}, {1, 0}}]
η0 := RotationTransform [Pi]
η1 := RotationTransform [Pi, {1, 0}]
σ3 := RotationTransform [2 Pi / 3]
σ4 := RotationTransform [Pi / 2]
σ6 := RotationTransform [Pi / 3]
```

I. Independent translations

<|1 → τh, 2 → InverseTF[τh], 3 → τv, 4 → InverseTF[τv|> .

T

II. Translations and Glide Reflection

```
In[ ]:= γ2 = glideReflectionTF2D[{{0, 0}, {1, .7}}];
```

```
<|1 → τh, 2 → InverseTF[τh], 3 → γ2, 4 → InverseTF[γ2]|>
```

T, G

III .Translations, reflections and glide reflections .

```
<|1 → τd, 2 → InverseTF[τd], 3 → ρh |>
```

Types T, RF, G

IV. Translations and half turns only.

```
<|1 → τh, 2 → τv, 3 → η0 |>
```

Types T, H

V. Translations, and reflections , glide reflections, half turns

```
<|1 → τh, 2 → τv, 3 → ρh, 4 → ρv|>
```

Alternate

```
γh2 = glideReflectionTF2D[{{.5, 1}, {1.5, 1}}];
τv2 = τv@* τv;
```

```
<|1 → γh2, 2 → τv2, 3 → ρh, 4 → ρv|>
```

Types T, G, RF, H

VI. Orthogonal Translations and order 2, 4 rotations only

```
<|1 → τh, 2 → σ4|>
```

Types T, G, RF, H, R4

VII. Orthogonal Translations and order 2, 4, rotations and selected reflections, glide reflections. (Full Square Group).

```
<|1 → τv, 2 → σ4, 3 → ρv|>
```

Types T, RF, G, H, R4

VIII. Translations and rotations of order 3 only .

```
<|1 → τh, 2 → σ3|>
```

Types T, R3

IX. Translations, reflections, glide reflections, rotations of order 3 only .

<| 1 → τh, 2 → σ3, 3 → ρh|>

Types T, RF,G, R3,

X.Translations, half turns, rotations of order 2, 3, and 6.

<| 1 → τh, 2 → σ3, 3 → ηΘ |>

Types T, H, R3, R6

XI.Translations, reflections and rotations order 2,3,6 and glide reflections. (full Triangle group)

<| 1 → τh, 2 → σ3, 3 → ηΘ , 4 → ρv|>

T, RF, G, H, R3, R6

4.4.1 Notes on calculation

We assume, with no loss of generality that the translation τh is in every tessellation group. By conjuga - tion and other compositions there will be many translations. Enumerating this does not seem to add information, there will be some mention in the discussion in 4.5.

We then have 6 other types of isometries which may be in a tessellation symmetry group of a tiling, abbreviated RF, G, H, R3, R4, R6, recall H is the half-turn otherwise known as R2. We classify these groups by which of these are present. There are $2^6 = 64$ possible combinations and I have checked each to see if they give a discrete group different from others. There are two basic rules which allow me to discard possible groups quickly.

The inclusionary rule, “inclusion” , has the following

If there is an R4 there must be an H, its square.
If there is an R6 there must be an H and an R3 the square and cube.
If there is an H and a R3 there is an R6, even with different centers.

The exclusionary rule, “exclusion” says

There cannot be both an R3 and R4 in the same group.
There cannot be both an R6 and R4 in the same group.

The reason for these is that the upper left hand 2×2 matrix in a rotation transform is an orthogonal matrix of determinant 1. The 2×2 rotations commute. Therefore the order of the composition of two rotations is the product of the orders.

One thing we will notice is that group **II** contains glide reflections but not reflections, whereas group **III** which contains reflections also has glide reflections. If one has a reflection then conjugation will generally give a translation in the same direction so there will be a glide reflection. But to make a reflection from a given glide reflection requires not only a translation in the same direction but of the same length. This may not happen.

4.5 Examples and properties of the groups

4.5.1 Group I, two independent translations

This is the most general and perhaps then the most useful group, especially when used as a construction group rather than a full symmetry group.

The group is always uncomplicated, group theoretically it is simply the product of two copies of the integers. So if *p*, *q* are two points, viewed as 2- vectors they define two translations

```
τp = TranslationTransform[p]
τq = TranslationTransform[q]
```

Then the, necessarily discrete, group they define consists simply of translations

```
TranslationTransform[m p + n q]
```

The parallelogram

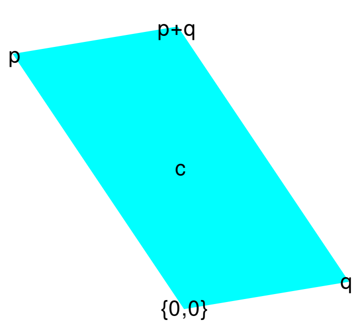
```
par = {{0, 0}, p, p + q, q}
```

is a cell for a tessellation.

as a random example

```
In[ ]:= {p, q} = {{-1.8337165994212103` , 2.736646713237377` },
          {1.7521455008048097` , 0.2910803142189895` }}
Out[ ]= {{-1.83372 , 2.73665}, {1.75215 , 0.29108}}

In[ ]:= par = {{0, 0}, p, p + q, q}
Out[ ]= {{0, 0}, {-1.83372 , 2.73665}, {-0.0815711 , 3.02773}, {1.75215 , 0.29108}}

In[ ]:= c = pcentroid[par]
Out[ ]= {-0.0407855 , 1.51386}

In[ ]:= Graphics[{{Cyan, Polygon[par]}, {Black, Text["{0,0}", {0, 0}], Text["p", p],
           Text["p+q", p + q], Text["q", q], Text["c", c]}}, BaseStyle → {FontSize → 12}]
```

Out[ ]=



The centroid is shown

In[ ]:=
```
τp = TranslationTransform[p]
τq = TranslationTransform[q]
```

Out[ ]= TransformationFunction[$\left(\begin{array}{cc|c} 1. & 0. & -1.83372 \\ 0. & 1. & 2.73665 \\ \hline 0. & 0. & 1. \end{array}\right)$]

Out[ ]= TransformationFunction[$\left(\begin{array}{cc|c} 1. & 0. & 1.75215 \\ 0. & 1. & 0.29108 \\ \hline 0. & 0. & 1. \end{array}\right)$]

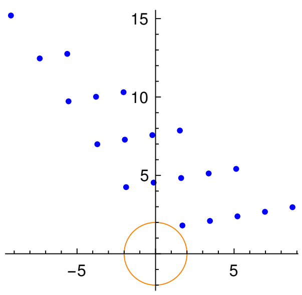
One special concern is with translations and glide reflection the inverses are not products. But groups should be closed under inverses, so they must be given. Without inverses we have

In[ ]:=
```
discreteTransGroup[<|1 → τp, 2 → τq|>, c, 2, 5]
```

» number of group elements calculated 20

Part : Part 1 of {} does not exist .

» number c points 3

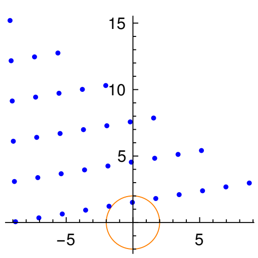
Out[ ]=


we should have gotten

In[ ]:=
```
discreteTransGroup[<|1 → τp, 2 → InverseTF[τq], 3 → τq, 4 → InverseTF[τq]|>, c, 2, 5]
```

» number of group elements calculated 36

» number c points 1

Out[ ]=


When half turns, or certain reflections or rotations, are available the inverses are come from composi - tions so don't need to be listed as generators, but here they must be.

For the full group we should use a random test point, but in this case it does not matter, for tessella - tions the centroid of the cell is what we should use.

In[ ]:=
```
G1a = discreteTransGroup[<|1 → τp, 2 → InverseTF[τp], 3 → τq, 4 → InverseTF[τq]|>,
  {.1376, .2107}, 2, 4, returnGroup → True]
```

» number of group elements calculated 41

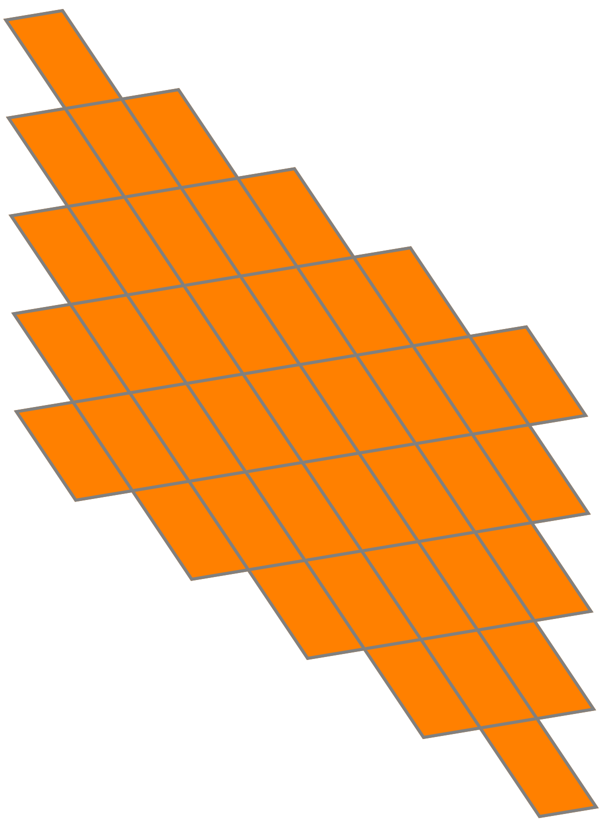
Out[ ]=
```
{{1}, {2}, {3}, {4}, {1, 1}, {1, 2}, {1, 3}, {1, 4}, {2, 2}, {2, 3}, {2, 4}, {3, 3}, {4, 4}, {1, 1, 1},
 {1, 1, 3}, {1, 1, 4}, {1, 3, 3}, {1, 4, 4}, {2, 2, 2}, {2, 2, 3}, {2, 2, 4}, {2, 3, 3},
 {2, 4, 4}, {3, 3, 3}, {4, 4, 4}, {1, 1, 1, 1}, {1, 1, 1, 3}, {1, 1, 1, 4}, {1, 1, 3, 3},
 {1, 1, 4, 4}, {1, 3, 3, 3}, {1, 4, 4, 4}, {2, 2, 2, 2}, {2, 2, 2, 3}, {2, 2, 2, 4},
 {2, 2, 3, 3}, {2, 2, 4, 4}, {2, 3, 3, 3}, {2, 4, 4, 4}, {3, 3, 3, 3}, {4, 4, 4, 4}}
```

Notice we get all combinations of the indices 1,2,3,4

Our tessellation is then

In[ ]:=
```
groupTessellate[
 G1a, <|1 → τp, 2 → InverseTF[τp], 3 → τq, 4 → InverseTF[τq]|>, par, Orange, Gray]
```

Out[ ]=



Now for this example, the half turn about the centroid is also a symmetry of the tessellation. So this group is not the full symmetry group but is the best construction group which will be group IV.

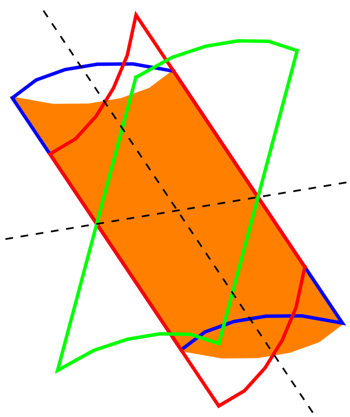
We can find a cell which has this group as a symmetry group by a little surgery.

```
par2 = {{0, -0}, p, .95 p + .2 q, .93 p + .4 q, .93 p + .6 q,
   .95 p + .8 q, p + q, q, {1.7521455008048097` , 0.2910803142189895` },
   {1.4934022306149084` , 0.09603191571332276` },
   {1.1796474624423705` , -0.016917081395222716` },
   {0.8292183622814087` , -0.0751331442390206` },
   {0.44211493013202247` , -0.07861627281807096` }, {0, 0}}
```

Out[ ]=
```
{{0, 0}, {-1.83372 , 2.73665}, {-1.3916 , 2.65803}, {-1.0045 , 2.66151},
 {-0.654069 , 2.71973}, {-0.340314 , 2.83268}, {-0.0815711 , 3.02773},
 {1.75215 , 0.29108}, {1.75215 , 0.29108}, {1.4934 , 0.0960319},
 {1.17965 , -0.0169171}, {0.829218 , -0.0751331}, {0.442115 , -0.0786163}, {0, 0}}
```

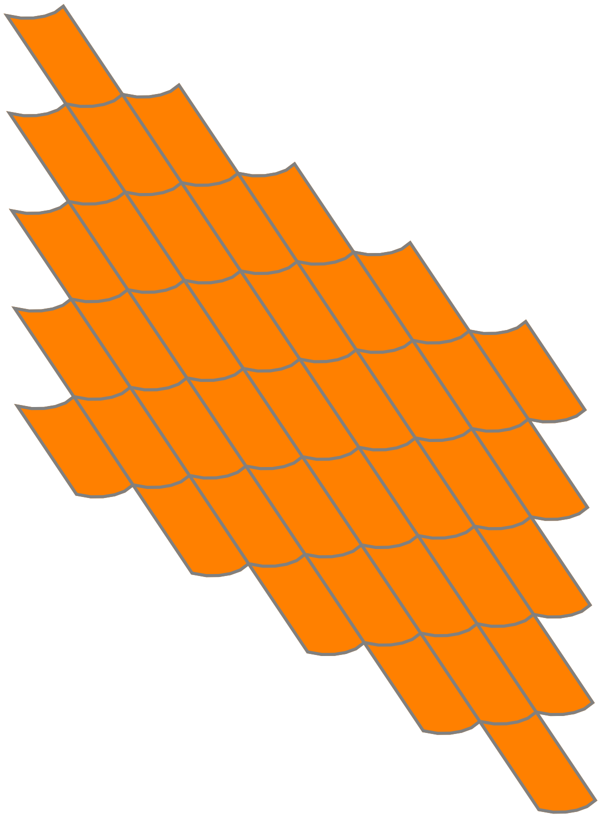
In[ ]:=
```
ηc = RotationTransform [Pi , c];
ρ1 = reflectionTF2D [{p + q / 2, q / 2}];
ρ2 = reflectionTF2D [{p / 2, p / 2 + q}];
```

In[ ]:=
```
Graphics [{{Orange , Polygon[par2]}, {Blue , Thickness [.01], Line[ηc @ par2]},
   {Red , Thickness [.01], Line[ρ1@ par2]},
   {Green , Thickness [.01], Line[ρ2@ par2]}, {Black , Thickness [.005], Dashed ,
    InfiniteLine [{p + q / 2, q / 2}], InfiniteLine [{p / 2, p / 2 + q}]}}, ImageSize → Small]
```

Out[ ]=


The graphic shows this plane figure has no symmetry. Then the tessellation

In[ ]:=
```
groupTessellate[
 G1a, <|1 → τp, 2 → InverseTF[τp], 3 → τq, 4 → InverseTF[τq]|>, par2, Orange, Gray]
```

Out[ ]=


has only translation symmetry. These graphics may be somewhat misleading but because your eyes may see these as a 3D pattern but they are only 2D so the tessellation has no reflection symmetry there are no glide reflections.

4.5.2 Group II Translations and Glide Reflections

It is possible in a tessellation group to have glide reflections and not reflections. In our discussion on Group III we will note that it is not possible to have reflections and no glide reflections.

Our example is the group generated by <| 1 → τh, 2 → InverseTF [τh], 3 → γ2, 4 → InverseTF [γ2] |> where

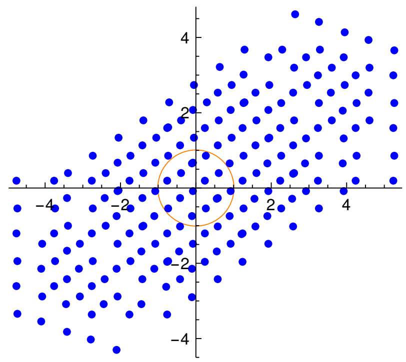
```
γ2 = glideReflectionTF2D[{{0, 0}, {1, .7}}]
```

Calculating the discrete group at a random test point we get

In[ ]:=
```
discreteTransGroup[<|1 → τh, 2 → γ2, 3 → InverseTF[τh], 4 → InverseTF[γ2]|>,
 {.2387, .1943}, 1.01, 5]
```

» number of group elements calculated 199

» number c points 15

Out[ ]=


shows not a jumble but a well organized set of points . Further levels show this to be a discrete group .

In[ ]:=
```
G2 = discreteTransGroup[<|1 → τh, 2 → γ2, 3 → InverseTF[τh], 4 → InverseTF[γ2]|>,
   {.2387, .1943}, 1.01, 4, returnGroup → True];
```

» number of group elements calculated 105

In[ ]:=
```
isoClassifierA[G2, <|1 → τh, 2 → γ2, 3 → InverseTF[τh], 4 → InverseTF[γ2]|>]
```

Out[ ]=
```
<|{1} → T, {2} → G, {3} → T, {4} → G, {1, 1} → T, {1, 2} → G, {1, 3} → T, {1, 4} → G,
 {2, 1} → G, {2, 2} → T, {2, 3} → G, {3, 2} → G, {3, 3} → T, {3, 4} → G, {4, 1} → G,
 {4, 3} → G, {4, 4} → T, {1, 1, 1} → T, {1, 1, 2} → G, {1, 1, 4} → G, {1, 2, 1} → G,
 {1, 2, 2} → T, {1, 2, 3} → G, {1, 4, 1} → G, {1, 4, 3} → G, {1, 4, 4} → T, {2, 1, 1} → G,
 {2, 1, 2} → T, {2, 1, 4} → T, {2, 2, 2} → G, {2, 2, 3} → T, {2, 3, 2} → T, {2, 3, 3} → G,
 {2, 3, 4} → T, {3, 2, 1} → G, {3, 2, 3} → G, {3, 3, 2} → G, {3, 3, 3} → T, {3, 3, 4} → G,
 {3, 4, 1} → G, {3, 4, 3} → G, {3, 4, 4} → T, {4, 1, 1} → G, {4, 1, 4} → T, {4, 3, 3} → G,
 {4, 3, 4} → T, {4, 4, 4} → G, {1, 1, 1, 1} → T, {1, 1, 1, 2} → G, {1, 1, 1, 4} → G,
 {1, 1, 2, 1} → G, {1, 1, 2, 2} → T, {1, 1, 2, 3} → G, {1, 1, 4, 1} → G, {1, 1, 4, 3} → G,
 {1, 1, 4, 4} → T, {1, 2, 1, 1} → G, {1, 2, 1, 2} → T, {1, 2, 1, 4} → T, {1, 2, 2, 2} → G,
 {1, 2, 3, 2} → T, {1, 2, 3, 3} → G, {1, 2, 3, 4} → T, {1, 4, 1, 1} → G, {1, 4, 1, 4} → T,
 {1, 4, 3, 3} → G, {1, 4, 3, 4} → T, {1, 4, 4, 4} → G, {2, 1, 1, 1} → G, {2, 1, 1, 2} → T,
 {2, 1, 1, 4} → T, {2, 1, 2, 2} → G, {2, 1, 2, 3} → T, {2, 1, 4, 3} → T, {2, 2, 2, 2} → T,
 {2, 2, 2, 3} → G, {2, 2, 3, 2} → G, {2, 2, 3, 3} → T, {2, 3, 2, 3} → T, {2, 3, 3, 2} → T,
 {2, 3, 3, 3} → G, {2, 3, 3, 4} → T, {2, 3, 4, 3} → T, {3, 2, 1, 1} → G, {3, 2, 3, 3} → G,
 {3, 3, 2, 1} → G, {3, 3, 2, 3} → G, {3, 3, 3, 2} → G, {3, 3, 3, 3} → T, {3, 3, 3, 4} → G,
 {3, 3, 4, 1} → G, {3, 3, 4, 3} → G, {3, 3, 4, 4} → T, {3, 4, 1, 1} → G, {3, 4, 1, 4} → T,
 {3, 4, 3, 3} → G, {3, 4, 3, 4} → T, {3, 4, 4, 4} → G, {4, 1, 1, 1} → G, {4, 1, 1, 4} → T,
 {4, 1, 4, 4} → G, {4, 3, 3, 3} → G, {4, 3, 3, 4} → T, {4, 3, 4, 4} → G, {4, 4, 4, 4} → T|>
```

shows we get only translations and glide reflections, no reflections or rotations .

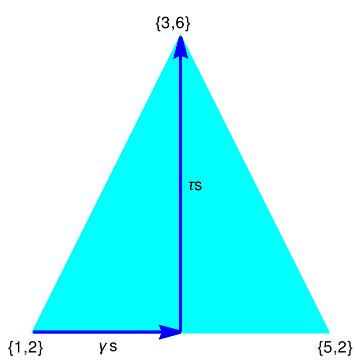
Rather than demonstrate a tessellation with this group we claim that there is a general family of tessellations, isosceles triangles. We give a pseudo random example

In[ ]:=
```
stri = {{1, 2}, {3, 6}, {5, 2}}
```
Out[ ]=
```
{{1, 2}, {3, 6}, {5, 2}}
```

We use one side to be the base of a glide reflection

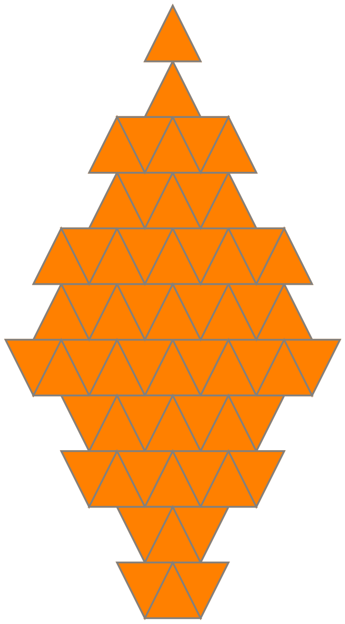
In[ ]:=
```
γs = Chop[glideReflectionTF2D [{{1, 2}, {3, 2}}]]
```
Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 1. & 0. & 2. \\ 0. & -1. & 4. \\ \hline 0. & 0. & 1. \end{array}\right)\right]$

In[ ]:=
```
τs = TranslationTransform [{0, 4}]
```
Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 1 & 0 & 0 \\ 0 & 1 & 4 \\ \hline 0 & 0 & 1 \end{array}\right)\right]$

In[ ]:=
```
Graphics [{{Cyan , Polygon [stri ]},
  {Blue , Thickness [.01], Arrowheads [.06], Arrow[{{1, 2}, {3, 2}}], Arrow[{{3, 2}, {3, 6}}]},
  {Black , Text["{1,2}", {.9, 1.8}], Text["{5,2}", {5.1, 1.8}], Text["{3,6}", {2.9, 6.2}],
   Text["γs", {2, 1.8}], Text["τs", {3.2, 4}]}}, ImageSize → Small]
```
Out[ ]=


In[ ]:=
```
Gs = finiteTransGroup [<| 1 → τs, 2 → InverseTF [τs], 3 → γs, 4 → InverseTF [γs] |> , {1, 2}, 5];
```
» number of group elements calculated 61

In[ ]:= groupTessellate[
 Gs, <|1 → τs, 2 → InverseTF[τs], 3 → γs, 4 → InverseTF[γs]|>, stri, Orange, Gray]

Out[ ]=


In section 4.6 .1 we will see that this group is useful for checkerboards.

4.5.3 Group III Translation and reflection

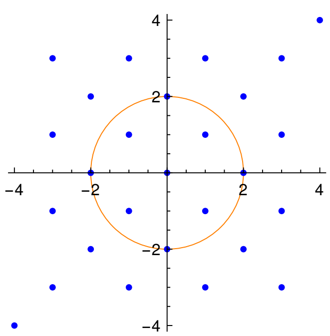
As mentioned above, when there is a reflection one can expect glide reflections also. This is true of our standard example

τd := TranslationTransform[{1, 1}]

In[ ]:= discreteTransGroup[<|1 → τd, 2 → InverseTF[τd], 3 → ρh|>, {0, -0}, 2, 4]

» number of group elements calculated 27

» number c points 5

Out[ ]=


In[∘]:=
```
G3 = discreteTransGroup [<|1 → τd, 2 → InverseTF [τd], 3 → ρh|> ,
  {0, -0}, 2, 4, returnGroup → True]
```

» number of group elements calculated 27

Out[∘]=
```
{{1}, {2}, {3}, {1, 1}, {2, 2}, {3, 1}, {3, 2}, {1, 1, 1}, {1, 3, 1}, {1, 3, 2}, {2, 2, 2}, {2, 3, 1},
 {2, 3, 2}, {3, 1, 1}, {3, 2, 2}, {1, 1, 1, 1}, {1, 1, 3, 1}, {1, 1, 3, 2}, {1, 3, 1, 1}, {1, 3, 2, 2},
 {2, 2, 2, 2}, {2, 2, 3, 1}, {2, 2, 3, 2}, {2, 3, 1, 1}, {2, 3, 2, 2}, {3, 1, 1, 1}, {3, 2, 2, 2}}
```

In[∘]:=
```
isoClassifierA [G3, <|1 → τd, 2 → InverseTF [τd], 3 → ρh|>]
```

Out[∘]=
```
<|{1} → T, {2} → T, {3} → RF, {1, 1} → T, {2, 2} → T, {3, 1} → G, {3, 2} → G,
 {1, 1, 1} → T, {1, 3, 1} → G, {1, 3, 2} → RF, {2, 2, 2} → T, {2, 3, 1} → RF,
 {2, 3, 2} → G, {3, 1, 1} → G, {3, 2, 2} → G, {1, 1, 1, 1} → T, {1, 1, 3, 1} → G,
 {1, 1, 3, 2} → G, {1, 3, 1, 1} → G, {1, 3, 2, 2} → G, {2, 2, 2, 2} → T, {2, 2, 3, 1} → G,
 {2, 2, 3, 2} → G, {2, 3, 1, 1} → G, {2, 3, 2, 2} → G, {3, 1, 1, 1} → G, {3, 2, 2, 2} → G|>
```

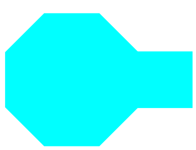
It is not surprising that there are glide reflections even though the defining reflection does not have a mirror parallel to the translation.

As our example we use for our cell paver3 which consists of an octagon with a square attached to one side.

Out[∘]=
```
{{0.707107 , -0.292893}, {1.29289 , -0.292893},
 {1.29289 , 0.292893}, {0.707107 , 0.292893}, {0.292893 , 0.707107},
 {-0.292893 , 0.707107}, {-0.707107 , 0.292893}, {-0.707107 , -0.292893},
 {-0.292893 , -0.707107}, {0.292893 , -0.707107}, {0.707107 , -0.292893}}
```

In[∘]:=
```
Graphics [{Cyan , Polygon [paver3]}, ImageSize → Tiny]
```

Out[∘]=

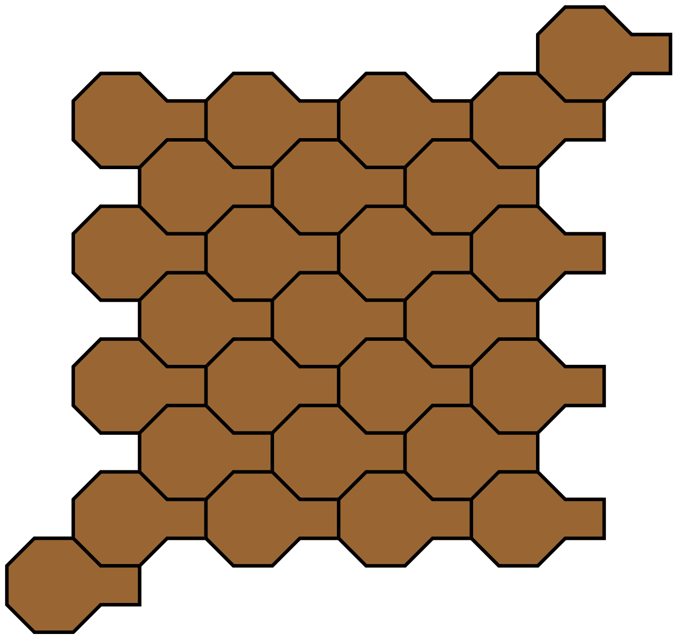


This is commercially available, for example

Our tessellation gives

In[∘]:=
```
groupTessellate[G3, <|1 → τd, 2 → InverseTF[τd], 3 → ρh|>, paver3, Brown, Black]
```

Out[∘]=



This is somewhat different from the commercial picture . In practice the square added to the octagon is separated from the octagon by a false joint. This is evident in the photo if you look for it. We will see this again and in the next section where we consider compound cells.

4.5.4 Translations and half turns

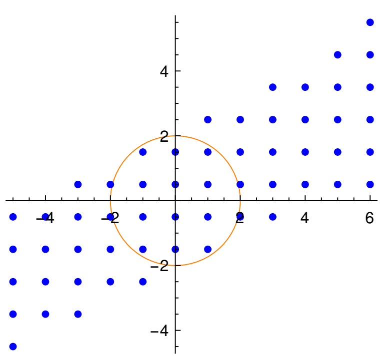
This is a group known for for parallelogram and scalene triangles tessellations.

In[∘]:=
```
discreteTransGroup[<|1 → τh, 2 → τd, 3 → η0 |>, {1, .5}, 2, 5]
```

» number of group elements calculated 56

» number c points 12

Out[∘]=



In[ ]:=
```
G4 = discreteTransGroup[<|1 → τh, 2 → τd, 3 → η0 |>, {1, .5}, 2, 4, returnGroup → True]
```

» number of group elements calculated 37

Out[ ]=
```
{{1}, {2}, {3}, {1, 1}, {1, 2}, {1, 3}, {2, 2}, {2, 3}, {3, 1}, {3, 2}, {3, 3},
 {1, 1, 1}, {1, 1, 2}, {1, 1, 3}, {1, 2, 2}, {1, 3, 2}, {2, 2, 2}, {2, 2, 3},
 {2, 3, 1}, {3, 1, 1}, {3, 1, 2}, {3, 2, 2}, {1, 1, 1, 1}, {1, 1, 1, 2}, {1, 1, 1, 3},
 {1, 1, 2, 2}, {1, 1, 3, 2}, {1, 2, 2, 2}, {1, 3, 2, 2}, {2, 2, 2, 2}, {2, 2, 2, 3},
 {2, 2, 3, 1}, {2, 3, 1, 1}, {3, 1, 1, 1}, {3, 1, 1, 2}, {3, 1, 2, 2}, {3, 2, 2, 2}}
```

In[ ]:=
```
isoClassifierA[G4, <|1 → τh, 2 → τd, 3 → η0 |>]
```

Out[ ]=
```
<|{1} → T, {2} → T, {3} → H, {1, 1} → T, {1, 2} → T, {1, 3} → H, {2, 2} → T, {2, 3} → H, {3, 1} → H,
 {3, 2} → H, {3, 3} → T, {1, 1, 1} → T, {1, 1, 2} → T, {1, 1, 3} → H, {1, 2, 2} → T, {1, 3, 2} → H,
 {2, 2, 2} → T, {2, 2, 3} → H, {2, 3, 1} → H, {3, 1, 1} → H, {3, 1, 2} → H, {3, 2, 2} → H,
 {1, 1, 1, 1} → T, {1, 1, 1, 2} → T, {1, 1, 1, 3} → H, {1, 1, 2, 2} → T, {1, 1, 3, 2} → H,
 {1, 2, 2, 2} → T, {1, 3, 2, 2} → H, {2, 2, 2, 2} → T, {2, 2, 2, 3} → H, {2, 2, 3, 1} → H,
 {2, 3, 1, 1} → H, {3, 1, 1, 1} → H, {3, 1, 1, 2} → H, {3, 1, 2, 2} → H, {3, 2, 2, 2} → H|>
```

So we just have translations and half turns, but many half turns.

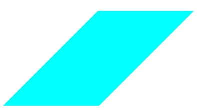
Our example of a tessellation is the parallelogram with centroid {0,0}

In[ ]:=
```
par4 = {{0, .5}, {1, .5}, {0, -.5}, {-1, -.5}}
```

Out[ ]=
```
{{0, 0.5}, {1, 0.5}, {0, -0.5}, {-1, -0.5}}
```

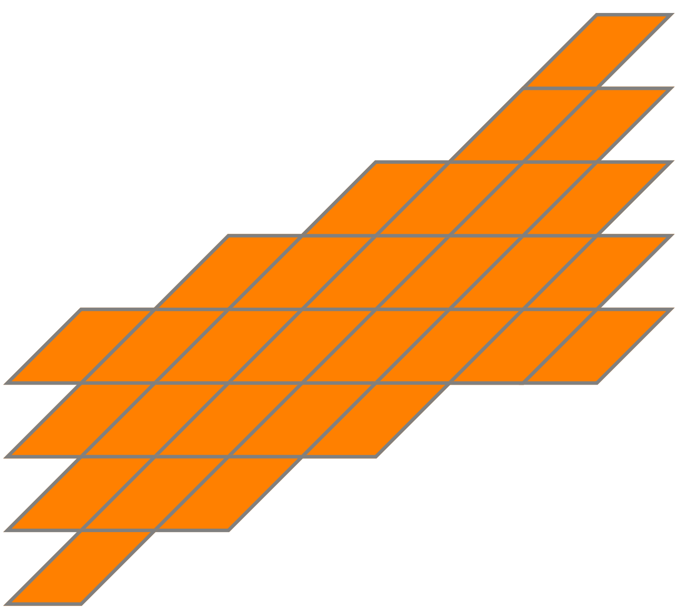
In[ ]:=
```
Graphics[{Cyan, Polygon[par4]}, ImageSize → Tiny]
```

Out[ ]=



In[ ]:=
```
groupTessellate[G4, <|1 → τh, 2 → τd, 3 → η0 |>, par4, Orange, Gray]
```

Out[ ]=



On the other hand, given a scalene triangle we can define this group using just half turns. Consider half the rectangle where the midpoints of the sides,

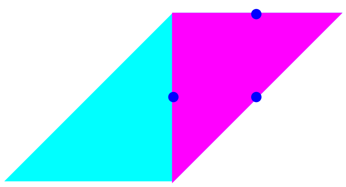
In[ ]:=
```
tri4 = {{0, 0.5`}, {1, 0.5`}, {0, -0.5`}}
```
Out[ ]=
```
{{0, 0.5}, {1, 0.5}, {0, -0.5}}
```

In[ ]:=
```
mid12 = Midpoint[{{0, .5}, {1, .5}}]
mid13 = Midpoint[{{0, .5}, {0, -.5}}]
mid23 = Midpoint[{{1, .5}, {0, -.5}}]
```
Out[ ]= $\left\{\frac{1}{2}, 0.5\right\}$

Out[ ]= {0, 0.}

Out[ ]= $\left\{\frac{1}{2}, 0.\right\}$

In[ ]:=
```
Graphics[{{Cyan, Polygon[par4]}, {Magenta, Polygon[tri4]},
  {Blue, PointSize[.03], Point[{mid12, mid13, mid23}]}}, ImageSize → Small]
```
Out[ ]=



In[ ]:=
```
η12 = RotationTransform[Pi, mid12];
η23 = RotationTransform[Pi, mid23];
```

In[ ]:=
```
G4a = discreteTransGroup[<|1 → η0, 2 → η12, 3 → η23|>, {1, .5}, 1, 4, returnGroup → True]
```
» number of group elements calculated 21

Out[ ]=
```
{{1}, {2}, {3}, {1, 1}, {2, 1}, {2, 3}, {3, 1}, {3, 2}, {1, 2, 1},
 {1, 2, 3}, {1, 3, 1}, {1, 3, 2}, {2, 3, 2}, {3, 2, 3}, {2, 1, 2, 1}, {2, 1, 2, 3},
 {2, 1, 3, 1}, {2, 3, 2, 3}, {3, 1, 3, 1}, {3, 1, 3, 2}, {3, 2, 3, 2}}
```

In[ ]:=
```
isoClassifierA[G4a, <|1 → η0, 2 → η12, 3 → η23|>]
```
Out[ ]=
```
<|{1} → H, {2} → H, {3} → H, {1, 1} → T, {2, 1} → T, {2, 3} → T, {3, 1} → T,
 {3, 2} → T, {1, 2, 1} → H, {1, 2, 3} → H, {1, 3, 1} → H, {1, 3, 2} → H,
 {2, 3, 2} → H, {3, 2, 3} → H, {2, 1, 2, 1} → T, {2, 1, 2, 3} → T, {2, 1, 3, 1} → T,
 {2, 3, 2, 3} → T, {3, 1, 3, 1} → T, {3, 1, 3, 2} → T, {3, 2, 3, 2} → T|>
```

Once again we get translations and half turns. Recall from elementary transformational geometry that the composition of 2 half turns is a translation which is shown by the 5 translations, {1,1} being the identity, given above by a doubleton.

The following computation shows that G4a is contained in G4.

In[∘]:=
```
G4A = discreteTransGroup [<| 1 → τh, 2 → τd,  3 → η0 |> ,
    {.2135 , .1057}, 3 , 4 , returnAssoc  → True];
AFindKeys [G4A , η12 , {.2135 , .1057}]
AFindKeys [G4A , η23 , {.2135 , .1057}]
```

» number of group elements calculated 47

Out[∘]= {{{2 , 3}}}

Out[∘]= {{{1 , 3}}}

Conversely

In[∘]:=
```
G4aA = discreteTransGroup [<| 1 → η0 , 2 → η12 ,  3 → η23 |> ,
    {.2135 , .1057}, 1 , 4 , returnAssoc  → True];
AFindKeys [G4aA , τh, {.2135 , .1057}]
AFindKeys [G4aA , τd, {.2135 , .1057}]
```

» number of group elements calculated 31

Out[∘]= {{{3 , 1}}}

Out[∘]= {{{2 , 1}}}

shows that G4 is contained in G4a . Thus these two infinite discrete groups are equal because they contain each others generators. Hence Group IV is also a group defined by 3 half turns.

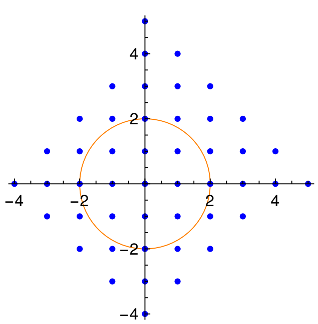
4.5.5: Translations, Reflections, half Turns

Here we don' t need to explicitly give the inverses of the translations as generators because the half turn is a generator and the inverse of any isometry is its conjugate by the half turn. Note also that **ρh** is just **η0@*ρv** so that is also included.

In[∘]:=
```
discreteTransGroup [<| 1 → τh, 2 → τv, 3 → ρv, 4 → η0 |> , {0, 0}, 2, 5]
```

» number of group elements calculated 47

» number c points 9

Out[∘]=



```
G5 =
  discreteTransGroup [<| 1 → τh, 2 → τv, 3 → ρv, 4 → η0 |> , {0, 0}, 2, 5, returnGroup  → True];
```

» number of group elements calculated 47

```
In[ ]:= isoClassifierA[G5, <|1 → τh, 2 → τv, 3 → ρv, 4 → η0|>]

Out[ ]= <|{1} → T, {2} → T, {3} → RF, {1, 1} → T, {1, 2} → T, {2, 2} → T, {3, 1} → RF,
  {4, 2} → H, {1, 1, 1} → T, {1, 1, 2} → T, {1, 2, 2} → T, {1, 4, 2} → H, {2, 2, 2} → T,
  {2, 3, 1} → G, {3, 1, 1} → RF, {4, 1, 2} → H, {4, 2, 2} → H, {1, 1, 1, 1} → T,
  {1, 1, 1, 2} → T, {1, 1, 2, 2} → T, {1, 1, 4, 2} → H, {1, 2, 2, 2} → T, {1, 4, 2, 2} → H,
  {2, 2, 2, 2} → T, {2, 2, 3, 1} → G, {2, 3, 1, 1} → G, {3, 1, 1, 1} → RF,
  {4, 1, 1, 2} → H, {4, 1, 2, 2} → H, {4, 2, 2, 2} → H, {1, 1, 1, 1, 1} → T,
  {1, 1, 1, 1, 2} → T, {1, 1, 1, 2, 2} → T, {1, 1, 1, 4, 2} → H, {1, 1, 2, 2, 2} → T,
  {1, 1, 4, 2, 2} → H, {1, 2, 2, 2, 2} → T, {1, 4, 2, 2, 2} → H, {2, 2, 2, 2, 2} → T,
  {2, 2, 2, 3, 1} → G, {2, 2, 3, 1, 1} → G, {2, 3, 1, 1, 1} → G, {3, 1, 1, 1, 1} → RF,
  {4, 1, 1, 1, 2} → H, {4, 1, 1, 2, 2} → H, {4, 1, 2, 2, 2} → H, {4, 2, 2, 2, 2} → H|>
```

In addition to the listed translations, reflections and half turns we see another example of the glide reflections created by reflections and translations.

For our example we modify this a tiny bit. Let

```
In[ ]:= τd1 = TranslationTransform[{1., .5}]
```

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 1. & 0. & 1. \\ 0. & 1. & 0.5 \\ \hline 0. & 0. & 1. \end{array}\right)\right]$

We use the group

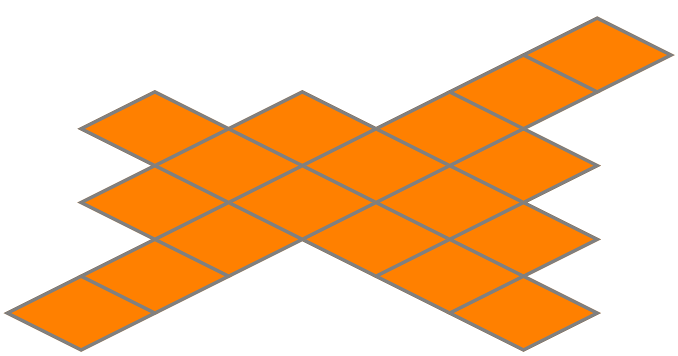
```
In[ ]:= G5M = discreteTransGroup[<|1 → τd1, 2 → ρh, 3 → η0|>, {0, 0}, 2, 4, returnGroup → True]

» number of group elements calculated 18

Out[ ]= {{1}, {2}, {1, 1}, {2, 1}, {3, 1}, {1, 1, 1}, {1, 2, 1}, {2, 1, 1}, {2, 3, 1}, {3, 1, 1}, {1, 1, 1, 1},
  {1, 1, 2, 1}, {1, 2, 1, 1}, {1, 2, 3, 1}, {2, 1, 1, 1}, {2, 3, 1, 1}, {3, 1, 1, 1}, {3, 1, 2, 1}}
```

Our cell is the diamond

```
diamond = {{-1, 0}, {0, .5}, {1, 0}, {0, -.5}}
```

Then we get the tessellation

```
In[ ]:= groupTessellate[G5M, <|1 → τd1, 2 → ρh, 3 → η0|>, diamond, Orange, Gray]
```

Out[ ]=


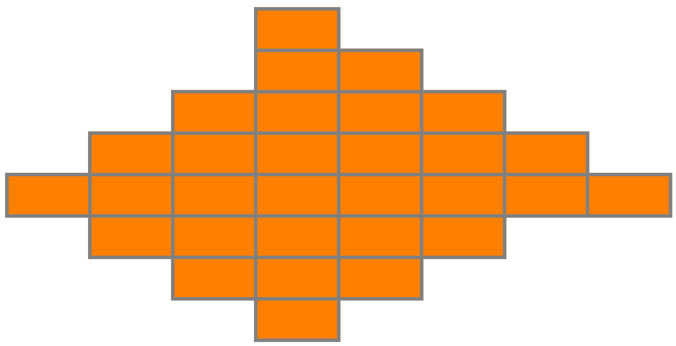
We note that this group is also the group of symmetries of a rectangle . Whereas in the diamond the mirrors are the diagonal lines between opposite vertices, in a rectangle the mirrors are between opposite midpoints. Here is an example

```
In[ ]:= rect5 = {{-.5, .25}, {.5, .25}, {.5, -.25}, {-.5, -.25}};
τvr = TranslationTransform [{0, .5}];
```

```
In[ ]:= G5M2 = discreteTransGroup [<|1 → τh, 2 → τvr, 3 → ρv, 4 → η0|>,
    {0, 0}, 2, 4, returnGroup → True];
```

» number of group elements calculated 30

```
In[ ]:= groupTessellate [G5M2, <|1 → τh, 2 → τvr, 3 → ρv, 4 → η0|>, rect5, Orange, Gray]
```

Out[ ]=


Because of this tessellation I may later call this group the rectangle group.

So far we have not used rotations other than half turns, the next group will have rotations of order 3, 4, and 6.

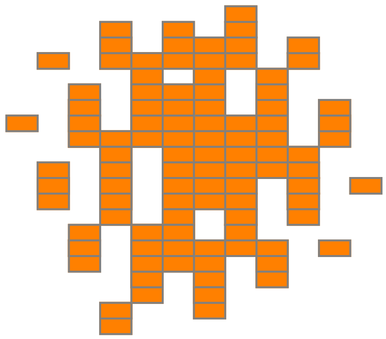
Later, when we consider compound cells we use an alternate version of Group V. Probably group theorists would classify this as a different type because it is probably not isomorphic to our standard example but since we are classifying by the types in the actual group rather than generators this still has T, G, RF ,H. The original group is generated by translations and reflections with glide reflections a consequence of these. This variation has a different glide reflection which is given as a generator.

```
γh2 = glideReflectionTF2D [{{.5, 1}, {1.5, 1}}];
τvr =
 <|1 → γh2, 2 → τv2, 3 → ρh, 4 → ρv|>.


G5a = discreteTransGroup [<|1 → γh2, 2 → τv2, 3 → ρh, 4 → ρv|>,
    {.2135, .33132}, 2, 6, returnGroup → True];
```

We get a different tessellation with our cell rect5.

In[ ]:=
```
groupTessellate [G5a, <| 1 → γh2, 2 → τvr, 3 → ρh, 4 → ρv|>, rect5, Orange, Gray]
```

Out[ ]=



See section 4.6 for more details

4.5.6 Translations and order 4 rotations

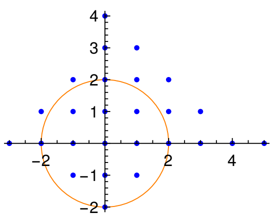
This is not a particularly useful group but must be included for completeness.

In[ ]:=
```
discreteTransGroup [<| 1 → τh, 2 → σ4|>, {0, 0}, 2, 5]
```

» number of group elements calculated 26

» number c points 9

Out[ ]=



In[ ]:=
```
G6 = discreteTransGroup [<| 1 → τh, 2 → σ4|>, {.2136, .1948}, 2, 5, returnGroup → True];
```

» number of group elements calculated 53

In[ ]:=
```
isoClassifierA [G6, <| 1 → τh, 2 → σ4|>]
```

Out[ ]=
```
<|{1} → T, {2} → R4, {1, 1} → T, {1, 2} → R4, {2, 1} → R4, {2, 2} → H, {1, 1, 1} → T, {1, 1, 2} → R4,
 {1, 2, 1} → R4, {1, 2, 2} → H, {2, 1, 1} → R4, {2, 1, 2} → H, {2, 2, 1} → H, {2, 2, 2} → R4,
 {1, 1, 1, 1} → T, {1, 1, 1, 2} → R4, {1, 1, 2, 1} → R4, {1, 1, 2, 2} → H, {1, 2, 1, 1} → R4,
 {1, 2, 1, 2} → H, {1, 2, 2, 2} → R4, {2, 1, 1, 1} → R4, {2, 1, 1, 2} → H, {2, 1, 2, 1} → H,
 {2, 1, 2, 2} → R4, {2, 2, 1, 1} → H, {2, 2, 1, 2} → R4, {2, 2, 2, 1} → R4, {2, 2, 2, 2} → T,
 {1, 1, 1, 1, 1} → T, {1, 1, 1, 1, 2} → R4, {1, 1, 1, 2, 1} → R4, {1, 1, 1, 2, 2} → H,
 {1, 1, 2, 1, 1} → R4, {1, 1, 2, 1, 2} → H, {1, 1, 2, 2, 2} → R4, {1, 2, 1, 1, 1} → R4,
 {1, 2, 1, 1, 2} → H, {1, 2, 1, 2, 2} → R4, {1, 2, 2, 2, 1} → R4, {2, 1, 1, 1, 1} → R4,
 {2, 1, 1, 1, 2} → H, {2, 1, 1, 2, 1} → H, {2, 1, 1, 2, 2} → R4, {2, 1, 2, 1, 1} → H,
 {2, 1, 2, 1, 2} → R4, {2, 1, 2, 2, 2} → T, {2, 2, 1, 1, 1} → H, {2, 2, 1, 1, 2} → R4,
 {2, 2, 1, 2, 1} → R4, {2, 2, 1, 2, 2} → T, {2, 2, 2, 1, 1} → R4, {2, 2, 2, 1, 2} → T|>
```

We see beside the generator types we also get half turns **η0 = σ4@*σ4** and others. Given half turns we get the inverse of τh so we don't need to include that as a generator. But there are no reflections or glide reflections. On the other hand all the translations are parallel to or orthogonal to τh. We get τv by

In[∘]:=
```
TasTF[{2, 1, 2, 2, 2}, <|1 → τh, 2 → σ4|>]
```

Out[∘]= TransformationFunction $\left[\left(\begin{array}{cc|c} 1 & 0 & 0 \\ 0 & 1 & 1 \\ \hline 0 & 0 & 1 \end{array}\right)\right]$

or

In[∘]:=
```
σ4@* τh@* η0 @* σ4
```

Out[∘]= TransformationFunction $\left[\left(\begin{array}{cc|c} 1 & 0 & 0 \\ 0 & 1 & 1 \\ \hline 0 & 0 & 1 \end{array}\right)\right]$

To get a tessellation without introducing reflections we use

In[∘]:=
```
τh2 = τh@* τh;
```

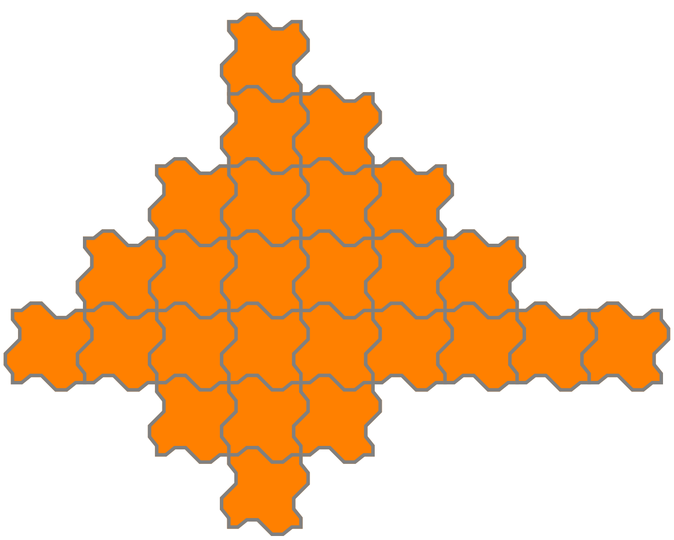
In[∘]:=
```
Q6 = {{1, 1}, {0.75` , 1}, {0.5` , 0.8`}, {0.2` , 0.8`}, {0.` , 1.`}, {-0.2` , 1.2`}, {-0.5` , 1.2`},
    {-0.75` , 1.`}, {-1.` , 1.`}, {-1.` , 1.`}, {-1.` , 0.75`}, {-0.8` , 0.5`}, {-0.8` , 0.2`},
    {-1.` , 0.`}, {-1.2` , -0.2`}, {-1.2` , -0.5`}, {-1.` , -0.75`}, {-1.` , -1.`}, {-1.` , -1.`},
    {-0.75` , -1.`}, {-0.5` , -0.8`}, {-0.2` , -0.8`}, {0.` , -1.`}, {0.2` , -1.2`},
    {0.5` , -1.2`}, {0.75` , -1.`}, {1.` , -1.`}, {1.` , -1.`}, {1.` , -0.75`}, {0.8` , -0.5`},
    {0.8` , -0.2`}, {1.` , 0.`}, {1.2` , 0.2`}, {1.2` , 0.5`}, {1.` , 0.75`}, {1.` , 1.`}};
```

We modify G6 slightly

In[∘]:=
```
G6M = discreteTransGroup [<|1 → τh2, 2 → σ4|> , {0, 0}, 2, 5, returnGroup → True];
```

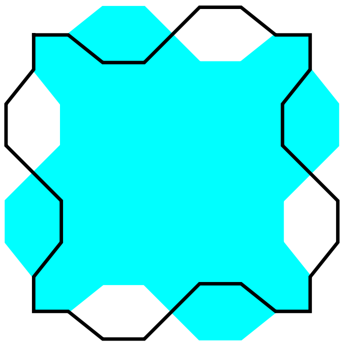
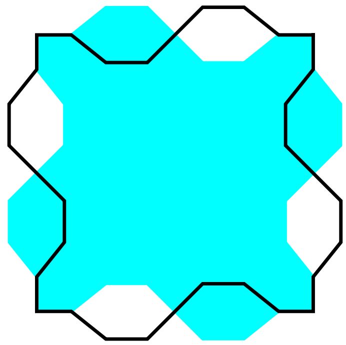
» number of group elements calculated 26

In[∘]:=
```
groupTessellate [G6M, <|1 → τh2, 2 → σ4|> , Q6, Orange , Gray]
```

Out[∘]=


Note that there are no reflection symmetries

```
In[ ]:= Row[
  Graphics[{{Cyan, Polygon[Q6]}, {Black, Thickness[.01], Line[ρv@Q6]}}, ImageSize → Small],
  Graphics[{{Cyan, Polygon[Q6]}, {Black, Thickness[.01], Line[ρd@Q6]}}, ImageSize → Small]]
```

Out[ ]= Row[ , ]

Thus G6M is not just a construction group but the symmetry group of the tessellation .

4.5.7 Group **VII** Translations, Rotations of order 4 and reflections

Our standard example is

<| 1 → τh, 2 → σ4, 3 → ρv|>

We picked ρv as our reflection but note

```
In[ ]:= ρv@* σ4
```

Out[ ]= TransformationFunction$\left[\left(\begin{array}{cc|c} 0. & 1. & 0. \\ 1. & 0. & 0. \\ \hline 0. & 0. & 1. \end{array}\right)\right]$

```
In[ ]:= ρd
```

Out[ ]= TransformationFunction$\left[\left(\begin{array}{cc|c} 0. & 1. & 0. \\ 1. & 0. & 0. \\ \hline 0. & 0. & 1. \end{array}\right)\right]$

```
In[ ]:= σ4@* σ4
```

Out[ ]= TransformationFunction$\left[\left(\begin{array}{cc|c} -1 & 0 & 0 \\ 0 & -1 & 0 \\ \hline 0 & 0 & 1 \end{array}\right)\right]$

```
In[ ]:= η0
```

Out[ ]= TransformationFunction$\left[\left(\begin{array}{cc|c} -1 & 0 & 0 \\ 0 & -1 & 0 \\ \hline 0 & 0 & 1 \end{array}\right)\right]$

```
In[ ]:= η0 @* ρv
```

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 1. & 0. & 0. \\ 0. & -1. & 0. \\ \hline 0. & 0. & 1. \end{array}\right)\right]$

```
In[ ]:= ρh
```

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 1. & 0. & 0. \\ 0. & -1. & 0. \\ \hline 0. & 0. & 1. \end{array}\right)\right]$

So we also have the half turn and reflections ρh and ρd and the other diagonal as well. We can also construct horizontal, vertical and diagonal glide reflections also. We get the other translation τv and since we have half turns we also get their inverses so we do not need to list these as generators.

The following association is as expected, translations, half turns, reflections, glide reflections and, of course order 4 rotations.

```
In[ ]:= G7 = discreteTransGroup [<| 1 → τh, 2 → σ4, 3 → ρv|> ,
          {.21331 , .1592}, 1, 4, returnGroup → True];
```

» number of group elements calculated 64

```
In[ ]:= isoClassifierA [G7, <| 1 → τv, 2 → σ4, 3 → ρv|>]
```

```
Out[ ]= <| {1} → T, {2} → R4, {3} → RF, {1, 1} → T, {1, 2} → R4, {1, 3} → G, {2, 1} → R4, {2, 2} → H,
  {2, 3} → RF, {3, 1} → G, {3, 2} → RF, {3, 3} → T, {1, 1, 1} → T, {1, 1, 2} → R4,
  {1, 1, 3} → G, {1, 2, 1} → R4, {1, 2, 2} → H, {1, 2, 3} → G, {1, 3, 2} → G,
  {2, 1, 1} → R4, {2, 1, 2} → H, {2, 1, 3} → G, {2, 2, 1} → H, {2, 2, 2} → R4,
  {2, 2, 3} → RF, {2, 3, 1} → G, {3, 1, 1} → G, {3, 1, 2} → G, {3, 1, 3} → T, {3, 2, 1} → G,
  {1, 1, 1, 1} → T, {1, 1, 1, 2} → R4, {1, 1, 1, 3} → G, {1, 1, 2, 1} → R4, {1, 1, 2, 2} → H,
  {1, 1, 2, 3} → G, {1, 1, 3, 2} → G, {1, 2, 1, 1} → R4, {1, 2, 1, 2} → H, {1, 2, 1, 3} → G,
  {1, 2, 2, 2} → R4, {1, 2, 2, 3} → RF, {1, 2, 3, 1} → G, {1, 3, 2, 1} → G, {2, 1, 1, 1} → R4,
  {2, 1, 1, 2} → H, {2, 1, 1, 3} → G, {2, 1, 2, 1} → H, {2, 1, 2, 2} → R4, {2, 1, 2, 3} → G,
  {2, 1, 3, 2} → RF, {2, 2, 1, 1} → H, {2, 2, 1, 2} → R4, {2, 2, 1, 3} → RF,
  {2, 2, 2, 1} → R4, {2, 3, 1, 1} → G, {2, 3, 1, 2} → RF, {2, 3, 1, 3} → R4, {3, 1, 1, 1} → G,
  {3, 1, 1, 2} → G, {3, 1, 1, 3} → T, {3, 1, 2, 1} → G, {3, 1, 3, 2} → R4, {3, 2, 1, 1} → G |>
```

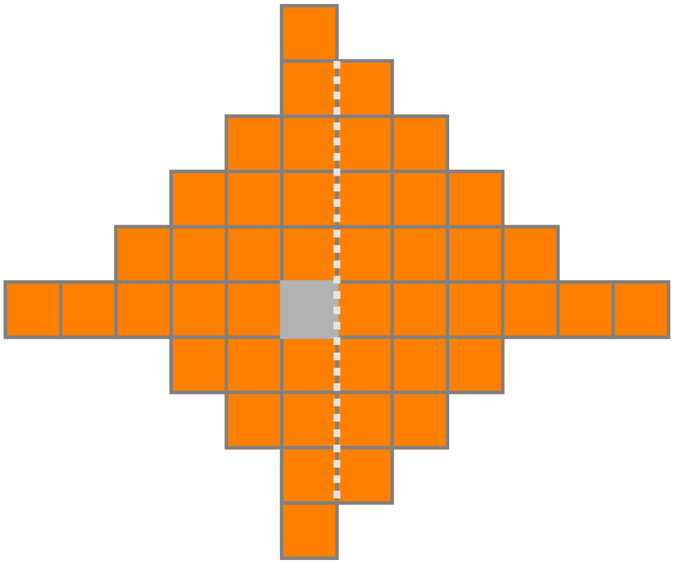
```
In[ ]:= sq = {{- .5 , .5}, {.5 , .5}, {.5 , - .5}, {- .5 , - .5}};
```

As expected our tessellation is

```
In[ ]:= G7 = discreteTransGroup [<| 1 → τh, 2 → σ4, 3 → ρv|> , {0, 0}, 1, 6, returnGroup → True];
```

» number of group elements calculated 46

In[ ]:=
```
Show[groupTessellate[G7, <|1 → τh, 2 → σ4, 3 → ρv|>, sq, Orange, Gray],
 Graphics[{{GrayLevel[.7], Polygon[sq]},
   {GrayLevel[.9], Dashed, Thickness[.01], Line[{{.5, 4.5}, {.5, -3.5}}]}}]]
```

Out[ ]=


Our original cell **sq** is shown and all the vertical and horizontal mirrors of reflections are evident, one vertical mirror is shown, it is

In[ ]:=
```
ρv7 = reflectionTF2D[{{.5, -3.5}, {.5, 4.5}}];
```

Note

In[ ]:=
```
ρv7@*ρv
```

Out[ ]=
$$\text{TransformationFunction}\left[\left(\begin{array}{cc|c} 1. & 0. & 1. \\ 0. & 1. & 0. \\ \hline 0. & 0. & 1. \end{array}\right)\right]$$

is the translation τh with inverse

In[ ]:=
```
ρv@*ρv7
```

Out[ ]=
$$\text{TransformationFunction}\left[\left(\begin{array}{cc|c} 1. & 0. & -1. \\ 0. & 1. & 0. \\ \hline 0. & 0. & 1. \end{array}\right)\right]$$

Thus an alternate set of generators for G7 is

<|1 → ρv, 2 → ρd, 3 → ρv8|>

We are done with order 4 rotations, the remaining groups concern order 3 and 6 which may have half turns but no order 4 rotations .

4.5.8 Group **VIII**, Order 3 rotations and translations only.

Our standard example is

<| 1 → τh, 2 → InverseTF[τh], 2 → σ3 |>

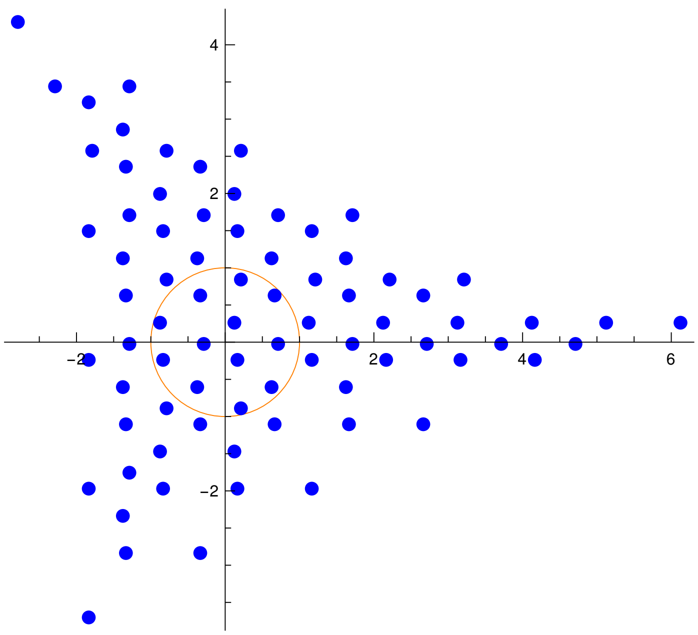
where we need to include the inverse of our translation as a generator .

In[ ]:=
```
discreteTransGroup[<|1 → τh, 2 → σ3|>, {.1238, .2617}, 1, 6]
```

» number of group elements calculated 78

» number c points 12

Out[ ]=



This does look discrete . Note that

In[ ]:=
```
Chop[σ3@* τh@* σ3@* τh@* σ3]
```

Out[ ]= TransformationFunction$\left[\left(\begin{array}{cc|c} 1. & 0. & -1. \\ 0. & 1. & 0. \\ \hline 0. & 0. & 1. \end{array}\right)\right]$

In[ ]:=
```
G8 = discreteTransGroup[<|1 → τh, 2 → σ3|>, {.1238, .2617}, 1, 5, returnGroup → True];
```

» number of group elements calculated 48

In[ ]:=
```
isoClassifierA[G8, <|1 → τh, 2 → InverseTF[τh], 3 → σ3|>]
```

Out[ ]=
```
<|{1} → T, {2} → T, {3} → R3, {1, 1} → T, {1, 2} → T, {1, 3} → R3, {2, 2} → T, {2, 3} → R3,
 {3, 1} → R3, {3, 2} → R3, {3, 3} → R3, {1, 1, 1} → T, {1, 1, 3} → R3, {1, 3, 1} → R3,
 {1, 3, 2} → R3, {1, 3, 3} → R3, {2, 2, 2} → T, {2, 2, 3} → R3, {2, 3, 1} → R3, {2, 3, 2} → R3,
 {2, 3, 3} → R3, {3, 1, 1} → R3, {3, 1, 3} → R3, {3, 2, 2} → R3, {3, 2, 3} → R3,
 {3, 3, 1} → R3, {3, 3, 2} → R3, {1, 1, 1, 1} → T, {1, 1, 1, 3} → R3, {1, 1, 3, 1} → R3,
 {1, 1, 3, 2} → R3, {1, 1, 3, 3} → R3, {1, 3, 1, 1} → R3, {1, 3, 2, 2} → R3, {1, 3, 2, 3} → R3,
 {1, 3, 3, 2} → R3, {2, 2, 2, 2} → T, {2, 2, 2, 3} → R3, {2, 2, 3, 1} → R3, {2, 2, 3, 2} → R3,
 {2, 2, 3, 3} → R3, {2, 3, 1, 1} → R3, {2, 3, 1, 3} → R3, {2, 3, 2, 2} → R3, {2, 3, 3, 1} → R3,
 {3, 1, 1, 1} → R3, {3, 1, 1, 3} → R3, {3, 1, 3, 2} → R3, {3, 1, 3, 3} → T, {3, 2, 2, 2} → R3,
 {3, 2, 2, 3} → R3, {3, 2, 3, 1} → R3, {3, 2, 3, 3} → T, {3, 3, 1, 1} → R3, {3, 3, 1, 3} → T,
 {3, 3, 2, 2} → R3, {3, 3, 2, 3} → T, {1, 1, 1, 1, 1} → T, {1, 1, 1, 1, 3} → R3,
 {1, 1, 1, 3, 1} → R3, {1, 1, 1, 3, 2} → R3, {1, 1, 1, 3, 3} → R3, {1, 1, 3, 1, 1} → R3,
 {1, 1, 3, 2, 2} → R3, {1, 1, 3, 2, 3} → R3, {1, 1, 3, 3, 2} → R3, {1, 3, 1, 1, 1} → R3,
 {1, 3, 2, 2, 2} → R3, {1, 3, 2, 2, 3} → R3, {1, 3, 2, 3, 3} → T, {1, 3, 3, 2, 2} → R3,
 {1, 3, 3, 2, 3} → T, {2, 2, 2, 2, 2} → T, {2, 2, 2, 2, 3} → R3, {2, 2, 2, 3, 1} → R3,
 {2, 2, 2, 3, 2} → R3, {2, 2, 2, 3, 3} → R3, {2, 2, 3, 1, 1} → R3, {2, 2, 3, 1, 3} → R3,
 {2, 2, 3, 2, 2} → R3, {2, 2, 3, 3, 1} → R3, {2, 3, 1, 1, 1} → R3, {2, 3, 1, 1, 3} → R3,
 {2, 3, 1, 3, 3} → T, {2, 3, 2, 2, 2} → R3, {2, 3, 3, 1, 1} → R3, {2, 3, 3, 1, 3} → T,
 {3, 1, 1, 1, 1} → R3, {3, 1, 1, 1, 3} → R3, {3, 1, 1, 3, 2} → R3, {3, 1, 1, 3, 3} → T,
 {3, 1, 3, 2, 2} → R3, {3, 1, 3, 2, 3} → T, {3, 2, 2, 2, 2} → R3, {3, 2, 2, 2, 3} → R3,
 {3, 2, 2, 3, 1} → R3, {3, 2, 2, 3, 3} → T, {3, 2, 3, 1, 1} → R3, {3, 2, 3, 1, 3} → T,
 {3, 3, 1, 1, 1} → R3, {3, 3, 1, 1, 3} → T, {3, 3, 2, 2, 2} → R3, {3, 3, 2, 2, 3} → T|>
```

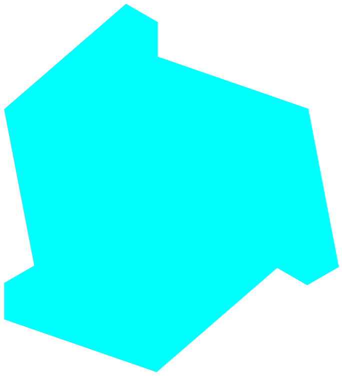
We see no other types appear to be present . Our tessellation example is given in the closed cell

In[ ]:=
```
tri8
```

Out[ ]=
```
{{-2.94668 × 10^-16, -0.57735}, {0.4, -0.23094}, {0.5, -0.288675}, {0.5, -0.288675},
 {0.6, -0.23094}, {0.5, 0.288675}, {0.5, 0.288675}, {1.01581 × 10^-16, 0.46188},
 {3.72924 × 10^-16, 0.57735}, {4.68608 × 10^-17, 0.57735}, {-0.1, 0.635085},
 {-0.5, 0.288675}, {-0.5, 0.288675}, {-0.4, -0.23094}, {-0.5, -0.288675},
 {-0.5, -0.288675}, {-0.5, -0.404145}, {-1.25117 × 10^-16, -0.57735}}
```

In[ ]:= Graphics[{Cyan, Polygon[tri8]}, ImageSize → Small]

Out[ ]=



This has no symmetries other than an order 3 rotation . Its tessellation is

In[ ]:= groupTessellate[G8, <|1 → τh, 2 → InverseTF[τh], 3 → σ3|>, tri8, Orange, Gray]

Out[ ]=

4.5.9 Group IX Translations, reflections and order 3 rotations

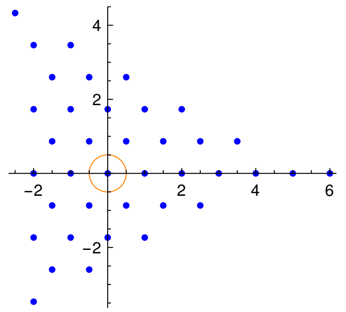
What makes this different from VIII and X is that it has a reflection but no half turn. Our example is

<|1 → τh, 2 → σ3, 3 → ρh|>

In[ ]:= discreteTransGroup[<|1 → τh, 2 → σ3, 3 → ρh|>, {0, 0}, .5, 6]

» number of group elements calculated 38

» number c points 1

Out[ ]=


We see this is an organized set of points so our group is discrete. As in Group VIII the inverse of of τh will appear so it is not necessary to include this as a generator.

In[ ]:=
```
G9 = discreteTransGroup[<|1 → τh, 2 → σ3, 3 → ρh|>,
   {.2143, .1592}, .5, 4, returnGroup → True];
```

» number of group elements calculated 46

In[ ]:=
```
isoClassifierA[G9, <|1 → τh, 2 → σ3, 3 → ρh|>]
```

Out[ ]=
```
<|{1} → T, {2} → R3, {3} → RF, {1, 1} → T, {1, 2} → R3, {1, 3} → G, {2, 1} → R3, {2, 2} → R3,
 {2, 3} → RF, {3, 2} → RF, {3, 3} → T, {1, 1, 1} → T, {1, 1, 2} → R3, {1, 1, 3} → G, {1, 2, 1} → R3,
 {1, 2, 2} → R3, {1, 2, 3} → G, {1, 3, 2} → G, {2, 1, 1} → R3, {2, 1, 2} → R3, {2, 1, 3} → G,
 {2, 2, 1} → R3, {3, 2, 1} → G, {1, 1, 1, 1} → T, {1, 1, 1, 2} → R3, {1, 1, 1, 3} → G,
 {1, 1, 2, 1} → R3, {1, 1, 2, 2} → R3, {1, 1, 2, 3} → G, {1, 1, 3, 2} → G, {1, 2, 1, 1} → R3,
 {1, 2, 1, 2} → R3, {1, 2, 1, 3} → G, {1, 2, 2, 1} → R3, {1, 3, 2, 1} → G, {2, 1, 1, 1} → R3,
 {2, 1, 1, 2} → R3, {2, 1, 1, 3} → G, {2, 1, 2, 1} → R3, {2, 1, 2, 2} → T, {2, 1, 2, 3} → G,
 {2, 1, 3, 2} → G, {2, 2, 1, 1} → R3, {2, 2, 1, 2} → T, {3, 2, 1, 1} → G, {3, 2, 1, 2} → G|>
```

Other than the generators of types T, R3,RF we have only G which is expected to appear when there is an RF. In particular there is no H.

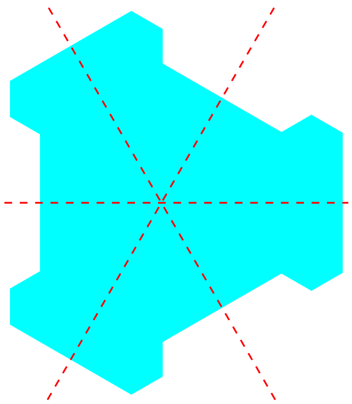
A nice example of a tessellation with just these symmetries follows.

In[ ]:=
```
Chop[panel9]
```

Out[ ]=
```
{{0, -0.57735}, {0, -0.46188}, {0.4, -0.23094}, {0.5, -0.288675}, {0.6, -0.23094},
 {0.6, 0.23094}, {0.5, 0.288675}, {0.4, 0.23094}, {0, 0.46188}, {0, 0.57735}, {0, 0.57735},
 {-0.1, 0.635085}, {-0.5, 0.404145}, {-0.5, 0.288675}, {-0.4, 0.23094}, {-0.4, -0.23094},
 {-0.5, -0.288675}, {-0.5, -0.288675}, {-0.5, -0.404145}, {-0.1, -0.635085}, {0, -0.57735}}
```

```
In[ ]:= Graphics[{{Cyan, Polygon[panel9]}, {Red, Dashed, Thickness[.005],
     InfiniteLine[{{-1, 0}, {1, 0}}], InfiniteLine[σ3@{{-1, 0}, {1, 0}}],
     InfiniteLine[σ3@* σ3@{{-1, 0}, {1, 0}}]}}, ImageSize → Small]
```

Out[ ]=



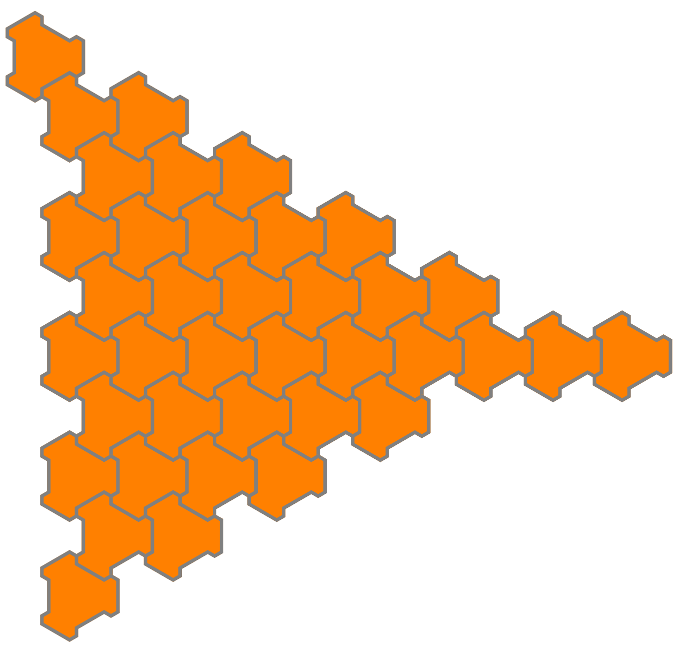
It is seen to have an order 3 rotation about its centroid {0,0} and three reflections one being ρh. Moreover there is a nice tessellation using this cell.

```
In[ ]:= G9 = discreteTransGroup[⟨|1 → τh, 2 → σ3, 3 → ρh|⟩, {0, 0}, .5, 6, returnGroup → True];
```

» number of group elements calculated 38

```
In[ ]:= groupTessellate[G9, ⟨|1 → τh, 2 → σ3, 3 → ρh|⟩, panel9, Orange, Gray]
```

Out[ ]=



4.5.10 Group **X** Translations, rotations of order 2,3,6.

This group will be called the *direct hexagon group*, it is the group of direct isometries of the hexagon.

Our basic example is

```
<|1 → τh, 2 → σ6|>
```

Note that an alternate generating set is

```
<|1 → τh, 2 → σ3, 3 → η0|>
```

In the second case the second two generators are powers of σ6 while in the first `η0@*σ3@*σ3 = σ6`

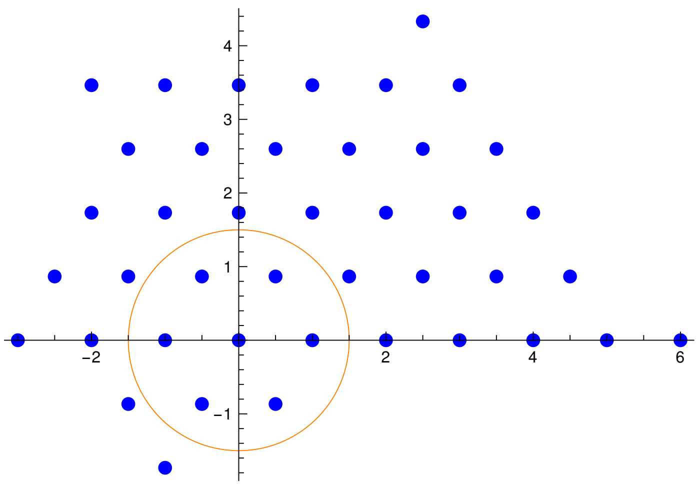
Also note that the inverse of τh is also a consequence of these generators as seen by the points {-2,0}, {-1,0} are in the orbit of {0,0}.

```
In[ ]:= discreteTransGroup [<|1 → τh, 2 → σ6|> , {0, 0}, 1.5, 6]
```

» number of group elements calculated 42

» number c points 7

Out[ ]=



also we see the points of norm 1 in the orbit of {0,0} form a hexagon

```
In[ ]:= hex1 = RecurrenceTable [{p[i + 1] == σ6@p[i], p[1] == {1, 0}}, p, {i, 6}]
```

Out[ ]= $\left\{\{1, 0\}, \left\{\frac{1}{2}, \frac{\sqrt{3}}{2}\right\}, \left\{-\frac{1}{2}, \frac{\sqrt{3}}{2}\right\}, \{-1, 0\}, \left\{-\frac{1}{2}, -\frac{\sqrt{3}}{2}\right\}, \left\{\frac{1}{2}, -\frac{\sqrt{3}}{2}\right\}\right\}$

```
In[ ]:= G10 = discreteTransGroup [<|1 → τv, 2 → σ6|> , {.2138 , .1392}, 1.5, 4, returnGroup → True];
```

» number of group elements calculated 29

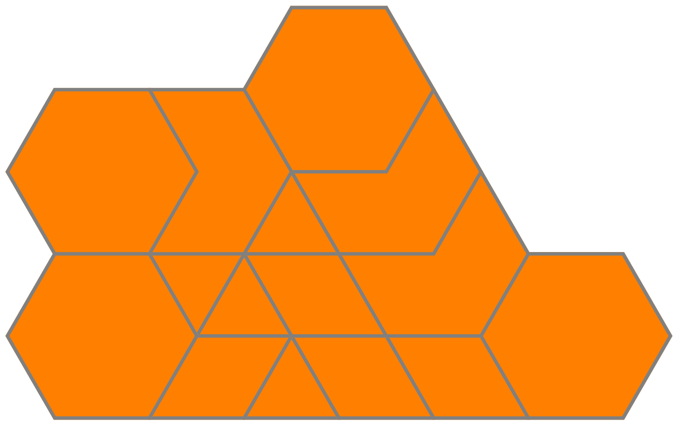
```
In[ ]:= isoClassifierA [G10, <|1 → τh, 2 → σ6|>]
```

```
Out[ ]= <|{1} → T, {2} → R6, {1, 1} → T, {1, 2} → R6, {2, 1} → R6, {2, 2} → R3, {1, 1, 1} → T, {1, 1, 2} → R6,
 {1, 2, 1} → R6, {1, 2, 2} → R3, {2, 1, 1} → R6, {2, 1, 2} → R3, {2, 2, 1} → R3, {2, 2, 2} → H,
 {1, 1, 1, 1} → T, {1, 1, 1, 2} → R6, {1, 1, 2, 1} → R6, {1, 1, 2, 2} → R3, {1, 2, 1, 1} → R6,
 {1, 2, 1, 2} → R3, {1, 2, 2, 2} → H, {2, 1, 1, 1} → R6, {2, 1, 1, 2} → R3, {2, 1, 2, 1} → R3,
 {2, 1, 2, 2} → H, {2, 2, 1, 1} → R3, {2, 2, 1, 2} → H, {2, 2, 2, 1} → H, {2, 2, 2, 2} → R3|>
```

So we just get translations and rotations of order 2, 3, 6.

The well known hexagon tessellation is not

In[ ]:= groupTessellate [G10 , <|1 → τh, 2 → σ6|> , hex1 , Orange , Gray]

Out[ ]=


Rather we let

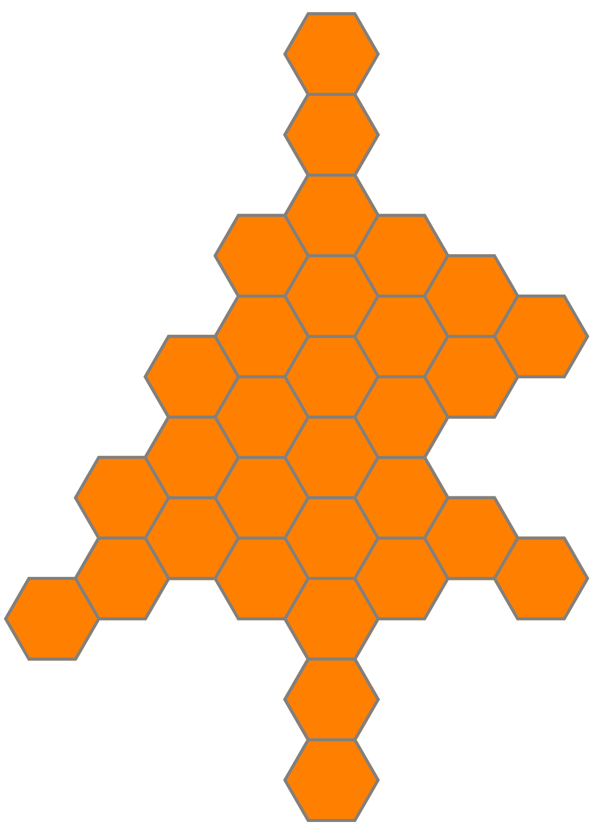
τhex := TranslationTransform [{0, $\sqrt{3.}$}]

In[ ]:= G10v = discreteTransGroup [<|1 → τhex, 2 → σ3, 3 → η0 |> , {1, 0}, 2, 4, returnGroup → True];

» number of group elements calculated 40

Then

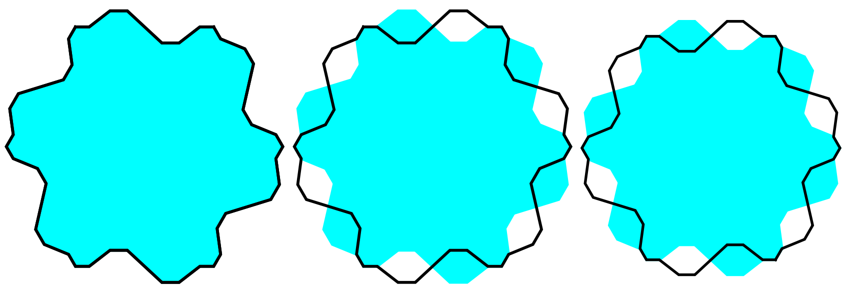
In[ ]:= groupTessellate [G10v , <|1 → τhex, 2 → σ3, 3 → η0 |> , hex1 , Orange , Gray]

Out[ ]=


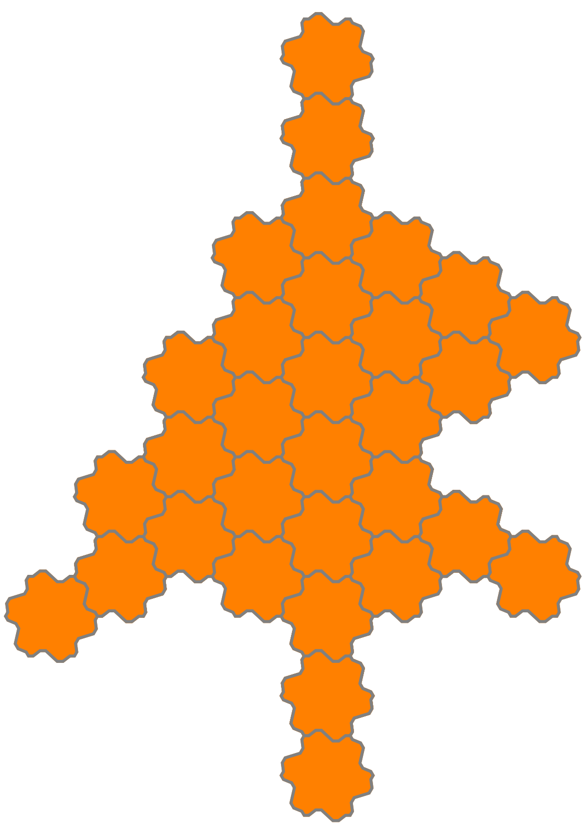
is the well known hexagonal tessellation of the plane.

This tessellation has reflections also so G10v is not the symmetry group. We can modify our cell in the usual way. We give the graphic only, the actual coordinates are in the hidden cell. Note this is fixed by σ6, but not by reflections ρv, ρh.

In[ ]:=
```
Row[{Graphics[{{Cyan, Polygon[newHex]},
     {Black, Thickness[.01], Line[newHex], Line[σ6@newHex]}}, ImageSize → 150],
  Graphics[{{Cyan, Polygon[newHex]}, {Black, Thickness[.01], Line[ρv@newHex]}},
   ImageSize → 150], Graphics[
   {{Cyan, Polygon[newHex]}, {Black, Thickness[.01], Line[ρh@newHex]}}, ImageSize → 140]}]
```

Out[ ]=


In[ ]:=
```
groupTessellate[G10v, <|1 → τhex, 2 → σ3, 3 → η0|>, newHex, Orange, Gray]
```

Out[ ]=


4.5.11 Group XI reflections, and rotations of order 2,3, 6.

This is known as the full equilateral triangle group giving all the symmetries of the triangle tessellation below. It also gives all symmetries of the hexagon tessellation above with reflections. In fact this group includes all isometry types except for order 4 rotations. The group of all isometries is not a discrete group.

Our standard example of the group is

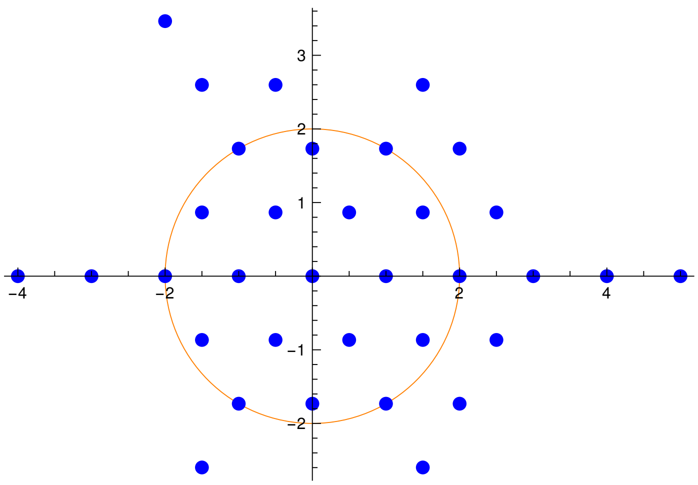
<| 1 → τh, 2 → σ3, 3 → η0, 4 → ρv |>

```
In[ ]:= discreteTransGroup [<| 1 → τh, 2 → σ3, 3 → η0, 4 → ρv |> , {0, 0}, 2, 5]
```

» number of group elements calculated 34

» number c points 13

Out[ ]=




```
In[ ]:= G11 = discreteTransGroup [<| 1 → τh, 2 → σ3, 3 → η0, 4 → ρv |> ,
     {.2137 , .1987}, 2, 3, returnGroup → True];
```

» number of group elements calculated 44

```
In[ ]:= isoClassifierA [G11, <| 1 → τh, 2 → σ3, 3 → η0, 4 → ρv |>]
```

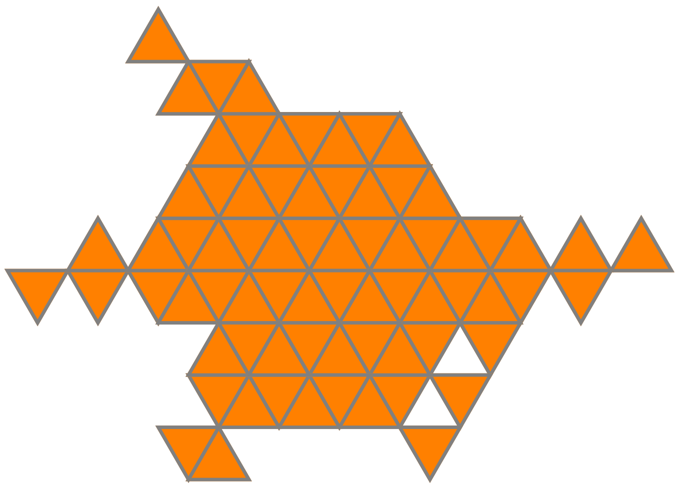
```
Out[ ]= <| {1} → T, {2} → R3, {3} → H, {4} → RF, {1, 1} → T, {1, 2} → R3, {1, 3} → H, {1, 4} → RF,
  {2, 1} → R3, {2, 2} → R3, {2, 3} → R6, {2, 4} → RF, {3, 1} → H, {3, 3} → T, {3, 4} → RF,
  {4, 1} → RF, {4, 2} → RF, {1, 1, 1} → T, {1, 1, 2} → R3, {1, 1, 3} → H, {1, 1, 4} → RF,
  {1, 2, 1} → R3, {1, 2, 2} → R3, {1, 2, 3} → R6, {1, 2, 4} → G, {1, 3, 4} → G, {1, 4, 2} → G,
  {2, 1, 1} → R3, {2, 1, 2} → R3, {2, 1, 3} → R6, {2, 1, 4} → G, {2, 2, 1} → R3,
  {2, 2, 3} → R6, {2, 3, 1} → R6, {2, 3, 4} → RF, {2, 4, 1} → G, {3, 1, 1} → H, {3, 1, 2} → R6,
  {3, 1, 3} → T, {3, 1, 4} → G, {3, 4, 2} → RF, {4, 1, 1} → RF, {4, 1, 2} → G, {4, 2, 1} → G |>
```

```
In[ ]:= etri = {{0, 0}, {1, 0}, {.5, Sqrt[3.] / 2}, {0, 0}}
```

```
Out[ ]= {{0, 0}, {1, 0}, {0.5, 0.866025}, {0, 0}}
```

In[ ]:=
```
{{0, 0}, {1, 0}, {0.5` , 0.8660254037844386` },}
```
Out[ ]=
```
{{0, 0}, {1, 0}, {0.5, 0.866025}, Null}
```
In[ ]:=
```
G11 = discreteTransGroup [<| 1 → τh, 2 → σ3, 3 → η0, 4 → ρh|> ,
    {.2213 , .3124}, 2, 5, returnGroup → True];
```
» number of group elements calculated 179

In[ ]:=
```
groupTessellate [G11, <| 1 → τh, 2 → σ3, 3 → η0, 4 → ρh|> , etri , Orange , Gray]
```
Out[ ]=



4.6 Examples of Compound Tessellations

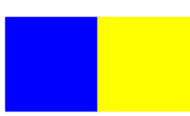
One can use our tessellation groups to tessellate the plane using more than one cell. Generally one should use a group that provides symmetry of all the cells and the union, however in most cases if there is no reason to use a bigger group one may use group I, translations. I illustrate with a number of my favorite examples.

4.6.1 Checkerboards

Our first example is the standard checker board .

In[ ]:=
```
S1 = {{0, 0}, {0, 1}, {1, 1}, {1, 0}};
S2 = {{1, 0}, {2, 0}, {2, 1}, {1, 1}};
```
The union is

In[ ]:=
```
Graphics [{{Blue , Polygon [S1]}, {Yellow , Polygon [S2]}}, ImageSize → Tiny]
```
Out[ ]=



We use group II, we will define specific translation and glide reflection.

In[ ]:=
```
τv2 = τv@* τv
γh2 = glideReflectionTF2D [{{.5 , 1}, {1.5 , 1}}]
```

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 1 & 0 & 0 \\ 0 & 1 & 2 \\ \hline 0 & 0 & 1 \end{array}\right)\right]$

Out[ ]= TransformationFunction $\left[\left(\begin{array}{cc|c} 1. & 0. & 1. \\ 0. & -1. & 2. \\ \hline 0. & 0. & 1. \end{array}\right)\right]$

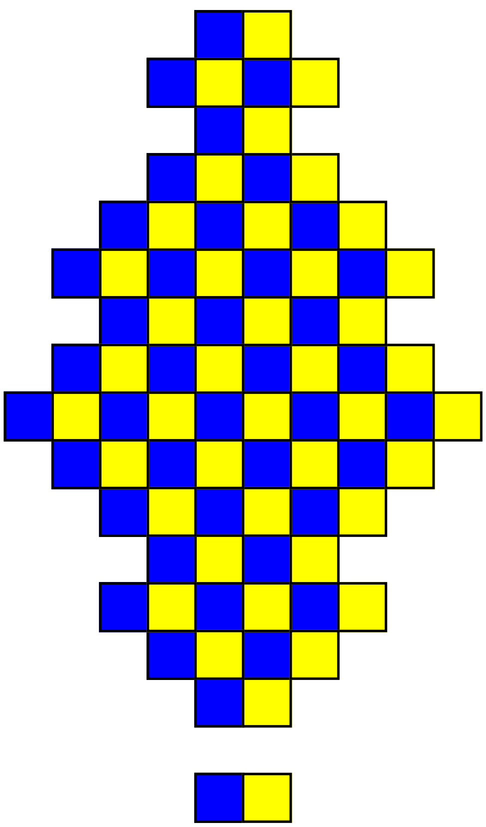
In[ ]:=
```
G2A = discreteTransGroup [<| 1 → γh2, 2 → τv2, 3 → InverseTF [τv2], 4 → InverseTF [γh2] |> ,
   {0, 0}, 2, 4, returnGroup → True];
```

» number of group elements calculated 41

It is easily checked that as in the specific example there are translations and glide transformations only. We get our tessellation.

In[ ]:=
```
Show[groupTessellate [G2A , <| 1 → γh2, 2 → τv2, 3 → InverseTF [τv2], 4 → InverseTF [γh2] |> ,
  S1, Blue , Black], groupTessellate [
  G2A , <| 1 → γh2, 2 → τv2, 3 → InverseTF [τv2], 4 → InverseTF [γh2] |> , S2, Yellow , Black]]
```

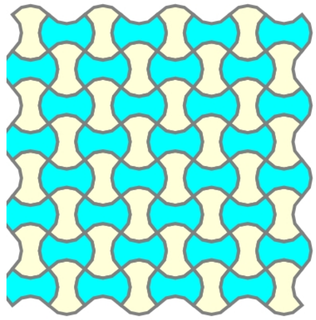
Out[ ]=



Since we are using group tessellation then we are using group elements at level 4. If we wanted just a standard 8 × 8 checker board we would use a higher level and just choose the group elements to use. This will be discussed in the next Section.

Here is my version of part of an actual pavement

In[ ]:=



It is sort of a checkerboard. The basic shape is

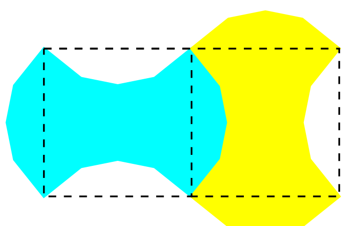
```
In[ ]:= σ2 = RotationTransform[-Pi/2, {1, 1}];
```

```
In[ ]:= bCheck1 = {{0.`, 1.`}, {0.2`, 0.75`}, {0.25`, 0.5`}, {0.2`, 0.25`}, {0.`, 0.`},
    {0.25`, -0.2`}, {0.5`, -0.25`}, {0.75`, -0.2`}, {1.`, 0.`}, {0.8`, 0.25`},
    {0.75`, 0.5`}, {0.8`, 0.75`}, {1.`, 1.`}, {0.75`, 1.2`}, {0.5`, 1.25`}, {0.25`, 1.2`}};
```

```
In[ ]:= bCheck2 = {{1.`, 2.`}, {0.75`, 1.8`}, {0.5`, 1.75`}, {0.25`, 1.8`}, {0.`, 2.`},
    {-0.2`, 1.75`}, {-0.25`, 1.5`}, {-0.2`, 1.25`}, {0.`, 1.`}, {0.25`, 1.2`},
    {0.5`, 1.25`}, {0.75`, 1.2`}, {1.`, 1.`}, {1.2`, 1.25`}, {1.25`, 1.5`}, {1.2`, 1.75`}};
```

```
In[ ]:= Graphics[{{Cyan, Polygon[σ2@bCheck1]}, {Yellow, Polygon[σ2@bCheck2]},
   {Black, Thickness[.005], Dashed, Line[{{0, 2}, {1, 2}, {1, 1}, {0, 1}, {0, 2}}],
    Line[{{0, 2}, {2, 2}, {2, 1}, {1, 1}}]}}, ImageSize → Small]
```

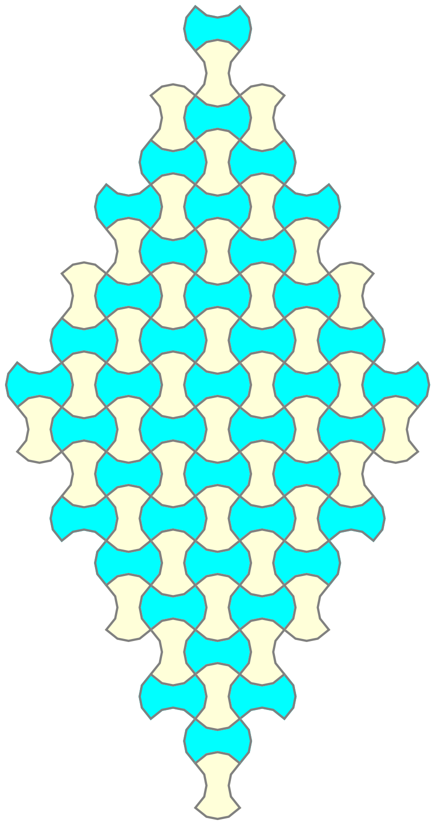
Out[ ]=



We see this is basically our checkerboard above . So using the same group

In[ ]:=
```
Show[groupTessellate [
   G2A , <| 1 → γh2, 2 → τv2, 3 → InverseTF [τv2], 4 → InverseTF [γh2] |> , bCheck2 , Cyan , Gray],
  groupTessellate [G2A , <| 1 → γh2, 2 → τv2, 3 → InverseTF [τv2], 4 → InverseTF [γh2] |> ,
   bCheck1 , LightYellow , Gray]]
```

Out[ ]=



Again we get our original picture by using a large enough level and picking cells .

We notice in both of these checkerboards that we have vertical and horizontal reflections through the centers of the cells. These in turn compose to form half turns about the midpoints of the cells. So the actual symmetry of these compound tessellations is Group V.

In[ ]:=
```
ρh5 = reflectionTF2D [{{- 1, .5}, {1, .5}}];
ρv5 = reflectionTF2D [{{.5, - 1}, {.5, 1}}];
```

In[ ]:=
```
G5a = discreteTransGroup [<| 1 → γh2, 2 → τv2, 3 → ρh5, 4 → ρv5, 5 → InverseTF [τv2] |> ,
    {.1394 , .2015}, 3, 3, returnGroup → True];
```

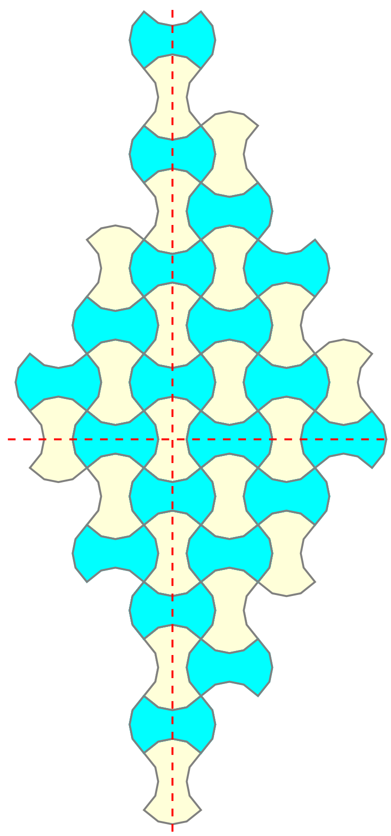
» number of group elements calculated 48

In[ ]:=
```
isoClassifierA[G5a, <|1 → γh2, 2 → τv2, 3 → ρh5, 4 → ρv5, 5 → InverseTF[τv2]|>]
```

Out[ ]=
```
<|{1} → G, {2} → T, {3} → RF, {4} → RF, {5} → T, {1, 1} → T, {1, 2} → G, {1, 3} → T, {1, 4} → H,
 {1, 5} → G, {2, 2} → T, {2, 3} → RF, {2, 4} → G, {2, 5} → T, {3, 1} → T, {3, 2} → RF, {3, 4} → H,
 {4, 1} → H, {4, 5} → G, {5, 5} → T, {1, 1, 1} → G, {1, 1, 2} → T, {1, 1, 3} → G, {1, 1, 4} → RF,
 {1, 1, 5} → T, {1, 2, 2} → G, {1, 2, 4} → H, {1, 3, 1} → G, {1, 3, 2} → T, {1, 3, 4} → G,
 {1, 4, 5} → H, {1, 5, 5} → G, {2, 2, 2} → T, {2, 2, 3} → RF, {2, 2, 4} → G, {2, 3, 4} → H,
 {2, 4, 1} → H, {3, 1, 4} → G, {3, 1, 5} → T, {3, 2, 2} → RF, {3, 2, 4} → H, {3, 4, 1} → G,
 {4, 1, 1} → RF, {4, 1, 2} → H, {4, 1, 3} → G, {4, 1, 4} → G, {4, 5, 5} → G, {5, 5, 5} → T|>
```

In[ ]:=
```
Show[groupTessellate[G5a, <|1 → γh2, 2 → τv2, 3 → ρh5, 4 → ρv5, 5 → InverseTF[τv2]|>,
   bCheck1, LightYellow, Gray],
  groupTessellate[G5a, <|1 → γh2, 2 → τv2, 3 → ρh5, 4 → ρv5, 5 → InverseTF[τv2]|>,
   bCheck2, Cyan, Gray], Graphics[{Red, Thickness[.005], Dashed,
    InfiniteLine[{{0, .5}, {1, .5}}], InfiniteLine[{{.5, 0}, {.5, 1}}]}]]
```

Out[ ]=



4.6.2 Bricks

Without alternating color or orientation we can make a brick wall by Groups IV or V.

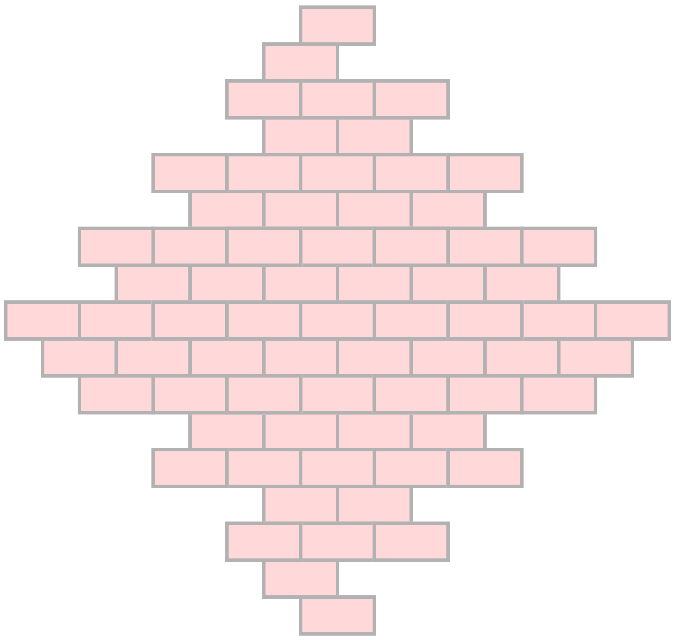
In[ ]:=
```
G5b = discreteTransGroup[<|1 → τh2, 2 → τv, 3 → ρh, 4 → ρv|>,
   {.1252, .2212}, 3, 4, returnGroup → True];
```

» number of group elements calculated 68

In[ ]:=
```
brick1 = {{-1, -.5}, {1, -.5}, {1, .5}, {-1, .5}};
brick2 = {{0, .5}, {2, .5}, {2, 1.5}, {0, 1.5}};
```

In[ ]:=
```
Show[groupTessellate [G5b, <|1 → τh2, 2 → τv2, 3 → ρh, 4 → ρv|>,
  brick1, LightRed, GrayLevel[.7]], groupTessellate [
  G5b, <|1 → τh2, 2 → τv2, 3 → ρh, 4 → ρv|>, brick2, LightRed, GrayLevel[.7]]]
```

Out[ ]=



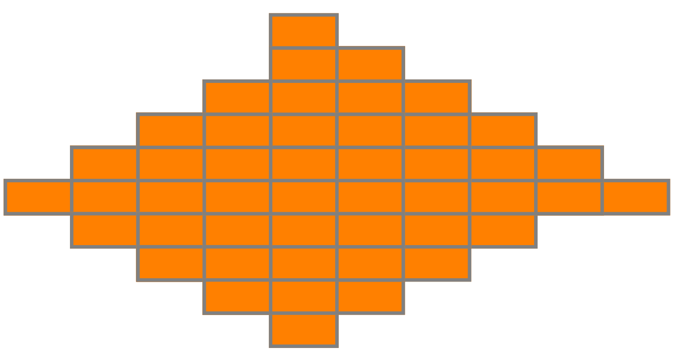
We saw above if we did not alternate we had

In[ ]:=
```
G5c = discreteTransGroup [<|1 → τh2, 2 → τv, 3 → ρv, 4 → η0|>,
   {.1249, .2137}, 2, 5, returnGroup → True];
```

» number of group elements calculated 546

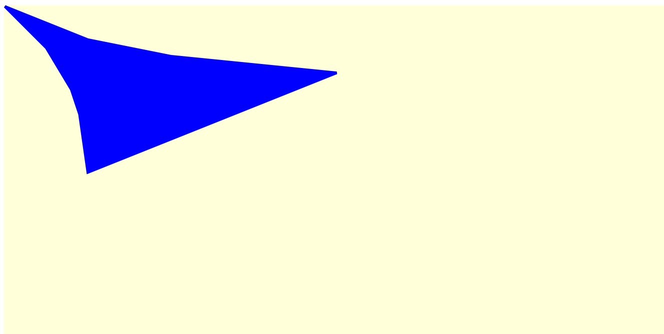
In[ ]:=
```
groupTessellate [G5c, <|1 → τh2, 2 → τv, 3 → ρv, 4 → η0|>, brick1, Orange, Gray]
```

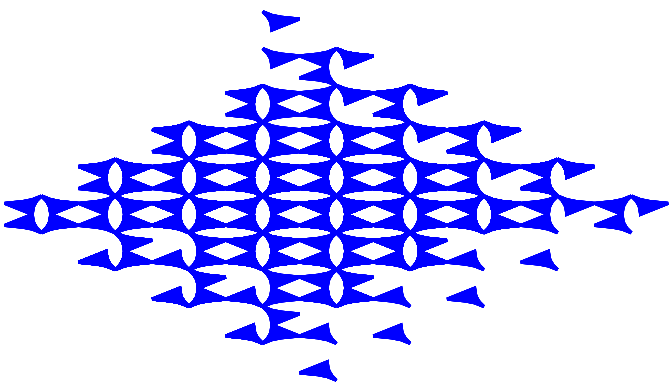
Out[ ]=



We can use this with a more complicated brick .

In[ ]:=
```
R0 = .5 {{-1.5, 0}, {-1.55, .35}, {-1.6, .5},
    {-1.75, .75}, {-2, 1}, {-1.5, .8}, {-1, .7}, {-.5, .65}, {0, .6}}
```
Out[ ]=
```
{{-0.75, 0.}, {-0.775, 0.175}, {-0.8, 0.25}, {-0.875, 0.375},
 {-1., 0.5}, {-0.75, 0.4}, {-0.5, 0.35}, {-0.25, 0.325}, {0., 0.3}}
```

In[ ]:=
```
Graphics[{{LightYellow, Polygon[brick1]}, {Blue, Polygon[R0]}}]
```
Out[ ]=


In[ ]:=
```
groupTessellate[G5c, <|1 → τh2, 2 → τv, 3 → ρv, 4 → η0|>, R0, Blue, Blue]
```
Out[ ]=


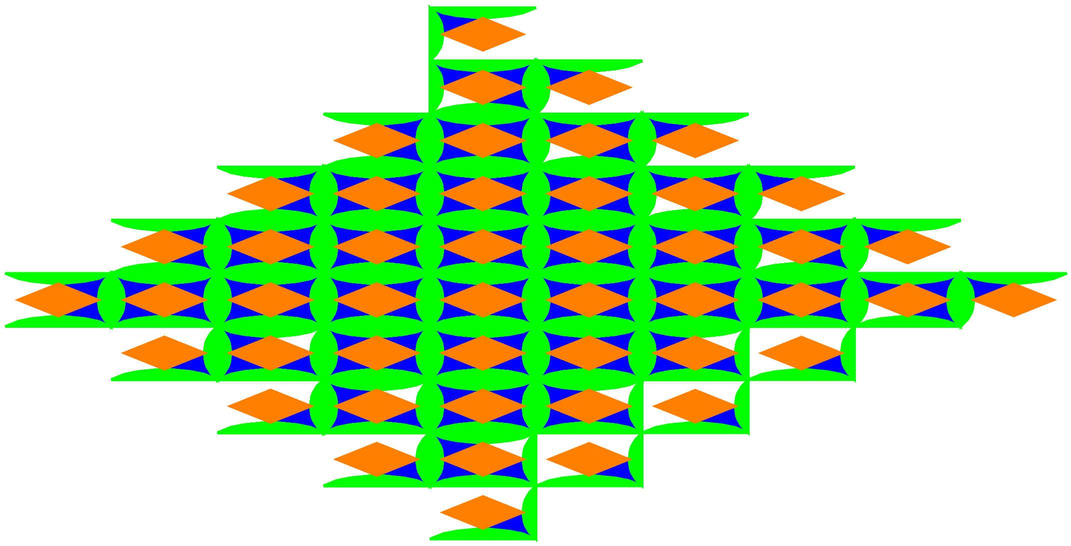
This gives us some ideas. For this we prefer not to have the border so we use

```
groupTessellateNB[G_, tas_, P_, col_] := Module[{tab},
   tab = Table[TasTF[g, tas]@P, {g, G}];
   Graphics[{col, Polygon[tab]}]]
```

```
R0a = {{-1.`, 0.5`}, {-0.75`, 0.4`}, {-0.5`, 0.35`}, {-0.25`, 0.325`}, {0.`, 0.3`}}

R2 = Append[Join[R0a, Reverse[ρv@R0a]], {-1, .5}]

R3a = Append[{{-0.75`, 0.`}, {-0.775`, 0.175`}, {-0.8`, 0.25`}, {-0.875`, 0.375`}}, {-1, .5}]

R3 = Append[Join[Reverse[R3a], ρh@R3a], {-1, .5}]

R4 = {{-.75, 0}, {0, .3}, {.75, 0}, {0, -.3}}
```

In[ ]:=
```
Show[groupTessellateNB [G5c, <| 1 → τh2, 2 → τv, 3 → ρv, 4 → η0 |> , R0, Blue],
 groupTessellateNB [G5c, <| 1 → τh2, 2 → τv, 3 → ρv, 4 → η0 |> , R2, Green],
 groupTessellateNB [G5c, <| 1 → τh2, 2 → τv, 3 → ρv, 4 → η0 |> , R3, Green],
 groupTessellateNB [G5c, <| 1 → τh2, 2 → τv, 3 → ρv, 4 → η0 |> , R4, Orange],
 ImageSize → Large]
```

Out[ ]=



4.6.4 More Bricks

A very popular use of rectangular paving stones sometimes goes by the name *Holland.* Your author had this on a section of his driveway. It is a nice example of Group IV.

In[ ]:=
```
Brick1 = {{-2, 1}, {2, 1}, {2, -1}, {-2, -1}};
Brick2 = {{-2, 1}, {-2, 5}, {0, 5}, {0, 1}};
```

Let

In[ ]:=
```
τd2 := TranslationTransform [{2, 2}]
τdm4 := TranslationTransform [{-4, 4}]
```

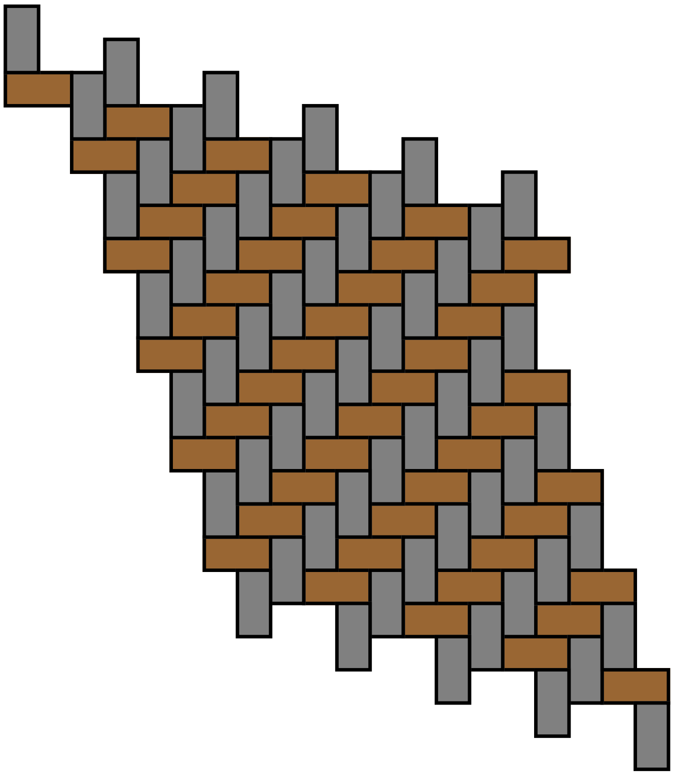
Again, to do a tessellation one needs a version of the group with a random test point. Also conjugating any translation by a half turn gives its inverse so the main importance of the half turns to get inverses for the translations. But it also shows our tessellations have half turn symmetry.

In[ ]:=
```
G4B = discreteTransGroup [<| 1 → τd2, 2 → τdm4, 3 → η0 |> ,
   {.2187 , .1326}, 2, 5, returnGroup → True];
```

» number of group elements calculated 77

In[ ]:=
```
Show[groupTesselate[G4B, <|1 → τd2, 2 → τdm4, 3 → η0|>, Brick1, Brown, Black],
 groupTesselate[G4B, <|1 → τd2, 2 → τdm4, 3 → η0|>, Brick2, Gray, Black]]
```

Out[ ]=

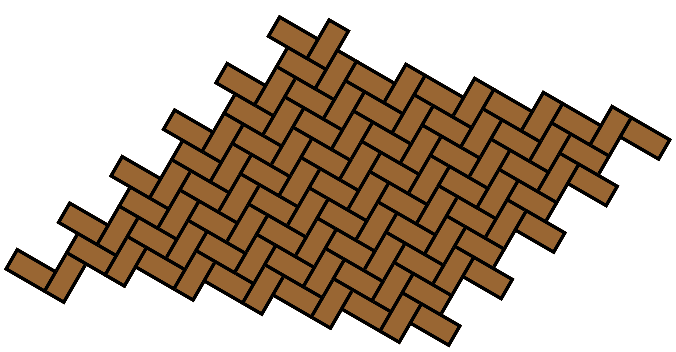


Often this is seen diagonally

In[ ]:=
```
Brick1D = σ6@Brick1;
Brick2D = σ6@Brick2;
τD2 = σ6@*τd2@*InverseTF[σ6];
τDm4 = σ6@*τdm4@*InverseTF[σ6];
```

In[ ]:=
```
Show[groupTesselate[G4B, <|1 → τD2, 2 → τDm4, 3 → η0|>, Brick1D, Brown, Black],
 groupTesselate[G4B, <|1 → τD2, 2 → τDm4, 3 → η0|>, Brick2D, Brown, Black]]
```

Out[ ]=



Note we did not need to recalculate the group since we are just conjugating. I changed the color of brick 2 so all bricks are the same as on my old driveway

4.6.5 Octagonal paver

This is commercially available, for example and similar to group III.

But here we regard the octogons and the squares separately .

```
In[ ]:= oct = {{0.7071067811865475` , -0.2928932188134522` },
    {0.7071067811865475` , 0.2928932188134525` },
    {0.2928932188134524` , 0.7071067811865475` }, {-0.2928932188134526` ,
     0.7071067811865475` }, {-0.7071067811865477` , 0.29289321881345237` },
    {-0.7071067811865475` , -0.2928932188134526` }, {-0.29289321881345226` ,
     -0.7071067811865477` }, {0.29289321881345276` , -0.7071067811865475` },
    {0.7071067811865475` , -0.2928932188134522` }};
```

```
In[ ]:= sqr = {{0.7071067811865475` , -0.2928932188134522` },
    {1.2928932188134525` , -0.2928932188134522` }, {1.2928932188134525` ,
     0.2928932188134525` }, {0.7071067811865475` , 0.2928932188134525` }};
```

Recall

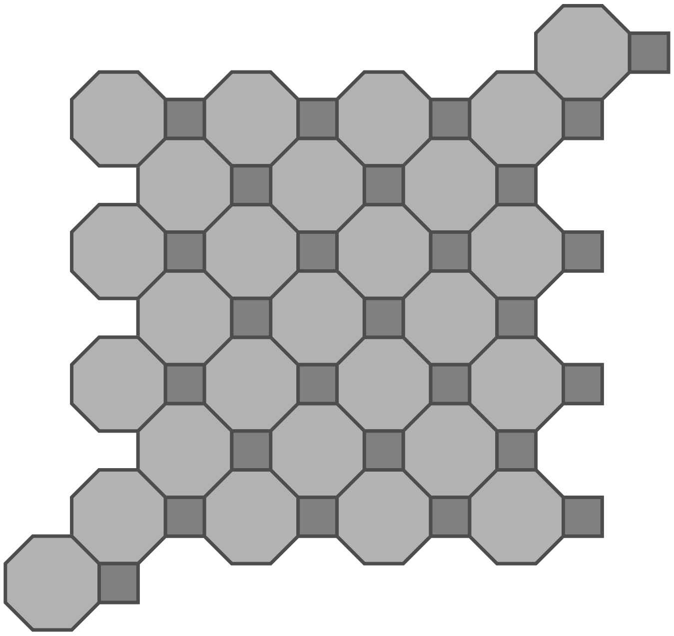
```
In[ ]:= τd = TranslationTransform [{1, 1}];
```

```
In[ ]:= G3 = discreteTransGroup [<| 1 → τd, 2 → InverseTF [τd], 3 → ρh|> ,
    {0, -0}, 2, 4, returnGroup → True];
```

» number of group elements calculated 27

In[ ]:=
```
Show[groupTessellate[G3, <|1 → τd, 2 → InverseTF[τd], 3 → ρh|>,
  oct, GrayLevel[.7], GrayLevel[.3]], groupTessellate[
  G3, <|1 → τd, 2 → InverseTF[τd], 3 → ρh|>, sqr, GrayLevel[.5], GrayLevel[.3]]]
```

Out[ ]=



4.7 Further comments on plane tessellations

This last section concerns practical considerations on actually working on tessellations so many readers may end here. Our group tessellation algorithm gives cells as they are generated in the group, this can depend on the generators. But they most likely do not give a nice display. So our first section below shows how to plot a user defined portion of a tessellation.

Our second section concerns the case when a practical tiling method does not give good borders, unlike as in the graphic above where borders define adjacent octagonal cells. We can distinguish cells by color with adjacent cells a different color. There is some mathematics here, specifically the four color theorem which says that 4 colors will suffice for any polygonal tessellation. We will see that unless triangles are involved 3 colors should be enough.

4.7.1 Choosing cells in a tessellation

The following algorithm will tessellate with a number at the centroid of each cell .

```
Options[groupATessL] = {returnAssoc → False};
groupATessL[G_, tas_, P_, OptionsPattern[]] := Module[{tabA, n},
  n = Length[G];
  tabA = <|Table[i → {TasTF[G⟦i⟧, tas]@P, pcentroid[TasTF[G⟦i⟧, tas]@P]}, {i, n}]|>;
  If[OptionValue[returnAssoc], Return[tabA]];
  Show[Table[Graphics[{{Black, Thickness[.005], Line[tabA[i]⟦1⟧]},
      {Black, Text[Style[i, 10], tabA[i]⟦2⟧]}}], {i, n}]]]
```

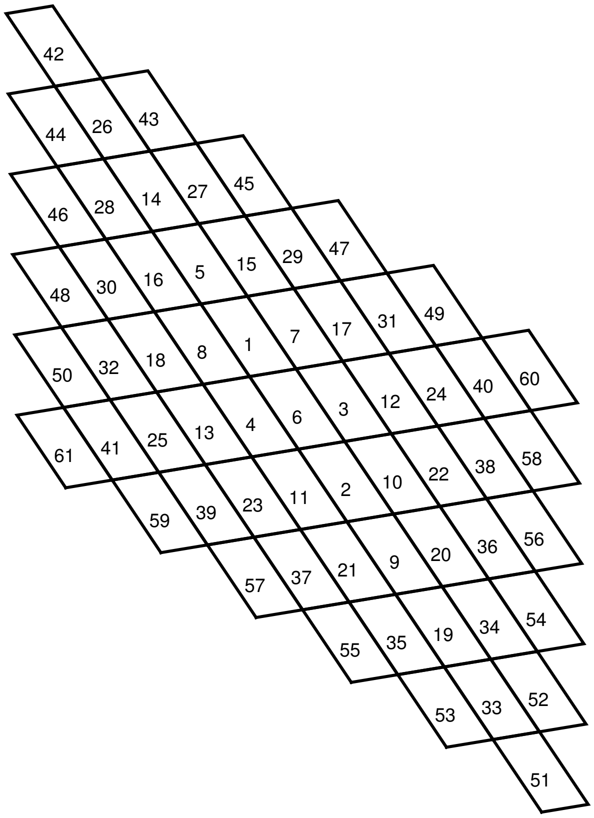
For our first example we use the parallelograms of our first example for Group I. Note our cell is line style, that the first and last points are the same.

```
In[ ]:= {p, q} = {{-1.8337165994212103` , 2.736646713237377` },
          {1.7521455008048097` , 0.2910803142189895` }}
Out[ ]= {{-1.83372, 2.73665}, {1.75215, 0.29108}}

In[ ]:= par = {{0, 0}, p, p + q, q, {0, 0}}
Out[ ]= {{0, 0}, {-1.83372, 2.73665}, {-0.0815711, 3.02773}, {1.75215, 0.29108}, {0, 0}}

In[ ]:= τp = TranslationTransform[p];
        τq = TranslationTransform[q];

In[ ]:= c = pcentroid[par]
Out[ ]= {-0.0407855, 1.51386}

In[ ]:= G1a = discreteTransGroup[<|1 → τp, 2 → InverseTF[τp], 3 → τq, 4 → InverseTF[τq]|>,
          c, 2, 5, returnGroup → True]

» number of group elements calculated 61

Out[ ]= {{1}, {2}, {3}, {4}, {1, 1}, {1, 2}, {1, 3}, {1, 4}, {2, 2}, {2, 3}, {2, 4}, {3, 3}, {4, 4}, {1, 1, 1},
  {1, 1, 3}, {1, 1, 4}, {1, 3, 3}, {1, 4, 4}, {2, 2, 2}, {2, 2, 3}, {2, 2, 4}, {2, 3, 3},
  {2, 4, 4}, {3, 3, 3}, {4, 4, 4}, {1, 1, 1, 1}, {1, 1, 1, 3}, {1, 1, 1, 4}, {1, 1, 3, 3},
  {1, 1, 4, 4}, {1, 3, 3, 3}, {1, 4, 4, 4}, {2, 2, 2, 2}, {2, 2, 2, 3}, {2, 2, 2, 4},
  {2, 2, 3, 3}, {2, 2, 4, 4}, {2, 3, 3, 3}, {2, 4, 4, 4}, {3, 3, 3, 3}, {4, 4, 4, 4},
  {1, 1, 1, 1, 1}, {1, 1, 1, 1, 3}, {1, 1, 1, 1, 4}, {1, 1, 1, 3, 3}, {1, 1, 1, 4, 4},
  {1, 1, 3, 3, 3}, {1, 1, 4, 4, 4}, {1, 3, 3, 3, 3}, {1, 4, 4, 4, 4}, {2, 2, 2, 2, 2},
  {2, 2, 2, 2, 3}, {2, 2, 2, 2, 4}, {2, 2, 2, 3, 3}, {2, 2, 2, 4, 4}, {2, 2, 3, 3, 3},
  {2, 2, 4, 4, 4}, {2, 3, 3, 3, 3}, {2, 4, 4, 4, 4}, {3, 3, 3, 3, 3}, {4, 4, 4, 4, 4}}
```

Note that the group elements are all n-tuples, $n \leq 5$, of the integers 1,2,3,4.

In[ ]:=
```
groupATessL[G1a, <|1 → τp, 2 → InverseTF[τp], 3 → τq, 4 → InverseTF[τq]|>, par]
```

Out[ ]=



To get a nice parallelogram tessellations we choose cells

In[ ]:=
```
Sa = {50, 32, 18, 8, 1, 7, 17, 31, 49, 41, 25,
   13, 4, 6, 3, 12, 24, 40, 59, 39, 23, 11, 2, 10, 22, 38, 58}
```

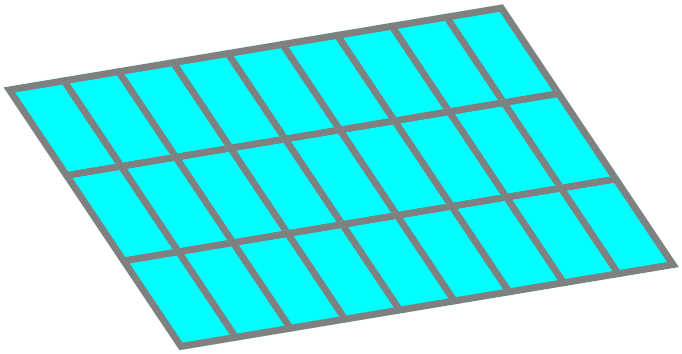
Out[ ]=
```
{50, 32, 18, 8, 1, 7, 17, 31, 49, 41, 25, 13,
  4, 6, 3, 12, 24, 40, 59, 39, 23, 11, 2, 10, 22, 38, 58}
```

We now use the group association to give a custom tessellation .

In[ ]:=
```
Aa = groupATessL[
    G1a, <|1 → τp, 2 → InverseTF[τp], 3 → τq, 4 → InverseTF[τq]|>, par, returnAssoc → True];
```

```
In[ ]:= Graphics[{Cyan, EdgeForm[{Gray, Thickness[.01]}], Polygon[Table[Aa[i][[1]], {i, Sa}]]}]
```

Out[ ]=



Here is a more complicated example with a compound tessellation.

```
τd = TranslationTransform[{1, 1.}];
```

```
In[ ]:= oct = {{0.7071067811865475` , -0.2928932188134522` },
    {0.7071067811865475` , 0.2928932188134525` },
    {0.2928932188134524` , 0.7071067811865475` }, {-0.2928932188134526` ,
     0.7071067811865475` }, {-0.7071067811865477` , 0.29289321881345237` },
    {-0.7071067811865475` , -0.2928932188134526` }, {-0.29289321881345226` ,
     -0.7071067811865477` }, {0.29289321881345276` , -0.7071067811865475` },
    {0.7071067811865475` , -0.2928932188134522` }};
```

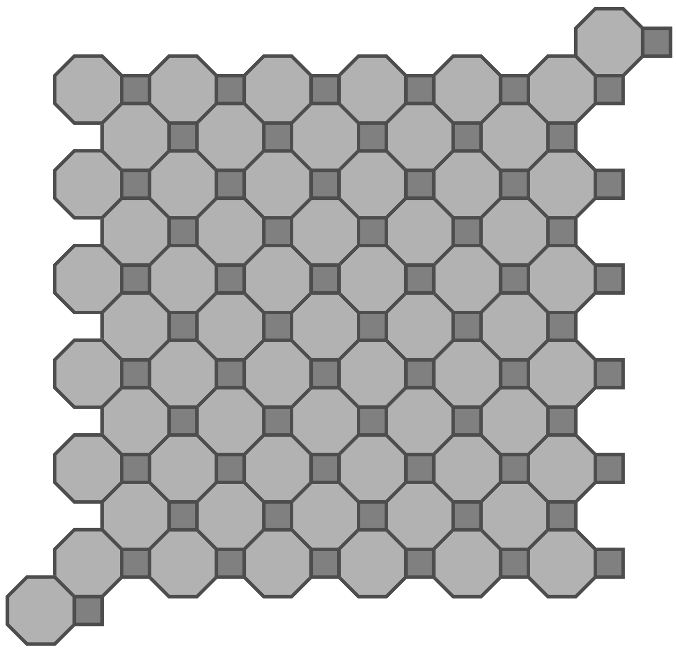
```
In[ ]:= sqr = {{0.7071067811865475` , -0.2928932188134522` },
    {1.2928932188134525` , -0.2928932188134522` },
    {1.2928932188134525` , 0.2928932188134525` }, {0.7071067811865475` ,
     0.2928932188134525` }, {0.7071067811865475` , -0.2928932188134522` }};
```

```
In[ ]:= G3 = discreteTransGroup[<|1 → τd, 2 → InverseTF[τd], 3 → ρh|>,
    {0, -0}, 2, 6, returnGroup → True];
```

» number of group elements calculated 63

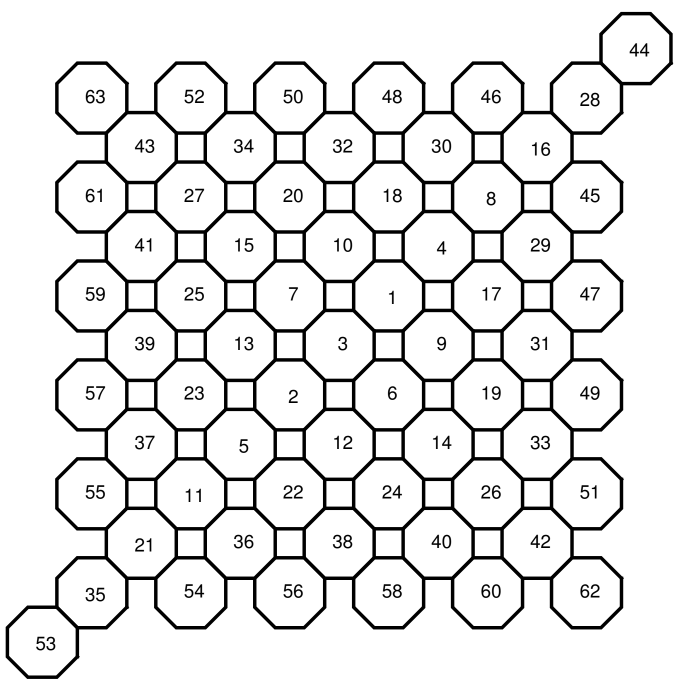
In[ ]:=
```
Show[groupTessellate [G3, <| 1 → τd, 2 → InverseTF [τd], 3 → ρh|> ,
  oct, GrayLevel[.7], GrayLevel[.3]], groupTessellate [
  G3, <| 1 → τd, 2 → InverseTF [τd], 3 → ρh|> , sqr, GrayLevel[.5], GrayLevel[.3]]]
```

Out[ ]=

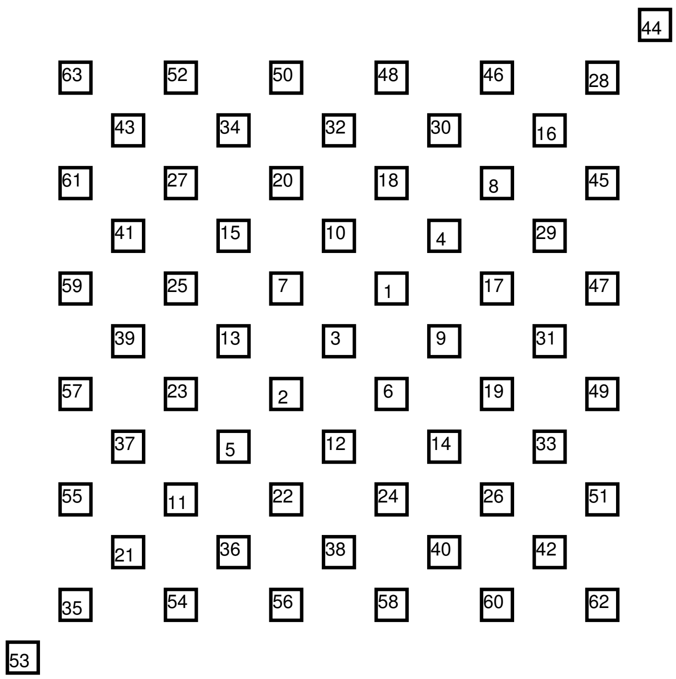


In[ ]:=
```
groupATessL [G3, <| 1 → τd, 2 → InverseTF [τd], 3 → ρh|> , oct]
```

Out[ ]=



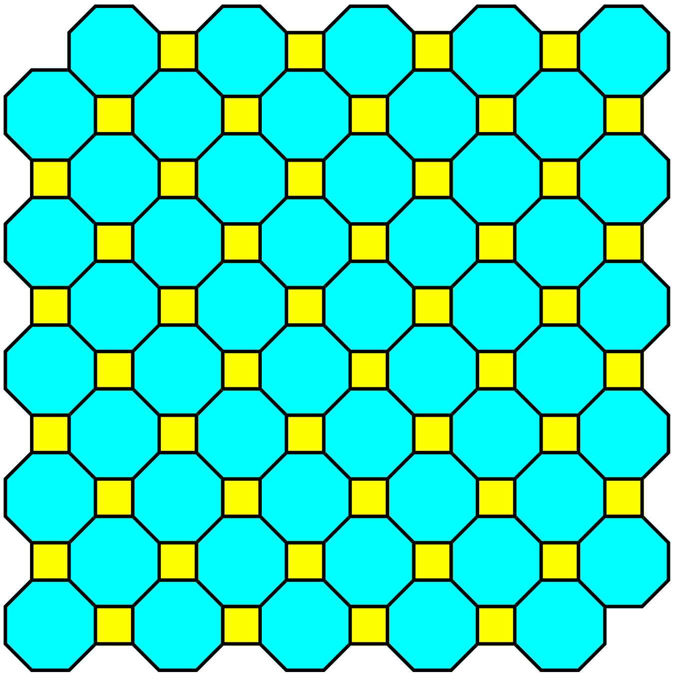
```
In[ ∘ ]:= S = Range[1, 63];
S1 = Delete[S, Partition[{44, 35, 54, 56, 58, 60, 62, 53, 63, 61, 59, 57, 55, 35}, 1]]

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

In[ ∘ ]:= groupATessL[G3, <|1 → τd, 2 → InverseTF[τd], 3 → ρh|>, sqr, returnAssoc → False]

Out[ ∘ ]=
```




```
In[ ∘ ]:= S2 = Delete[S,
   Partition[{53, 44, 35, 54, 56, 58, 60, 62, 28, 45, 47, 49, 51, 62, 62, 63, 42}, 1]]

Out[ ∘ ]= {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25,
  26, 27, 29, 30, 31, 32, 33, 34, 36, 37, 38, 39, 40, 41, 43, 46, 48, 50, 52, 55, 57, 59, 61}

In[ ∘ ]:= A1 = groupATessL[G3, <|1 → τd, 2 → InverseTF[τd], 3 → ρh|>, oct, returnAssoc → True];

In[ ∘ ]:= A2 = groupATessL[G3, <|1 → τd, 2 → InverseTF[τd], 3 → ρh|>, sqr, returnAssoc → True];
```

In[ ]:=
```
Graphics[{{Cyan, EdgeForm[{Black, Thickness[.005]}], Polygon[Table[A1[i][[1]], {i, S1}]]},
  {Yellow, EdgeForm[{Black, Thickness[.005]}], Polygon[Table[A2[i][[1]], {i, S2}]]}}]
```

Out[ ]=



This is much nicer.

We want to clean up our example for group 11.

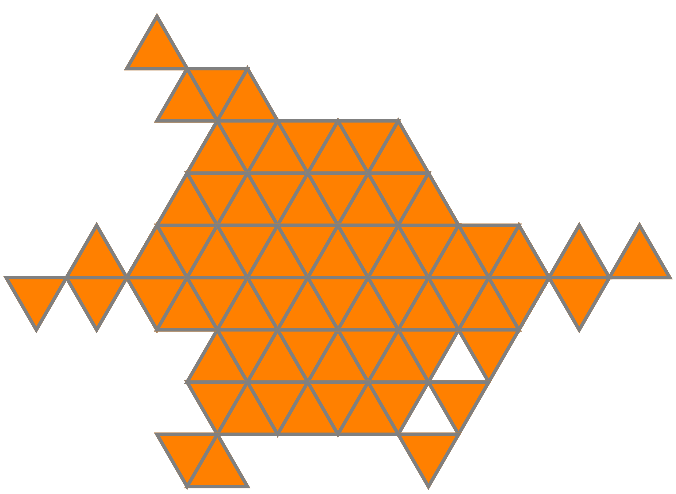
In[ ]:=
```
etri = {{0, 0}, {1, 0}, {.5, Sqrt[3.]/2}, {0, 0}};
```

In[ ]:=
```
G11 = discreteTransGroup[<|1 → τh, 2 → σ3, 3 → η0, 4 → ρh|>,
   {.2213, .3124}, 2, 5, returnGroup → True];
```

» number of group elements calculated 179

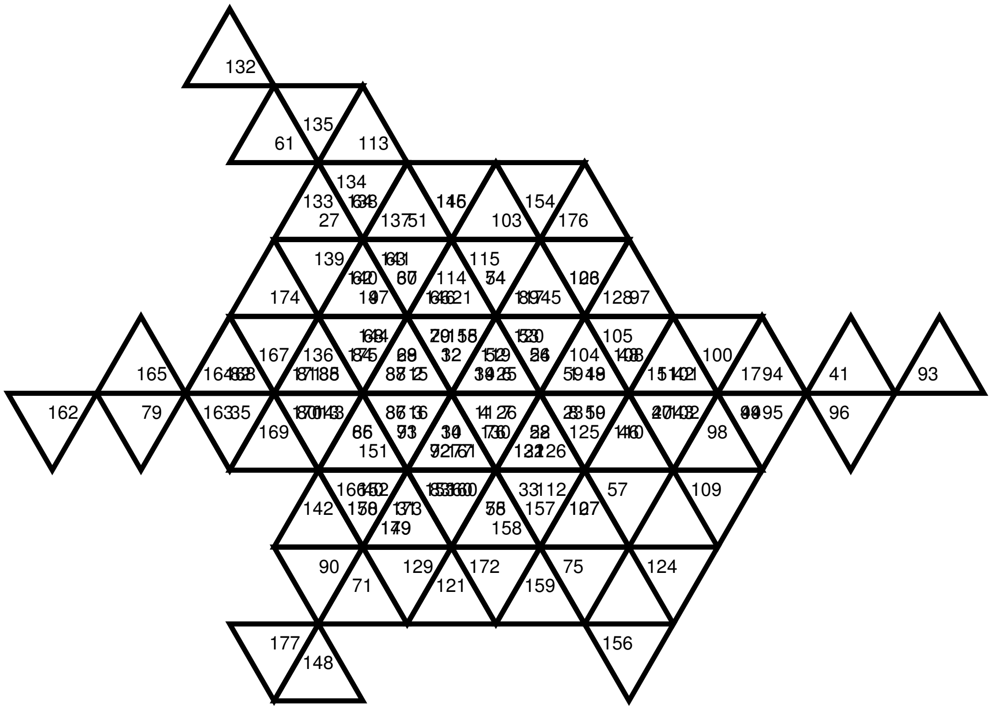
In[ ]:=
```
groupTessellate[G11, <|1 → τh, 2 → σ3, 3 → η0, 4 → ρh|>, etri, Orange, Gray]
```

Out[ ]=



In[ ]:=
```
groupATessL[G11, <|1 → τh, 2 → σ3, 3 → η0, 4 → ρh|>, etri]
```

Out[ ]=



This is very messy as there are several group elements giving the same cell. G11 is the symmetry group but we can use a simpler construction group. We find another translation in G11

In[ ]:=
```
τe = Chop[TasTF[{2, 1, 2, 2}, <|1 → τh, 2 → σ3, 3 → η0, 4 → ρh|>]]
```

Out[ ]= TransformationFunction[

$$\left(\begin{array}{cc|c} 1. & 0. & -0.5 \\ 0. & 1. & 0.866025 \\ \hline 0. & 0. & 1. \end{array}\right)$$

]

Then the level 7 group

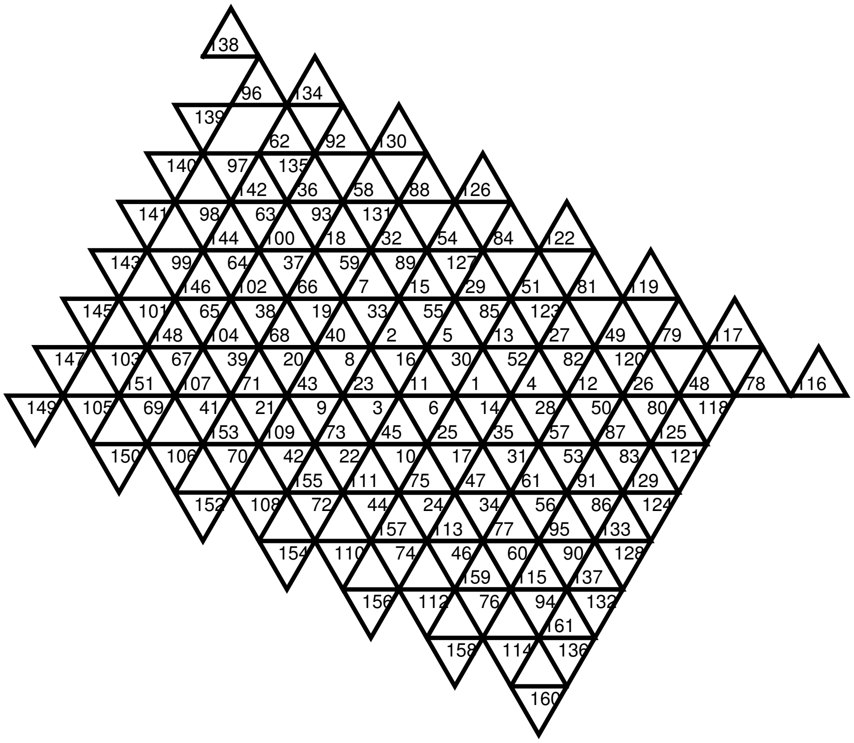
In[ ]:=
```
G11rev = discreteTransGroup[<|1 → τh, 2 → τe, 3 → η0|>,
   {.1259, .2134}, 3, 7, returnGroup → True];
```

» number of group elements calculated 161

gives

In[∘]:= `groupATessL [G11rev , <| 1 → τh, 2 → τe, 3 → η0 |> , etri]`

Out[∘]=



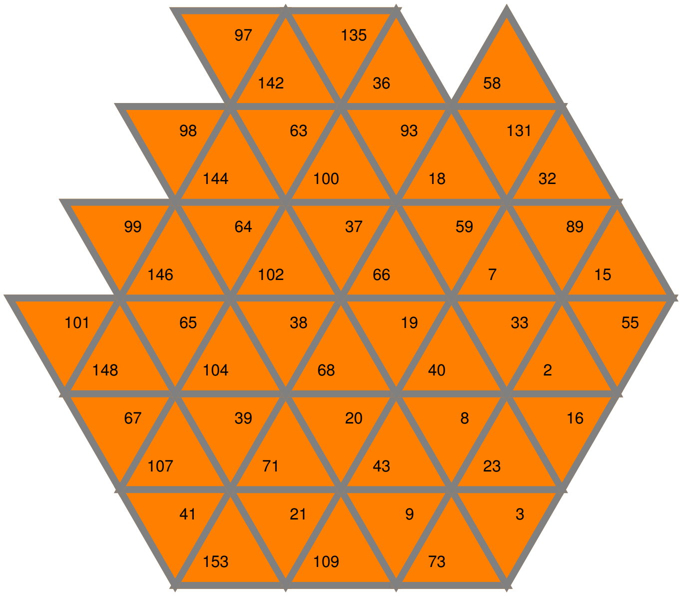
In[∘]:= `B1 = groupATessL [G11rev , <| 1 → τh, 2 → τe, 3 → η0 |> , etri , returnAssoc → True];`

It is a help picking out cells to first construct a working diagram which would look like this.

In[∘]:=
```
S3 = {153, 107, 148, 109, 41, 67, 101, 73, 21, 9, 3, 71, 43, 23, 39,
   20, 8, 16, 104, 68, 40, 2, 65, 38, 19, 33, 55, 146, 102, 66, 7, 15, 99,
   64, 37, 59, 89, 144, 100, 18, 32, 98, 97, 135, 63, 142, 93, 36, 131, 58};
```

In[ ]:=
```
Graphics[{{Orange, EdgeForm[{Gray, Thickness[.01]}], Polygon[Table[B1[i][[1]], {i, S3}]]},
  {Black, Table[Text[i, B1[i][[2]]], {i, S3}]}}]
```

Out[ ]=



We have 3 missing cells in the graphic we want. We could go to level 8, but an alternate strategy is to use our association B1 which identifies the numbered cells. We notice that the missing cells are translates of cells we have so the missing cells are given by

In[ ]:=
```
τhi = InverseTF[τh]
```

Out[ ]= TransformationFunction$\left[\begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}\right]$

In[ ]:=
```
c1 = τhi@B1[144][[1]]
```

Out[ ]= {{-4.5, 2.59808}, {-3.5, 2.59808}, {-4., 3.4641}, {-4.5, 2.59808}}

In[ ]:=
```
c2 = τhi@B1[142][[1]]
```

Out[ ]= {{-4., 3.4641}, {-3., 3.4641}, {-3.5, 4.33013}, {-4., 3.4641}}

In[ ]:=
```
c3 = τh@B1[135][[1]]
```

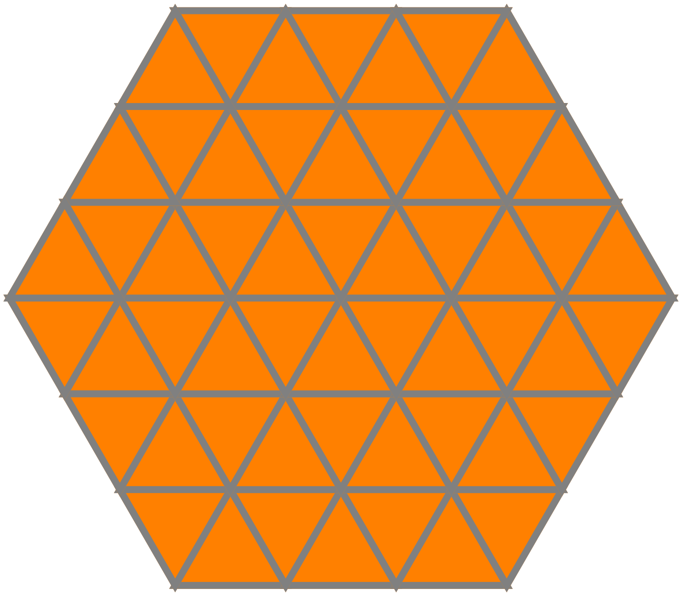
Out[ ]= {{-0.5, 4.33013}, {-1.5, 4.33013}, {-1., 3.4641}, {-0.5, 4.33013}}

In[ ]:=
```
c4 = τhi@B1[146][[1]]
```

Out[ ]= {{-5., 1.73205}, {-4., 1.73205}, {-4.5, 2.59808}, {-5., 1.73205}}

In[ ]:= Now the desired graphic is

```
In[ ]:= Graphics[{{Orange, EdgeForm[{Gray, Thickness[.01]}],
    Polygon[Table[B1[i][[1]], {i, S3}]], Polygon[{c1, c2, c3, c4}]}}]
```

Out[ ]=


4.7.2 Coloring part of a tessellation.

For some applications the cell edges are either not desirable or possible. One such example is using a stencil to place a graphic on a t-shirt as in the picture below of your author’s daughter, the maker of the shirt (www.makewithtanya.com).

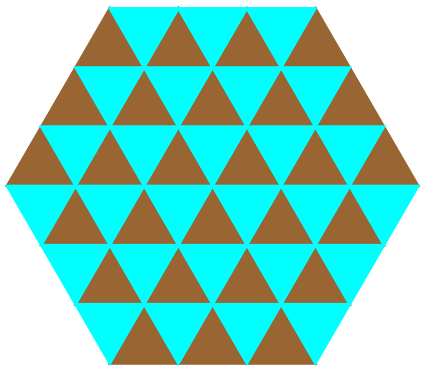
Here adjacent cells must have a different color. We a saved from a lot of colors by the 4-color theorem that says this can be done with at most 4 colors since our cells are polygonal. In fact, for most tessellations 3 colors suffice. It is not clear that this holds for a symmetric coloring but in my examples it does. We may use the groupATessL association to facilitate this.

Our first example is the hexagon above without edges using `groupTessL`.

```
In[ ]:= T1 = {153, 109, 73, 107, 71, 43, 23, 148, 104,
          68, 40, 2, 146, 102, 66, 7, 15, 144, 100, 18, 32, 142, 36, 58}
        T2 = Complement[S3, T1]
```

```
In[ ]:= Graphics[{{Brown, Polygon[Table[B1[i][[1]], {i, T1}]], Polygon[{c1, c2, c4}]},
          {Cyan, Polygon[Table[B1[i][[1]], {i, T2}]], Polygon[{c3}]}}]
```

Out[ ]=



Here we need only 2 colors . A tessellation needing 3 colors we have a figure I call a pseudoTriangle with symmetry group IX . To avoid duplication of cells we will use a construction group of type group I.

```
In[ ]:= τw = TransformationFunction[( 1.`  0.` | 0.9954`
                                      0.`  1.` | 0.5746944579513535`
                                      0.`  0.` | 1.` )];
```

```
In[ ]:= τu = TransformationFunction[( 1.`  0.` | 0.`
                                      0.`  1.` | 1.149388915902707`
                                      0.`  0.` | 1.` )];
```

```
corner1 = {{-2/3, 0.5773502691896257`}, {-0.66`, 0.4`},
   {-0.64`, 0.3`}, {-0.59`, 0.2`}, {-0.5`, 0.1`}, {-0.3333333333333333`, 0.`}};
```

```
In[ ]:= indent2 = ρh@* σ6@corner1
Out[ ]= {{-0.333333, 0.}, {-0.17641, 0.0829016}, {-0.0798076, 0.115581},
         {0.0317949, 0.12228}, {0.163397, 0.0943376}, {0.333333, 0.}}
```

```
In[ ]:= tile0 = Join[corner1, indent2, σ3@corner1,
           σ3@indent2, σ3@* σ3@corner1, σ3@* σ3@indent2];
```

This tile has rotation symmetry σ3 but also reflection ρv and hence 2 other reflections.

```
In[ ]:= Graphics[{Orange, Polygon[tile0]}, ImageSize → Tiny]
```

Out[ ]=

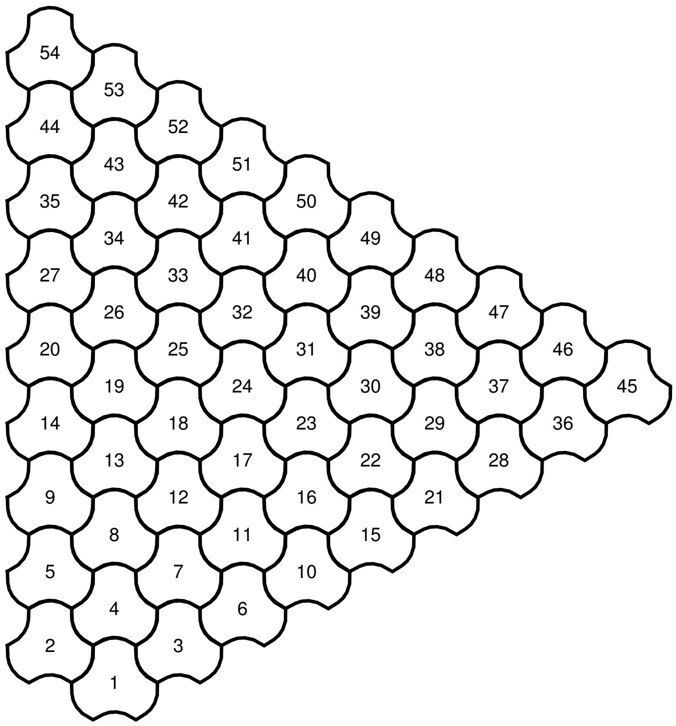
In[ ]:=
```
G1uw = finiteTransGroup [<|1 → τw, 2 → τu|>, {.2174, 3246}, 9];
```

» number of group elements calculated 54

To use stencils to make a shirt we will need three colors, we number the cells of a level 9 tessellation.

In[ ]:=
```
groupATessL [G1uw, <|1 → τw, 2 → τv|>, tile0]
```

Out[ ]=



In[ ]:=
```
APT = groupATessL [G1uw, <|1 → τw, 2 → τv|>, tile0, returnAssoc → True];
```

In[ ]:=
```
C1 = {9, 12, 16, 21, 19, 24, 30, 37, 27, 33, 40, 48}
```

Out[ ]= {9, 12, 16, 21, 19, 24, 30, 37, 27, 33, 40, 48}

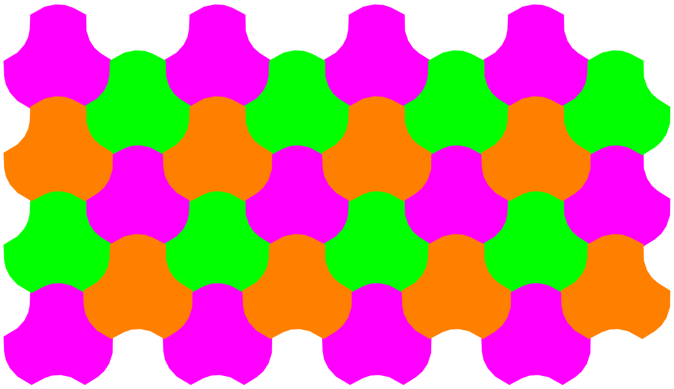
In[ ]:=
```
C2 = {14, 18, 23, 29, 26, 32, 39, 47}
```

Out[ ]= {14, 18, 23, 29, 26, 32, 39, 47}

In[ ]:=
```
C3 = {20, 25, 31, 38, 13, 17, 22, 28}
```

Out[ ]= {20, 25, 31, 38, 13, 17, 22, 28}

In[ ]:=
```
Show[Graphics[{Magenta, Polygon[Table[APT[i][[1]], {i, C1}]]}],
 Graphics[{Green, Polygon[Table[APT[i][[1]], {i, C2}]]}],
 Graphics[{Orange, Polygon[Table[APT[i][[1]], {i, C3}]]}]]
```

Out[ ]=


The next example comes from the entrance porch of the Ridgefield CT recreation center that I visit frequently.

This did strike me as somewhat unusual, with only translation symmetry, but until writing this subsec - tion I did not grasp the importance: to use stencils I need 4 colors. To construct this

```
RC1 = {{0, 0}, {0, 2}, {2, 2}, {2, 0}, {0, 0}};
RC2 = {{2, 1}, {2, 2}, {3, 2}, {3, 1}, {2, 1}};
RC3 = {{3, 1}, {3, 2}, {5, 2}, {5, 1}, {3, 1}};
τr1 := TranslationTransform [{2, -1}];
τr2 := TranslationTransform [{3, 2}];.
```

Since there are no rotations or reflections we need the inverses so

In[ ]:=
```
τr1i = InverseTF[τr1]
τr2i = InverseTF[τr2]
```

Out[ ]= TransformationFunction $\left[\begin{pmatrix} 1 & 0 & -2 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}\right]$

Out[ ]= TransformationFunction $\left[\begin{pmatrix} 1 & 0 & -3 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{pmatrix}\right]$

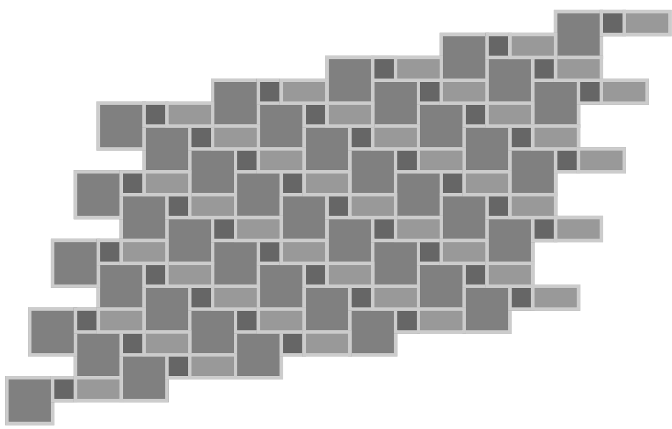
We have then the transformation group of type Group I

In[ ]:=
```
GRC = discreteTransGroup [<| 1 → τr1, 2 → τr1i, 3 → τr2, 4 → τr2i |>,
   {2.1212, -3.2456}, 3, 4, returnGroup → True];
```

» number of group elements calculated 41

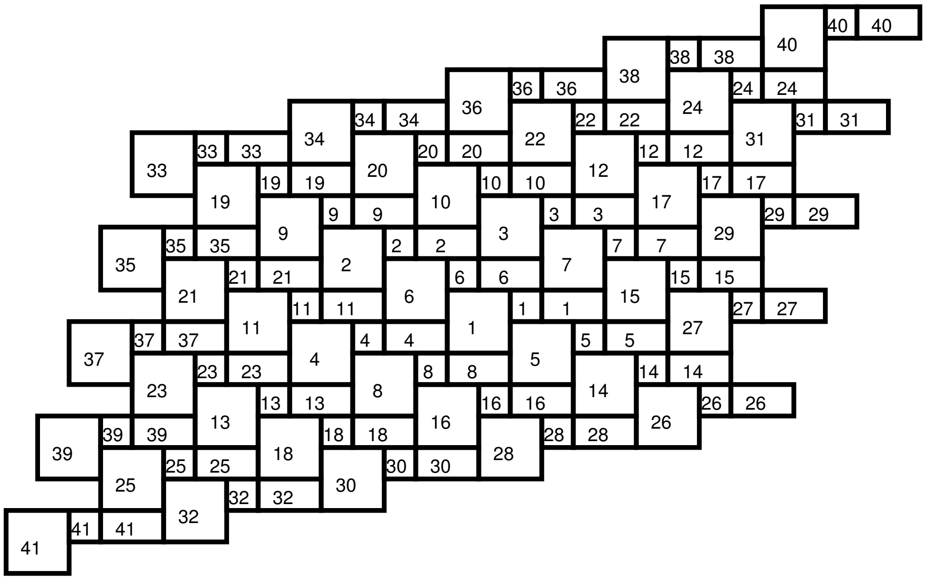
Using edges this is easy

In[ ]:=
```
Show[groupTessellate [GRC, <| 1 → τr1, 2 → τr1i, 3 → τr2, 4 → τr2i |>, RC1, GrayLevel[.5],
  GrayLevel[.8]], groupTessellate [GRC, <| 1 → τr1, 2 → τr1i, 3 → τr2, 4 → τr2i |>,
  RC2, GrayLevel[.4], GrayLevel[.8]], groupTessellate [
  GRC, <| 1 → τr1, 2 → τr1i, 3 → τr2, 4 → τr2i |>, RC3, GrayLevel[.6], GrayLevel[.8]]]
```

Out[ ]=


But our large squares share part of a boundary so adjacent ones need to be another color. Also we would like a rectangular paving rather than this parallelogram. So we use groupATessL. Most details will be omitted here. We get an outline with numbered cells

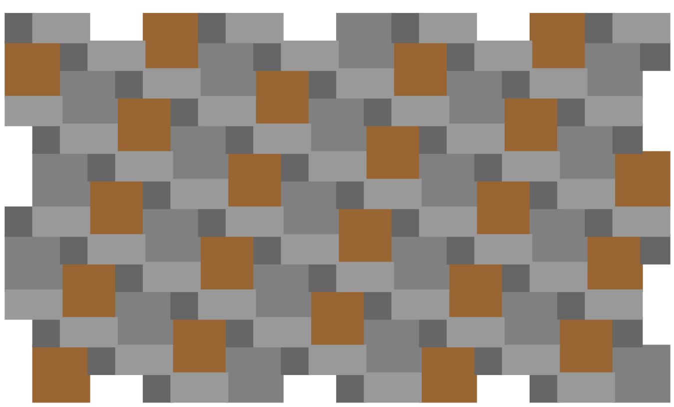
```
Show[groupATessL[GRC, <|1 → τr1, 2 → τr1i, 3 → τr2, 4 → τr2i|>, RC1],
 groupATessL[GRC, <|1 → τr1, 2 → τr1i, 3 → τr2, 4 → τr2i|>, RC2],
 groupATessL[GRC, <|1 → τr1, 2 → τr1i, 3 → τr2, 4 → τr2i|>, RC3]]
```



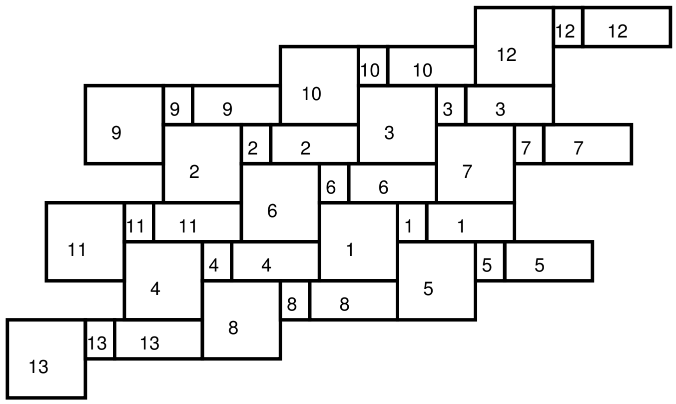
The numbers refer to the same transformation applied to the different polygons. We pick out the ones we want but separate the larger squares in to gray and brown. We get 4 associations with slightly different cells chosen. We put the data in a hidden cell. These can be accessed by those using the notebook form of this chapter.

Or resulting tiling is

```
Show[Graphics[{{Brown, Polygon[Table[RCA[i][[1]], {i, SB}]]},
   {GrayLevel[.5], Polygon[Table[RCA[i][[1]], {i, SG}]]}, {GrayLevel[.4],
    Polygon[Table[RCB[i][[1]], {i, SS}]]}, {GrayLevel[.6], Polygon[Table[RCC[i][[1]], {i, SR}]]}}]]
```



Intuitively making this tiling more complicated should require more colors. But in fact that is not the case. For example we could make the small square into a smaller rectangle and the rectangles larger so that those who meet at a vertex would have a common edge piece. Leaving out details the outline might be



with adjacent rectangles needing to be different colors . But that can be also easily achieved more elegantly than this example. We leave coloring this with the same translation symmetries as the example above to the reader.